Note : The formulation for vortex panel method is taken from the book " Foundations of Aerodynamics: Bases of Aerodynamic Design" by Arnold M. Kuethe and Chuen-Yen Chow

Vortex Panel Method

The vortex panel method is a discretized computational method for vortex sheet. Vortex panel method is one of the panel methods which are used for solving incompressible, potential and lifting flows over arbitrary bodies like airfoil. The mathematical model of vortex panel method along with the matlab code is described in this section.

The airfoil is represented by a closed polygon of vortex panels. In this method, the circulation density on each panel varies linearly from one corner to the other and is continuous across the corner, as indicated in Fig 1. The *n* number of planes are assumed to be planar and are named in clockwise direction, starting from the trailing edge and ending at the trailing edge.

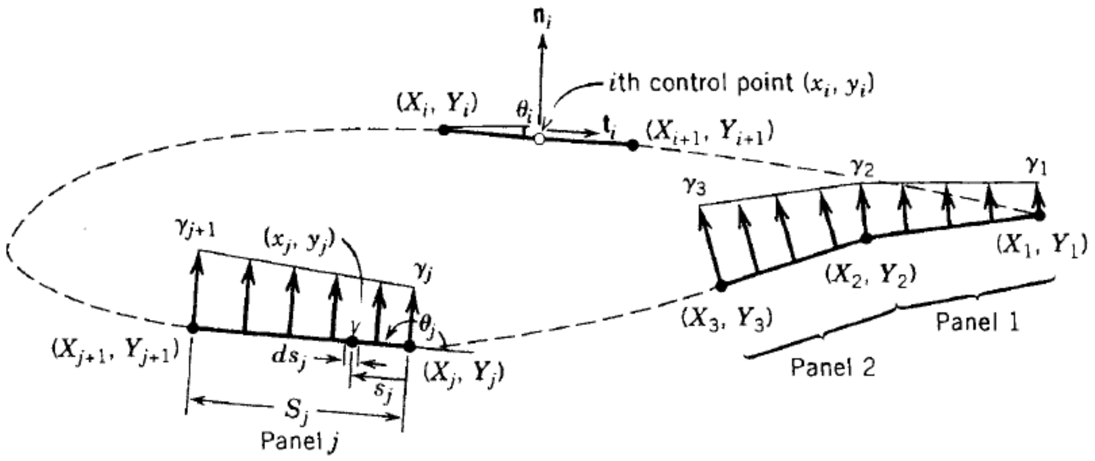

Fig. 1 Vortex panels with linear varying strength distributed over the surface of an airfoil

The following program creates nodes on airfoil

```
nacaseries = input('Enter the 4-digit naca series = ');
 c = input('Enter the chord length = ');
 n = input('Enter the number of nodes = ');
 a = input('Enter the angle of attack in degree = ');
 s = num2str(nacaseries);

 % creating nodes on airfoil
if numel(s)==2
    s1 = str2double(s(1));
 s2 = str2double(s(2));
 m=0;p=0;t=(10*s1+s2)*0.01;
else
    s1 = str2double(s(1));
 s2 = str2double(s(2));
 s3 = str2double(s(3));
 s4 = str2double(s(4));
    m = s1*0.01; p = s2*0.1 ; t = (10*s3+s4)*0.01;
end


for i= n:-1:1

    theta = (i-n)*2*pi/n;
    x = 0.5*c*(1+cos(theta));

if(x/c)<p
    yc(n+1-i) = m*c/p^2*(2*p*(x/c)-(x/c)^2);
    dydx(n+1-i) = (2*m/p^2)* (p-x/c);
    beta(n+1-i) = atan(dydx(n+1-i));
else
    yc(n+1-i) = m*c/(1-p)^2 * ((1-2*p)+2*p*(x/c)-(x/c)^2);
    dydx(n+1-i) = (2*m/(1-p)^2)* (p-x/c);
    beta(n+1-i) = atan(dydx(n+1-i));
end
yt=5*t*c*(0.2969*sqrt(x/c)-0.1260*(x/c)...
    -0.3516*(x/c)^2+0.2843*(x/c)^3-0.1036*(x/c)^4);

if(i<(0.5*n+1))
    xa(n+1-i)=x - yt*sin(beta(n+1-i));
    ya(n+1-i)=yc(n+1-i)+yt*cos(beta(n+1-i));
else
    xa(n+1-i)=x + yt*sin(beta(n+1-i));
    ya(n+1-i)=yc(n+1-i)-yt*cos(beta(n+1-i));
end

end
xa(n+1)= c;
ya(n+1) = 0;
yc(n+1) = 0;  % trailing edge
```

The following program creates control points on each panel

```
%This loop calculates the location of the control points

for i = 1:n
    xmid(i) = (xa(i)+xa(i+1))/2;
    ymid(i) = (ya(i)+ya(i+1))/2;
    Sj(i) = sqrt((xa(i+1)-xa(i))^2+(ya(i+1)-ya(i))^2);%array of panel lengths
    phi(i) = atan2((ya(i+1)-ya(i)),(xa(i+1)-xa(i)));
    rhs(i)= sin(phi(i)-alpha);% RHS of equation 4
end
```

In the presence of a uniform flow V_∞, at an angle of attack α and n vortex panels, the velocity potential at the ith control point (x_i, y_i) due to free stream and all other panels is

$$\phi\left(x_i, y_i\right) = V_\infty\left(x_i \cos\alpha + y_i \sin\alpha\right) - \sum_{j=1}^{n}\int_j \frac{\gamma(s_j)}{2\pi}\tan^{-1}\left(\frac{y_i - y_j}{x_i - x_j}\right) ds_j \tag{1}$$

Where,

$$\gamma(s_j) = \gamma_j + (\gamma_{j+1} - \gamma_j)\frac{s_j}{S_j} \tag{2}$$

Applying boundary condition of impermeability, we get

$$\frac{\partial}{\partial n_i}\phi(x_i, y_i) = 0; \quad i = 1, 2, \ldots, n \tag{3}$$

Carrying out the involved differentiation and integration, we obtain

$$\sum_{j=1}^{n}\left(C_{n1ij}\gamma_j^{'} + C_{n2ij}\gamma_{j+1}^{'}\right) = \sin(\theta_i - \alpha); \qquad i = 1, 2, \ldots, n \tag{4}$$

Here, $\gamma^{'} = \gamma / 2\pi V_\infty$ is a dimensionless circulation density , θ_i is the orientation angle of the ith panel measured from the x axis to the panel surface , and the coefficients are expressed as follows:

$$C_{n1ij} = 0.5DF + CG - C_{n2ij}$$
$$C_{n2ij} = D + 0.5QF / S_j - (AC + DE)G / S_j$$

The constants are defined as :

$$A = -\left(x_i - X_j\right)\cos\theta_j - \left(y_i - Y_j\right)\sin\theta_j$$

$$B = \left(x_i - X_j\right)^2 + \left(y_i - Y_j\right)^2$$

$$C = \sin(\theta_i - \theta_j)$$

$$D = \cos(\theta_i - \theta_j)$$

$$E = \left(x_i - X_j\right)\sin\theta_j - \left(y_i - Y_j\right)\cos\theta_j$$

$$F = \ln\left(1 + \frac{S_j^2 + 2AS_j}{B}\right)$$

$$G = \tan^{-1}\left(\frac{ES_j}{B + AS_j}\right)$$

$$P = \left(x_i - X_j\right)\sin(\theta_i - 2\theta_j) + \left(y_i - Y_j\right)\cos(\theta_i - 2\theta_j)$$

$$Q = \left(x_i - X_j\right)\cos(\theta_i - 2\theta_j) - \left(y_i - Y_j\right)\sin(\theta_i - 2\theta_j)$$

$$S_j = \sqrt{\left(X_{j+1} - X_j\right)^2 + \left(Y_{j+1} - Y_j\right)^2}$$

,

For $i = j$, the coefficients become:

$$C_{n1ii} = -1 \quad \text{and} \quad C_{n1ii} = 1$$

```
% This loop calculates the coefficients to find the matrix Aij for equation
% 6.Variables are as given in report.
for i = 1:n
    for j = 1:n
        if i==j
            cn1(i,j) = -1;
            cn2(i,j) = 1;
        else
        A=-(xmid(i)-xa(j))*cos(phi(j))-(ymid(i)-ya(j))*sin(phi(j));
        B = (xmid(i)-xa(j))^2+(ymid(i)-ya(j))^2;
        C = sin(phi(i)-phi(j));
        D = cos(phi(i)-phi(j));
        E = (xmid(i)-xa(j))*sin(phi(j))-(ymid(i)-ya(j))*cos(phi(j));
        F = log(1+(Sj(j)^2+2*A*Sj(j))/B);
        G = atan2((E*Sj(j)),(B+A*Sj(j)));
        P = (xmid(i)-xa(j))*sin(phi(i)-2*phi(j))+(ymid(i)-ya(j))*cos(phi(i)-
2*phi(j));
        Q = (xmid(i)-xa(j))*cos(phi(i)-2*phi(j))-(ymid(i)-ya(j))*sin(phi(i)-
2*phi(j));
       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);

        end
    end
end
```

which describe the self-induced normal velocity at the ith control point.

To ensure a smooth flow at the trailing edge ,the kutta condition is applied as follows:

(kutta condition demands that the strength of the vorticity at the trailing edge be zero)

$$\gamma_1^{'} + \gamma_{n+1}^{'} = 0 \tag{5}$$

There are altogether n+1 equations after combining equations (4) and (5) which are sufficient to solve n+1 unknown $\gamma_j^{'}$ values. Rewriting this system of simultaneous equations in a more convenient form, we get

$$\sum_{j=1}^{n+1} A_{nij}\gamma_j^{'} = \mathrm{RHS}_i; \qquad i = 1, 2, ..., n+1 \tag{6}$$

in which , for $i < n+1$:

$$\begin{aligned} A_{n_{i1}} &= C_{n1_{i1}} \\ A_{n_{ij}} &= C_{n1_{ij}} + C_{n2_{ij-1}}; \quad j = 2, 3, ..., n \\ A_{n_{in+1}} &= C_{n_{2in}} \\ \mathrm{RHS}_i &= \sin(\theta_i - \alpha) \end{aligned}$$

And for $i = n+1$:

$$\begin{aligned} A_{n_{i1}} &= A_{n_{in+1}} = 1 \\ A_{n_{ii}} &= 0; \quad j = 2, 3, ..., n \\ \mathrm{RHS}_i &= 0 \end{aligned}$$

Except for $i = n+1$, $A_{n_{ij}}$ is called the normal-velocity influence coefficients representing the influences of $\gamma_j^{'}$ on the normal velocity at the ith control point.

```
%This loop calculates the coefficients Anij and Atij used in eq. 6.

for i=1:n
        an(i,1)=cn1(i,1);
        an(i,n+1)=cn2(i,n);
        for j=2:n
            an(i,j)=cn1(i,j)+cn2(i,(j-1));
        end
end
%kutta condition

  an(n+1,1)=1;
    an(n+1,n+1)=1;
    rhs(n+1)=0;
    for j=2:n
        an(n+1,j)=0;
    end

% calculating circulation density by solving linear equations given by
    % eq. 6
    g = an\rhs';
```

After solving the above equations we obtain the values of circulation densities γ' at each panel. We use these values to compute the velocity and pressure at the control points. At such point, the velocity has only tangential component at the panel surface because of the vanishing of the normal component there. If t_i designate the unit tangential vector on the $i\mathrm{th}$ panel, the local dimensionless velocity defined as $V_i = \left(\partial\phi/\partial t_i\right)/V_\infty$ can be computed as

$$V_i = \cos(\theta_i - \alpha) + \sum_{j=1}^{n}\left(C_{t1ij}\gamma'_j + C_{t2ij}\gamma'_{j+1}\right); \qquad i = 1, 2, \ldots, n \tag{7}$$

in which

$$C_{t1_{ij}} = 0.5CF - DG - C_{t2_{ij}}$$

$$C_{t2_{ij}} = C + 0.5PF/S_j + (AD - CE)G/S_j$$

$$C_{t1_{ij}} = C_{t2_{ij}} = \frac{\pi}{2}$$

```
% This loop calculates the coefficients to find the matrix Aij for equation
% 7.Variables are as given in report.
for i = 1:n
    for j = 1:n
        if i==j

            ct1(i,j) = pi/2;
            ct2(i,j) = pi/2;
        else
        A=-(xmid(i)-xa(j))*cos(phi(j))-(ymid(i)-ya(j))*sin(phi(j));
        B = (xmid(i)-xa(j))^2+(ymid(i)-ya(j))^2;
        C = sin(phi(i)-phi(j));
        D = cos(phi(i)-phi(j));
        E = (xmid(i)-xa(j))*sin(phi(j))-(ymid(i)-ya(j))*cos(phi(j));
        F = log(1+(Sj(j)^2+2*A*Sj(j))/B);
        G = atan2((E*Sj(j)),(B+A*Sj(j)));
        P = (xmid(i)-xa(j))*sin(phi(i)-2*phi(j))+(ymid(i)-ya(j))*cos(phi(i)-
2*phi(j));
        Q = (xmid(i)-xa(j))*cos(phi(i)-2*phi(j))-(ymid(i)-ya(j))*sin(phi(i)-
2*phi(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
```

Equation (7) can be further written as

$$V_i = \cos(\theta_i - \alpha) + \sum_{j=1}^{n+1} A_{t_{ij}} \gamma'_j; \qquad i = 1, 2, \ldots, n \tag{8}$$

where tangential-velocity influence coefficients are defined as follows:

$$A_{t_{i1}} = C_{t1_{i1}}$$
$$A_{t_{ij}} = C_{t1_{ij}} + C_{t2_{ij-1}}; \quad j = 2, 3, \ldots, n$$
$$A_{t_{in+1}} = C_{t_{2in}}$$

```
%This loop calculates the coefficients Anij and Atij used in
%eq. 8


for i=1:n
        an(i,1)=cn1(i,1);
        an(i,n+1)=cn2(i,n);
        at(i,1)=ct1(i,1);
        at(i,n+1)=ct2(i,n);
        for j=2:n
            an(i,j)=cn1(i,j)+cn2(i,(j-1));
            at(i,j)=ct1(i,j)+ct2(i,(j-1));
        end
end
```

The pressure coefficient at the ith control point is

$$C_{pi} = 1 - V_i^2 \tag{9}$$

```
% calculating tangential velocity and coefficent of pressure
      for i=1:n
        sum=0;
        for j=1:n+1;
            sum=sum+at(i,j)*g(j);
         end
        v(i) = (cos(phi(i)-alpha)+sum);
        cp(i) = 1-v(i)*v(i);
      end
```

The matlab code to compute coefficient of pressure using linearly varying strength vortex panel method is given below.

```
clc
clear all
close all
nacaseries = input('Enter the 4-digit naca series = ');
 c = input('Enter the chord length = ');
 n = input('Enter the number of nodes = ');
 a = input('Enter the angle of attack in degree = ');
 s = num2str(nacaseries);

 % creating nodes on airfoil
    s1 = str2double(s(1));
 s2 = str2double(s(2));
 s3 = str2double(s(3:4));

    m = s1*0.01; p = s2*0.1 ; t = s3*0.01;
end


for i= n:-1:1

    theta = (i-n)*2*pi/n;
    x = 0.5*c*(1+cos(theta));

if(x/c)<p
    yc(n+1-i) = m*c/p^2*(2*p*(x/c)-(x/c)^2);
    dydx(n+1-i) = (2*m/p^2)* (p-x/c);
    beta(n+1-i) = atan(dydx(n+1-i));
else
    yc(n+1-i) = m*c/(1-p)^2 * ((1-2*p)+2*p*(x/c)-(x/c)^2);
    dydx(n+1-i) = (2*m/(1-p)^2)* (p-x/c);
    beta(n+1-i) = atan(dydx(n+1-i));
end
yt=5*t*c*(0.2969*sqrt(x/c)-0.1260*(x/c)...
    -0.3516*(x/c)^2+0.2843*(x/c)^3-0.1036*(x/c)^4);

if(i<(0.5*n+1))
    xa(n+1-i)=x - yt*sin(beta(n+1-i));
    ya(n+1-i)=yc(n+1-i)+yt*cos(beta(n+1-i));
else
    xa(n+1-i)=x + yt*sin(beta(n+1-i));
    ya(n+1-i)=yc(n+1-i)-yt*cos(beta(n+1-i));
end

end
xa(n+1)= c;
ya(n+1) = 0;
yc(n+1) = 0;  % trailing edge


 alpha = a*pi/180;
%This loop calculates the location of the control points

for i = 1:n
    xmid(i) = (xa(i)+xa(i+1))/2;
    ymid(i) = (ya(i)+ya(i+1))/2;
    Sj(i) = sqrt((xa(i+1)-xa(i))^2+(ya(i+1)-ya(i))^2);%array of panel lengths
    phi(i) = atan2((ya(i+1)-ya(i)),(xa(i+1)-xa(i)));
    rhs(i)= sin(phi(i)-alpha);% RHS of equation 4
end

% This loop calculates the coefficients to find the matrix Aij for equation
% 6.Variables are as given in report.
for i = 1:n
    for j = 1:n
        if i==j
            cn1(i,j) = -1;
            cn2(i,j) = 1;
            ct1(i,j) = pi/2;
            ct2(i,j) = pi/2;
        else
        A=-(xmid(i)-xa(j))*cos(phi(j))-(ymid(i)-ya(j))*sin(phi(j));
        B = (xmid(i)-xa(j))^2+(ymid(i)-ya(j))^2;
        C = sin(phi(i)-phi(j));
        D = cos(phi(i)-phi(j));
        E = (xmid(i)-xa(j))*sin(phi(j))-(ymid(i)-ya(j))*cos(phi(j));
        F = log(1+(Sj(j)^2+2*A*Sj(j))/B);
        G = atan2((E*Sj(j)),(B+A*Sj(j)));
        P = (xmid(i)-xa(j))*sin(phi(i)-2*phi(j))+(ymid(i)-ya(j))*cos(phi(i)-
2*phi(j));
        Q = (xmid(i)-xa(j))*cos(phi(i)-2*phi(j))-(ymid(i)-ya(j))*sin(phi(i)-
2*phi(j));
       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

%This loop calculates the coefficients Anij and Atij used in eq. 6 and
%eq. 8 respectively

for i=1:n
        an(i,1)=cn1(i,1);
        an(i,n+1)=cn2(i,n);
        at(i,1)=ct1(i,1);
        at(i,n+1)=ct2(i,n);
        for j=2:n
            an(i,j)=cn1(i,j)+cn2(i,(j-1));
            at(i,j)=ct1(i,j)+ct2(i,(j-1));
        end
end

%kutta condition

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

```
% calculating circulation density by solving linear equations given by
% eq. 6
g = an\rhs';



% calculating tangential velocity and coefficent of pressure
  for i=1:n
    sum=0;
    for j=1:n+1;
        sum=sum+at(i,j)*g(j);
     end
    v(i) = (cos(phi(i)-alpha)+sum);
    cp(i) = 1-v(i)*v(i);
  end
  % compute coefficients of lift and drag

  cy = 0.0;
  cx = 0.0;
  for i = 1:n
      dx = xa(i+1)-xa(i);
      dy = ya(i+1)-ya(i);
      cy = cy-cp(i)*dx;
      cx = cx+cp(i)*dy;
  end

  % print cl and cd on the screen
  cl = cy*cos(alpha)-cx*sin(alpha)
  cd = cy*sin(alpha)+cx*cos(alpha)

plot(xmid(1:n/2)/c,cp(1:n/2),'-*r')
set(gca,'Ydir','reverse')
hold on
plot(xmid(n/2+1:n)/c,cp(n/2+1:n),'-*b')
hold on
%plot(xa,ya,'-k')
 xlabel('x/c')
ylabel('cp')
title('cp vs x/c')
```

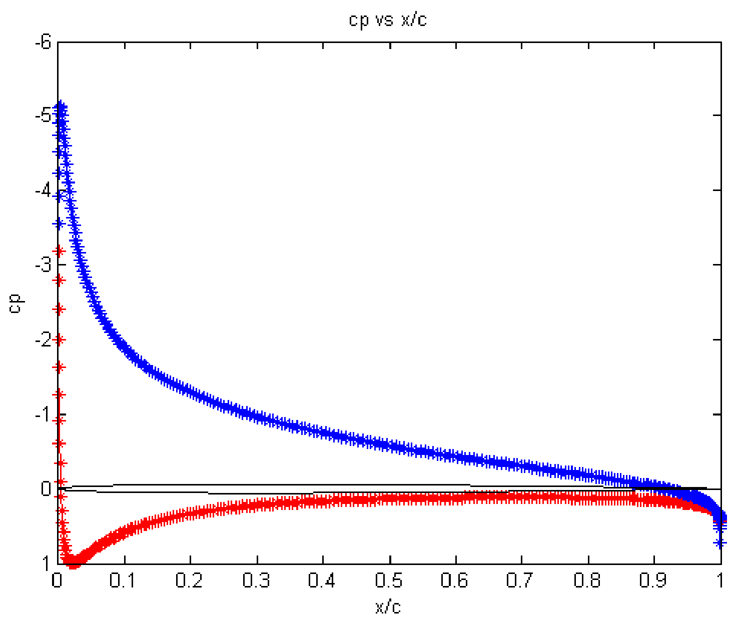

Fig.2 Pressure distribution over the surface of NACA0012 airfoil at 9^0 AOA