NAME - YASH DADLANI

Roll No. - 22101079

COURSE - AERODYNAMICS-II

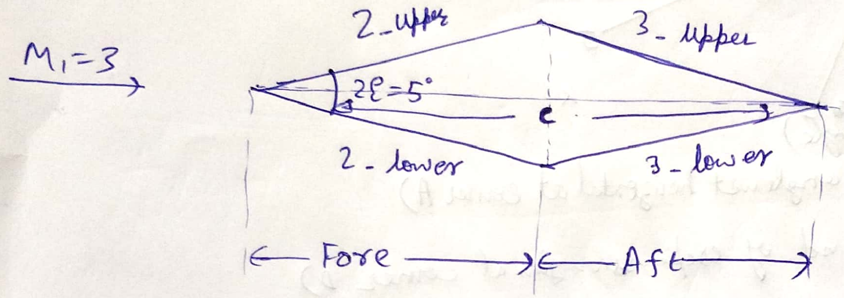
CODE - AE 612A

INSTRUCTOR - PROF. PRADEEP MOISE

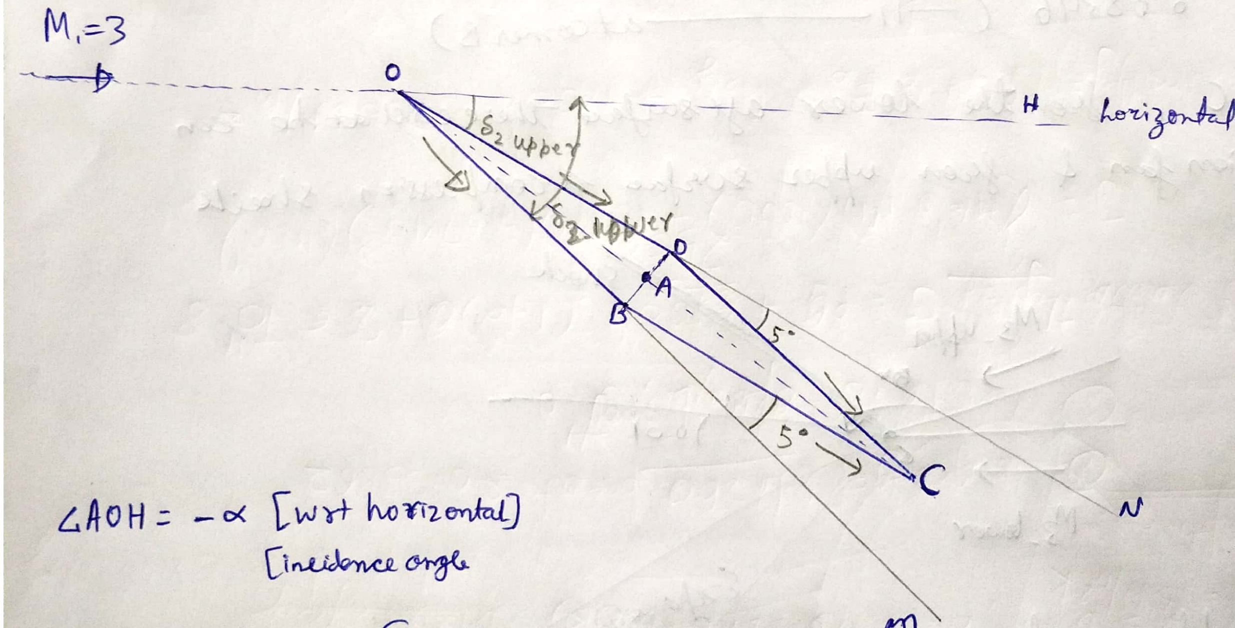
Q1 Diamond Airfoil

$2\varepsilon = 5^\circ$; $M_1 = 3$; $\rho_1 = P_1 = T_1 = 1$ {free stream} ; chord = c = 1

α = incidence angle



(a)



$\angle AOH = -\alpha$ [wrt horizontal]
[incidence angle]

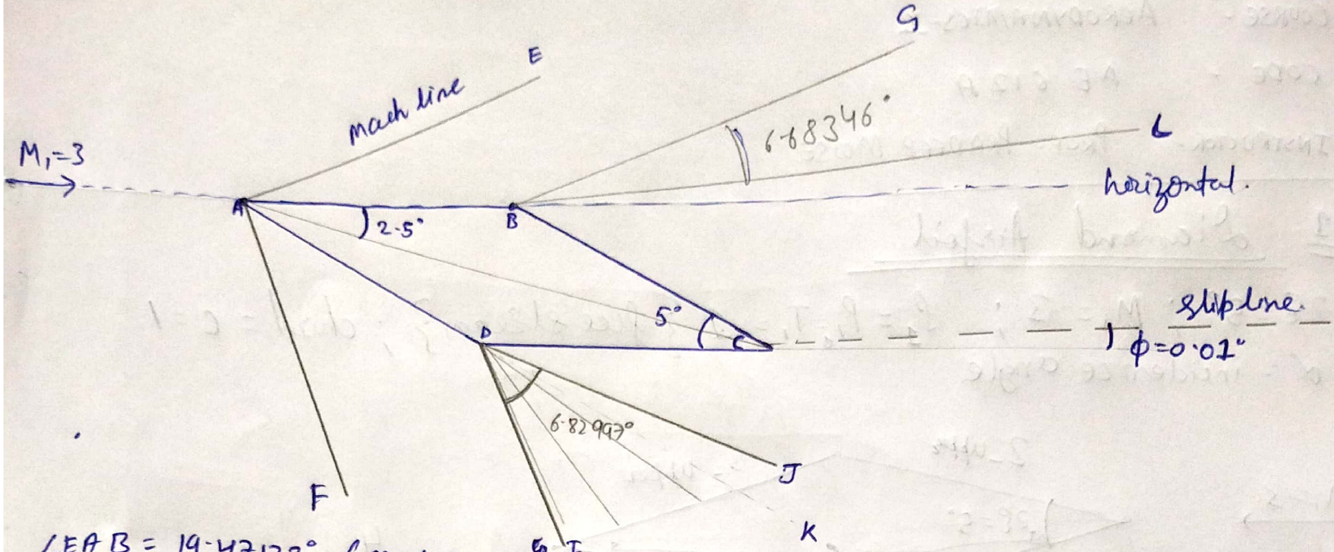
$\angle DOH = -\delta_{2\,upper}$ {Expansion corner} → flow angle$_{2_upper}$

$\angle BOH = +\delta_{2\,lower}$ {Compression Corner} → flow angle$_{2_lower}$

flow angle on aft lower = $\delta_{2_{lower}} - 5^\circ$ (Surface BC)

flow angle on aft upper = $-\delta_{2,upper} - 5^\circ$ (Surface (DC))

(b) * For $\alpha = 2.5°$



$\angle EAB = 19.47122°$ (mach line Angle)

$\angle FAB = 23.1337°$ (Shock wave angle wrt horizontal at corner A)

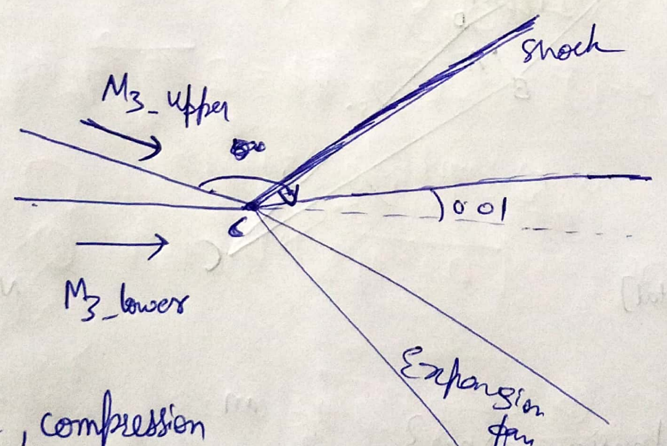
$\angle IDK = 21.32637°$ (lower bound of expansion fan at corner D)

$\angle JDC = 19.4964°$ (upper bound of — " — at corner D)

$\angle IDJ = 6.82997°$ (Expansion fan angle at Corner D)

$\angle GBL = 6.68346$ (— " — at corner B)

At corner C, from the lower aft surface there will be an expansion fan & from upper surface a compression shock



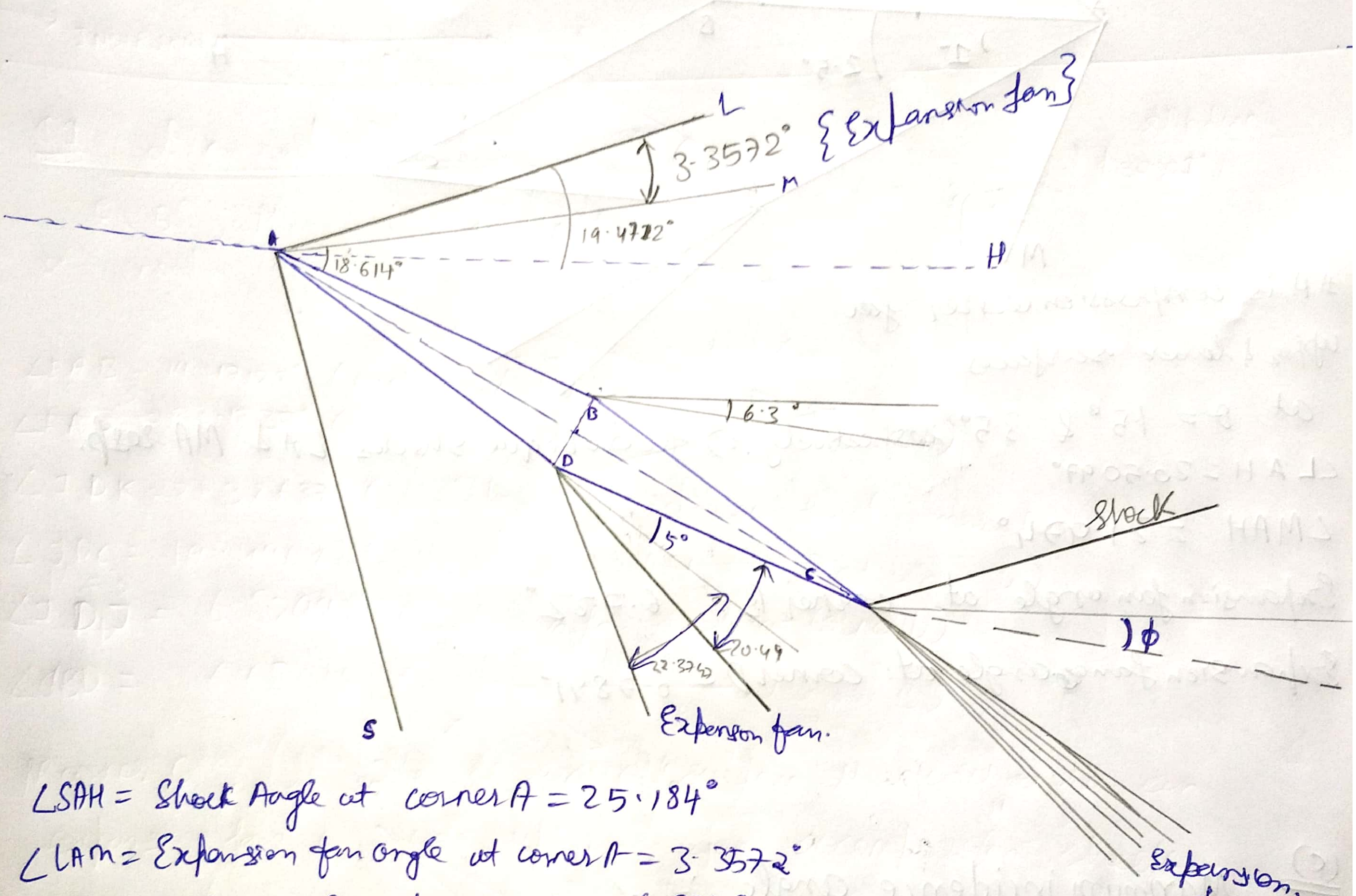
for M_3_upper, compression

Corner angle is $= 5 + 0.01 = 5.01°$

for M_3 -lower, expansion corner angle is $= 0.01°$

The slip line angle calculation is further shown in the next part.

* For $\boxed{\alpha = 5^\circ}$



$\angle SAH$ = Shock Angle at corner A = 25.184°

$\angle LAM$ = Expansion fan angle at corner A = 3.3572°

Angle of Expansion fan at corner D = 6.884°

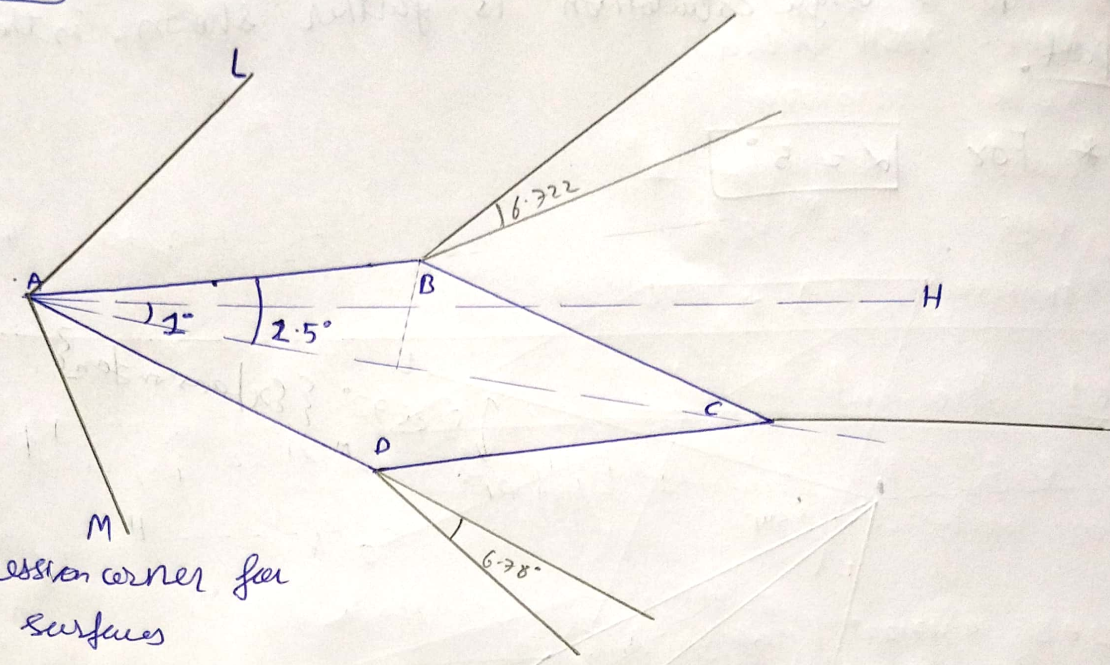
Angle of Expansion fan at corner B = ≃ 6.3°

Slip angle at corner C = −1.35°

We will have a shock from upper surface emanating at corner D & expansion fan from lower surface —ıı——.

∘ A' is an compression corner at $\theta = 7.5^\circ$
& expansion corner at $\theta = 2.5^\circ$

* for $\boxed{\alpha = 1^\circ}$



A is compression corner for upper & lower surfaces at $\theta = 1.5^\circ$ & 3.5° respectively. ⇒ two oblique shocks LA & MA resp.

$\angle LAH = 20.5097^\circ$

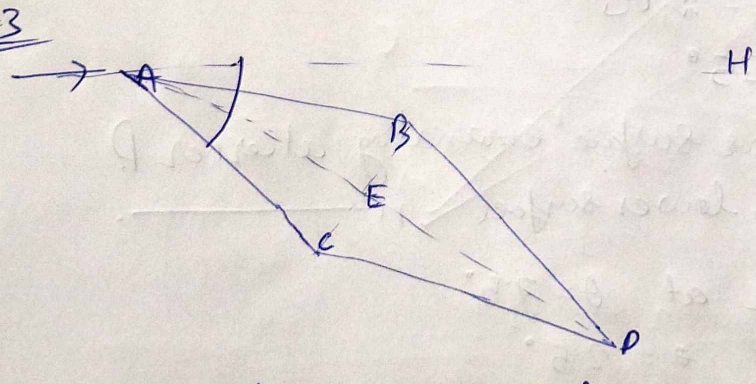
$\angle MAH = 21.974^\circ$

Expansion fan angle at corner B = 6.722°

Expansion fan angle at corner D = 6.7841°

(c) Maximum incidence angle :- α_{max}

$M = 3$



It is computed by maximum deflection angle for $M_1 = 3$ at corner A

$\angle CAH = \theta_{max}$, $\angle EAH = \theta_{max} - \angle CAE = \theta_{max} - 2.5^\circ$

for $M_1 = 3$; Beta_array = linspace$(\sin^{-1}(\frac{1}{3}), \pi/2)$

computing theta_array by

$$\tan\theta[i] = \frac{2\cot\beta[i]\{M_1^2\sin^2\beta[i]-1\}}{M_1^2(\gamma+\cos 2\beta[i])+2}$$

we get correspondingly mapped θ-array.

max value of θ-array = max.(θ_array)

$$\theta_{max} = 34.073439$$

$\therefore$ $\boxed{\text{Max Incidence angle} = 31.5734^\circ}$

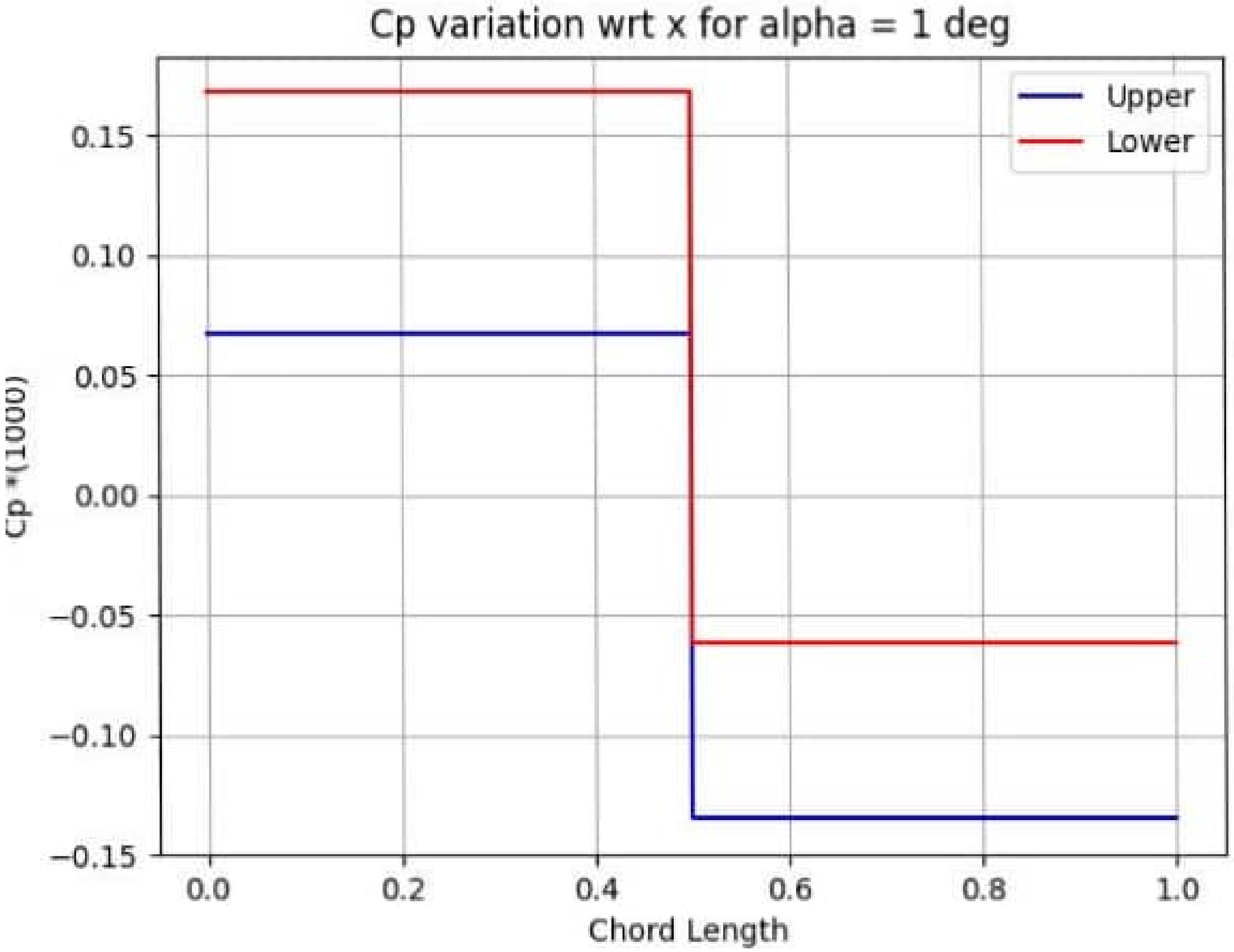
Beyond this incidence angle, the shock wave detaches from corner A.

(d) From the plots of C_p with x, it is evident that upper surface has negative C_p implying lower pressure.

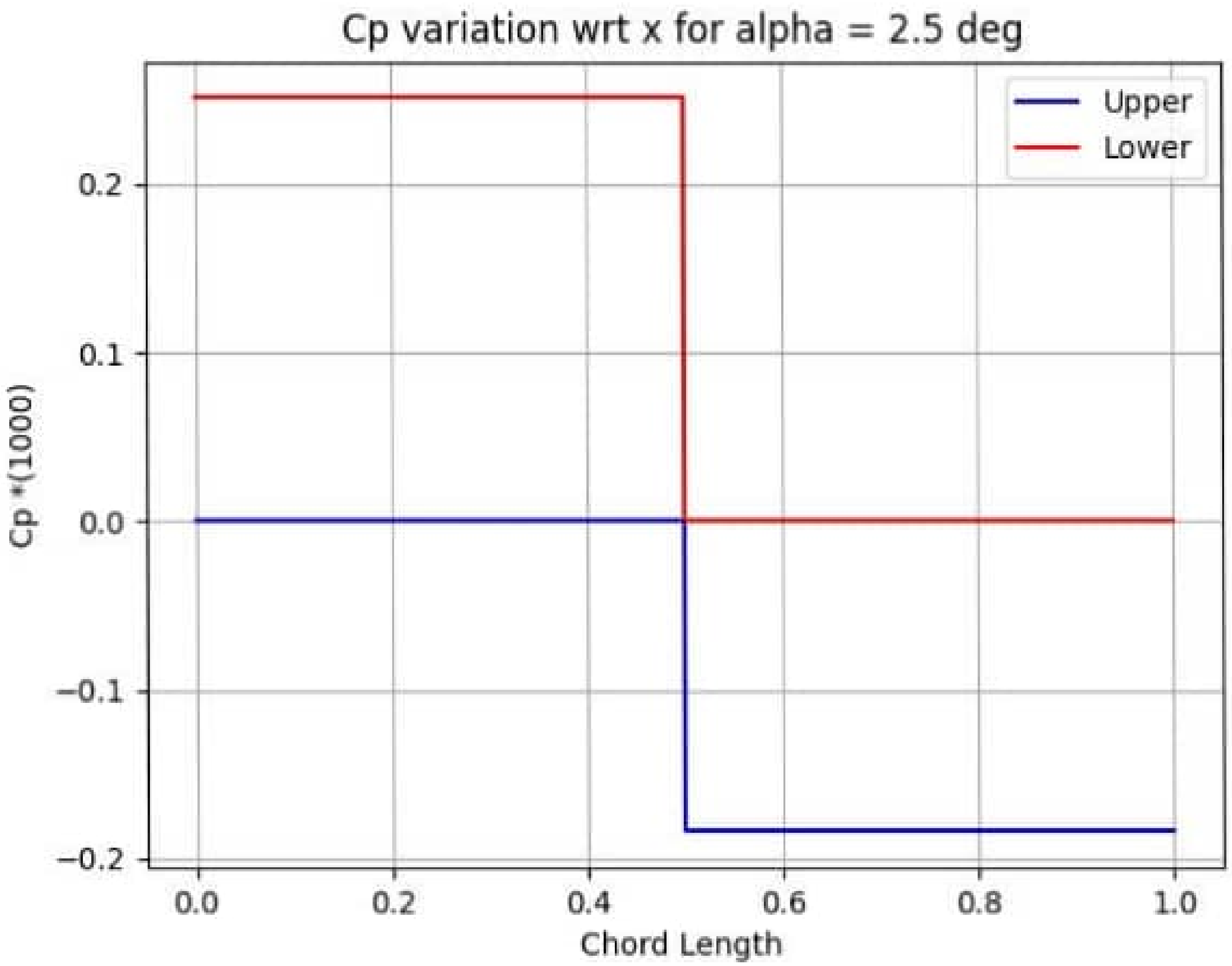
for $\alpha = 1^\circ$, C_p for upper is positive near $x=0$ implying presence of oblique shock on upper surface corner.

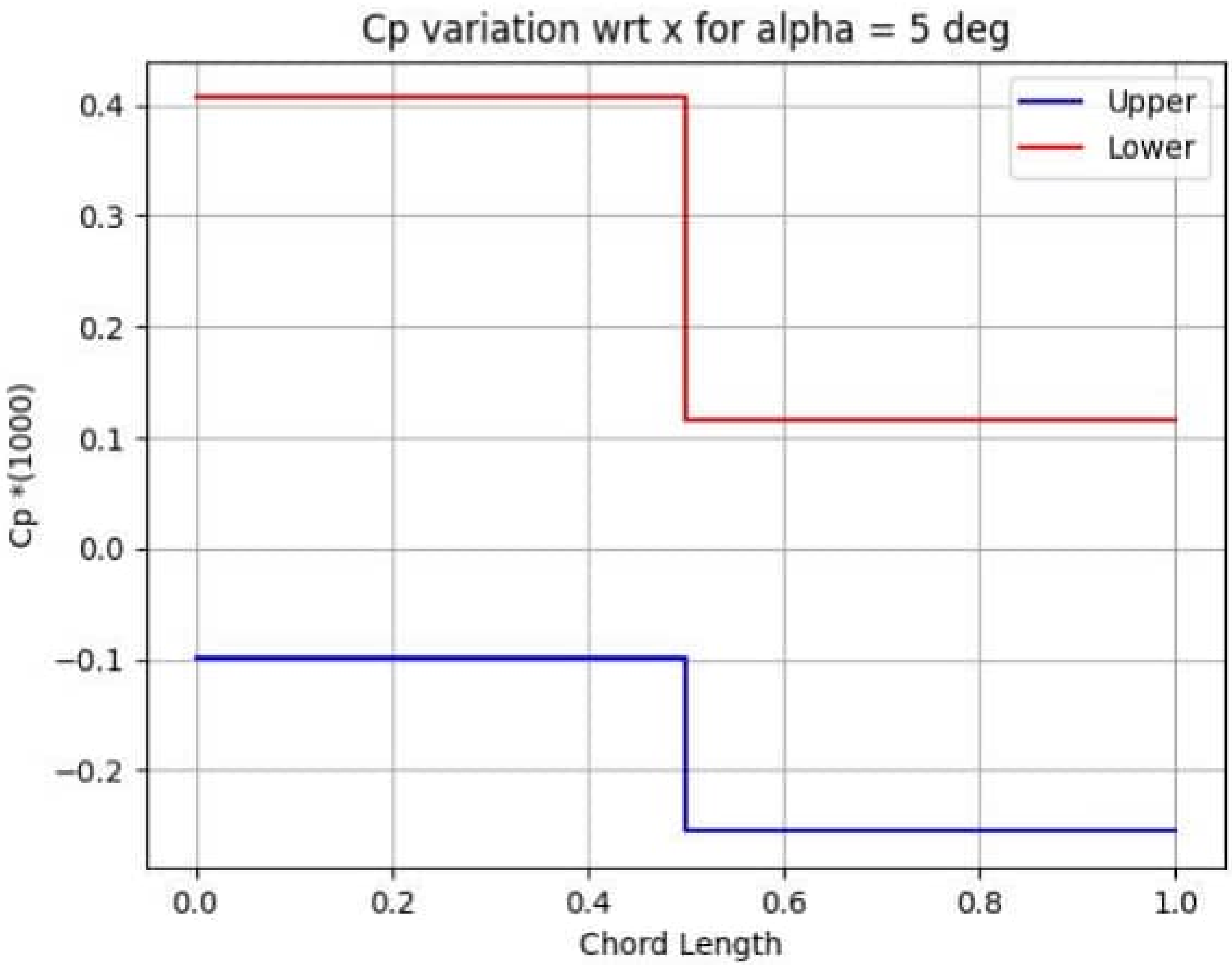
for $\alpha = 2.5$, C_p for upper is zero, because the upper surface is parallel (fore-upper) to free-stream, ie no oblique shock but there is a mach line present.

for $\alpha = 5^\circ$, C_p for fore upper is negative from $x=0$ because there is presence of oblique shock. at corner A.

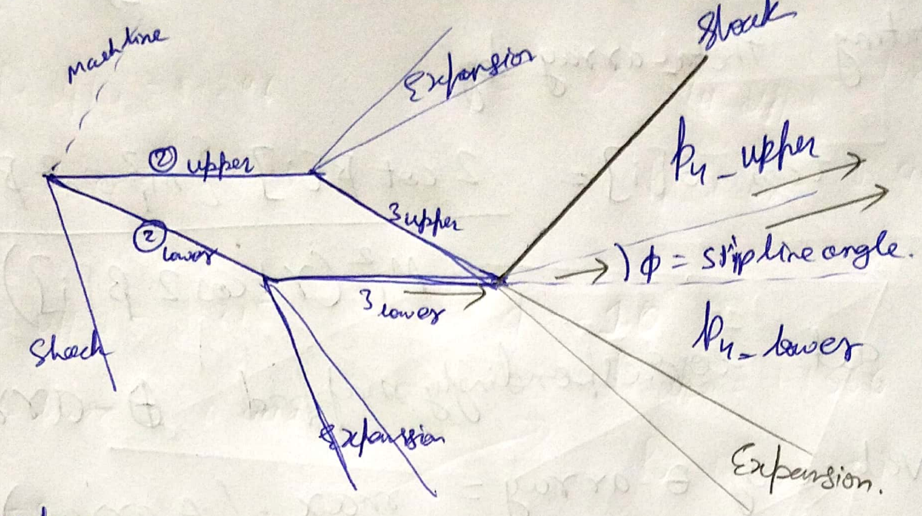


Cp variation wrt x for alpha = 1 deg
Upper
Lower
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
Cp *(1000)
0.0
0.2
0.4
0.6
0.8
1.0
Chord Length


Cp variation wrt x for alpha = 2.5 deg
Upper
Lower
0.2
0.1
0.0
−0.1
−0.2
Cp *(1000)
0.0
0.2
0.4
0.6
0.8
1.0
Chord Length


Cp variation wrt x for alpha = 5 deg
Upper
Lower
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
Cp *(1000)
0.0
0.2
0.4
0.6
0.8
1.0
Chord Length

(e) for $\boxed{\alpha = 2.5°}$



for Aft lower surface

⇒ there is an expansion fan if ϕ is assumed to be greater than 0°

for aft upper surface

→ there is shock for $\phi > 0$.

following are the pressure values

⇒ $P_{2\,upper} = 1$ $\qquad P_{3\,upper} = 0.6673$

$P_{2\,lower} = 1.45405$ $\qquad P_{3\,lower} = 1.00026$

* ϕ is assumed & then P_4_upper & P_4_lower is computed. When they have a difference of 10^{-2}, 10^{-3}, we say the pressure are equal across slip line.

So for $\alpha = 2.5$, following ϕ's were assumed.

$\phi = 2°$ $\quad P_4$_lower $= 1.1634$

P_4_upper $= 0.85433$ $\qquad \phi = 0.01° \Rightarrow P_4$_lower $= 0.9994$

$\phi = 1°$ $\quad P_4$_lower $= 1.07889$ $\qquad P_4$_upper $= 0.9998$.

P_4_upper $= 0.92496$

So, $\phi = 0.01$ is the flow angle in wake for $\alpha = 2.5°$

* $\alpha = 5^\circ$

for $\phi = 1^\circ$, $P_{4\_upper} = 1.296$; for $\phi = 0.05$, $P_{4\_upper} = 1.218275$

$P_{4\_lower} = 0.9255$; $P_{4\_lower} = 0.99685$

for $\phi = 0.01^\circ$, $P_{4\_upper} = 1.215$; for $\phi = -1.0^\circ$

$P_{4\_lower} = 0.9999$; $P_{4\_upper} = 1.1366$

$P_{4\_lower} = 1.0807$

for $\phi = -1.5^\circ$

$P_{4\_upper} = 1.0994$ $\Rightarrow -1.5 < \phi < -1$

$P_{4\_lower} = 1.12259$

$\boxed{\text{We get } \phi = -1.35^\circ} \longrightarrow P_{4\_upper} = 1.1105$

$P_{4\_lower} = 1.1099.$

* for $\alpha = 1^\circ$

for $\phi = 0^\circ$, straightaway, we get $P_{4\,lower} = 0.9997$

$P_{4\,upper} = 0.99914$

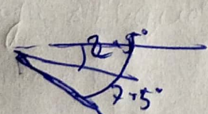
$\Rightarrow \therefore$ for $\alpha = 1^\circ$, flow angle is approximately $= 0^\circ$

Code for case of $\alpha = 5^\circ$

This code estimates the flow variables [Machno, Pressure] & gives an approximate value of slip line angle for $\alpha = 5^\circ$

① Comp. angle $\Rightarrow 7.5^\circ$

② Exp. angle $\Rightarrow 2.5^\circ$ } →



③ Gives values of M2_upper, M3 upper, P2 upper, P3 upper.

Scanned with OKEN Scanner

(4) Now, select an index as the we run $\overset{\text{the}}{\wedge}$ code

→ [8100] because, it gives comp angle of 7.5°

from Beta-array

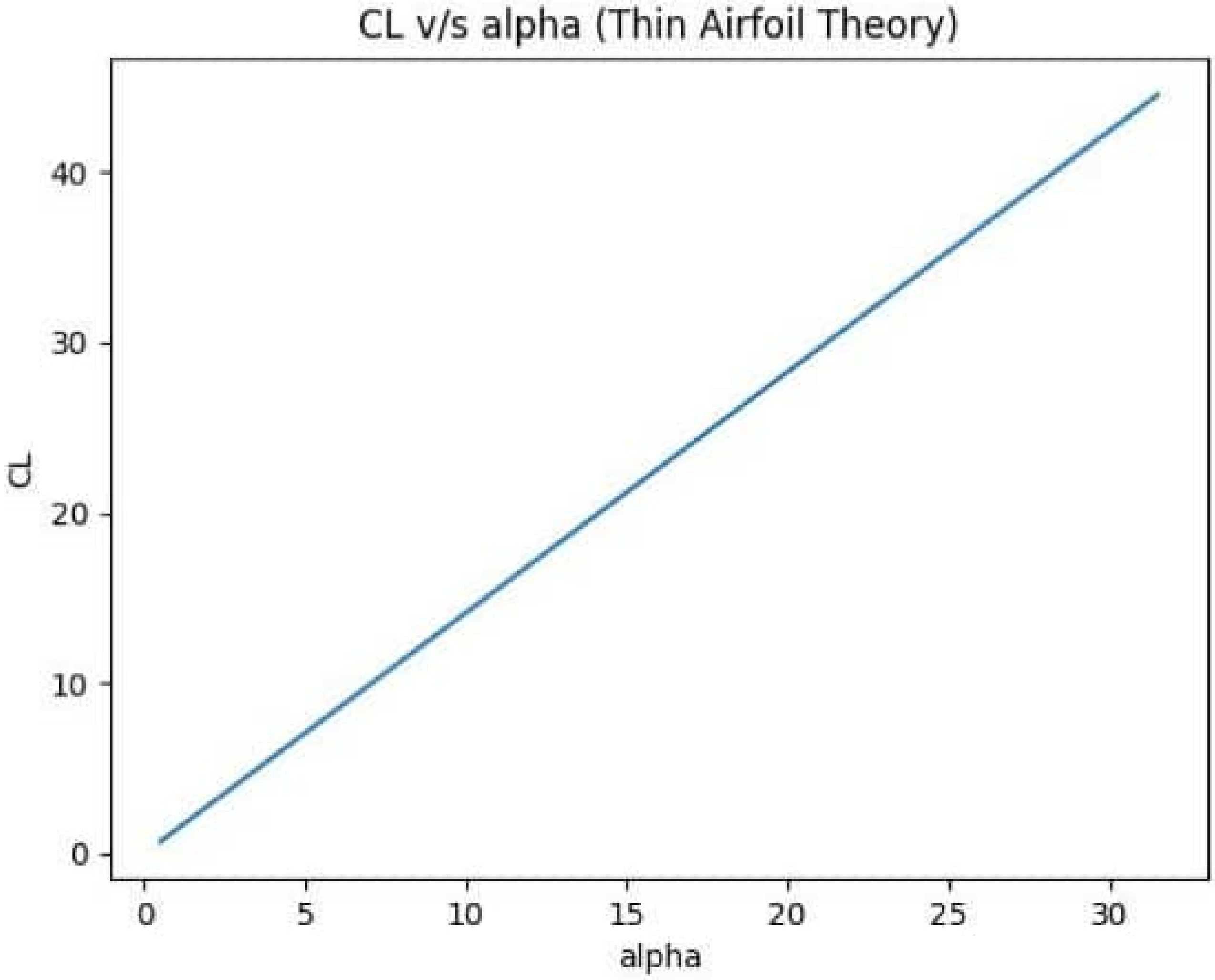
(5) Now Assume a flow angle.

(6) ~~Here~~ check for the values of P_upper(1) & P_lower(4)

(7) If they match upto 3 digits; we have an approximate answer for ~~ϕ=0~~ $\alpha = 5°$

(8) Then, it also plots the $-C_p$ variation wrt chord wise distance

(9) for magnifying the y-label, is multiplied by 1000.


CL v/s alpha (Thin Airfoil Theory)
CL
alpha
0
10
20
30
40
0
5
10
15
20
25
30

This code is for alpha = 2.5 degree, Calculates all the flow variables and displays the Cp plot.

```
import numpy as np

from math import *
import matplotlib.pyplot as plt
k = 1.4 ; a = k+1 ; b = k-1
M_infi = 3; p_infi = 1 ; n = 100001 ;
comp_angle = float(input("Enter a comp angle: "))
def down_mach(M):
    return sqrt((2 + b*(M**2)) / (2*k*(M**2) - b))
def pressure_ratio(M):
    return 1 + ((2*k / (a)) *((M**2) - 1))
def mu(M):
    x = 1/M
    y = np.arcsin(x)
    return y
def beta(M):
    return np.linspace(mu(M),np.pi/2,n )
def isen_stag_PR(M):
    return ((1 + 0.5*b*(M**2))**(k/b))
def prandtl_meyer(M):
    return (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1)))
M = M_infi
beta_array = beta(M)    #np.empty(1001,float) # this is better
theta_array = np.empty(n,float)
for i in range(0,n):
    theta_array[i] = np.arctan((2/tan(beta_array[i]))*((M*sin(beta_array[i]))**2 - 1) / (
(M**2)*(k+ cos(2*beta_array[i])) + 2 ))   # M is M_infi
for i in range(0,n):
    beta_array[i] = beta_array[i] * (180/np.pi)
    theta_array[i] = theta_array[i] * (180/np.pi)
for i in range(0,n):
    if abs(theta_array[i] - comp_angle) <= 0.001:
        print(i, theta_array[i], beta_array[i])

Mn1_lower = M_infi*sin((np.pi/180)*beta_array[5193])
Mn1_upper = 1
Mn2_lower = down_mach(Mn1_lower)
M2_lower = Mn2_lower / (sin((np.pi/180)*(beta_array[5193] - theta_array[5193])))
M2_upper = 3
M3_upper = 3.27341
M3_lower = 2.99627
# print(M2_lower)
p2_upper = p_infi*pressure_ratio(Mn1_upper)
p2_lower = p_infi*pressure_ratio(Mn1_lower)
p3_upper = p2_upper*((isen_stag_PR(M2_upper))/isen_stag_PR(M3_upper))
p3_lower = p2_lower*((isen_stag_PR(M2_lower))/isen_stag_PR(M3_lower))


p_upper = [p2_upper, p2_upper,p3_upper,p3_upper]
cp_upper = np.empty(4,float)
```

```
x = [0,1/2,1/2, 1]
p_lower = [p2_lower, p2_lower,p3_lower,p3_lower]
cp_lower = np.empty(4,float)
for i in range(0,4):
    cp_upper[i] = (2/(9*1.4*287))*(p_upper[i] - 1)*1000
for i in range(0,4):
    cp_lower[i] = (2/(9*1.4*287))*(p_lower[i] - 1)*1000
A, = plt.plot(x,cp_upper, color = 'blue', label = 'Upper')
B, = plt.plot(x,cp_lower, color = 'red', label = 'Lower')
plt.xlabel("Chord Length ")
plt.ylabel("Cp *(1000)")
plt.legend(['Upper', "Lower"])
plt.title("Cp variation wrt x for alpha = 2.5 deg")
plt.grid()
plt.show()
```

Below is the code for slip angle estimation. The Flow angle is chosen as 0.01 degree and the pressure in the wake upper and lower regions are displayed as results.

```
import numpy as np
from math import *
from scipy.optimize import newton
k = 1.4 ; a = k+1 ; b = k-1 ; n = 100001
# M3_upper = float(input("Enter M3_upper : "))
# M3_lower = float(input("Enter M3_lower : "))
M3_upper = 3.27341
M3_lower = 2.99627
phi = 0.01

comp_angle = 5 + phi
expan_angle = 0 + phi
def down_mach(M):
    return sqrt((2 + b*(M**2)) / (2*k*(M**2) - b))
def pressure_ratio(M):
    return 1 + ((2*k / (a)) *((M**2) - 1))
def mu(M):
    x = 1/M
    y = np.arcsin(x)
    return y
def beta(M):
    return np.linspace(mu(M),np.pi/2,n )
def isen_stag_PR(M):
    return ((1 + 0.5*b*(M**2))**(k/b))
def prandtl_meyer(M):
    return (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1)))

M = M3_upper
beta_array = beta(M)    #np.empty(1001,float) # this is better
theta_array = np.empty(n,float)
```

```
for i in range(0,n):
    theta_array[i] = np.arctan((2/tan(beta_array[i]))*((M*sin(beta_array[i]))**2 - 1) / (
(M**2)*(k+ cos(2*beta_array[i])) + 2 ))   # M is M_infi
for i in range(0,n):
    beta_array[i] = beta_array[i] * (180/np.pi)
    theta_array[i] = theta_array[i] * (180/np.pi)
for i in range(0,n):
    if abs(theta_array[i] - comp_angle) <= 0.001:
        print(i, theta_array[i], beta_array[i])
d = int(input("Enter a beta array index of your choice: "))

Mn3_upper = M3_upper*sin((np.pi/180)*beta_array[d])
p3_upper = 0.6673
p3_lower  = 1.00026
# Mn4_upper = down_mach(Mn3_upper)
nu_M4_lower = phi + prandtl_meyer(M3_lower)
f = lambda M: (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1))) - nu_M4_lower
M4_lower = newton(f,3)
p4_upper = p3_upper*pressure_ratio(Mn3_upper)
p4_lower = p3_lower*((isen_stag_PR(M3_lower))/isen_stag_PR(M4_lower))



print( "Upper pressure : " + str(p4_upper))
print("Lower Pressure : " + str(p4_lower))
# print("value  of nu_M4_lower is : " + str(nu_M4_lower))
```

Below is the complete code for alpha = 5 degree. This code gives all the results such as flow variables, slip angle, and the Cp Plots.

```
import numpy as np
from math import *
from scipy.optimize import newton
import matplotlib.pyplot as plt
k = 1.4 ; a = k+1 ; b = k-1
M_infi = 3; p_infi = 1 ; n = 100001 ; R = 287
comp_angle = float(input("Enter a comp angle: "))
expan_angle = float(input("Enter a expan angle: "))
def down_mach(M):
    return sqrt((2 + b*(M**2)) / (2*k*(M**2) - b))
def pressure_ratio(M):
    return 1 + ((2*k / (a)) *((M**2) - 1))
def mu(M):
    x = 1/M
    y = np.arcsin(x)
    return y
def beta(M):
    return np.linspace(mu(M),np.pi/2,n )
def isen_stag_PR(M):
    return ((1 + 0.5*b*(M**2))**(k/b))
```

```
def prandtl_meyer(M):
    return (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1)))
nu_M2_upper = prandtl_meyer(M_infi) + expan_angle

f = lambda M: (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1))) - nu_M2_upper
M2_upper = newton(f,3)
print("M2_upper is : " + str(M2_upper))
p2_upper = p_infi*((isen_stag_PR(M_infi))/isen_stag_PR(M2_upper))
print("p2_upper is :" + str(p2_upper))
nu_M3_upper = nu_M2_upper + 5
f = lambda M: (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1))) - nu_M3_upper
M3_upper = newton(f,3.2)
print("M3_upper is : " + str(M3_upper))
p3_upper = p2_upper*((isen_stag_PR(M2_upper))/isen_stag_PR(M3_upper))
print("p3_upper is :" + str(p3_upper))
M = M_infi
beta_array = beta(M)                           #np.empty(1001,float) # this is better
theta_array = np.empty(n,float)
for i in range(0,n):
    theta_array[i] = np.arctan((2/tan(beta_array[i]))*((M*sin(beta_array[i]))**2 - 1) / (
(M**2)*(k+ cos(2*beta_array[i])) + 2 ))    # M is M_infi
for i in range(0,n):
    beta_array[i] = beta_array[i] * (180/np.pi)
    theta_array[i] = theta_array[i] * (180/np.pi)
for i in range(0,n):
    if abs(theta_array[i] - comp_angle) <= 0.001:
        print(i, theta_array[i], beta_array[i])
d = int(input("Enter a beta array index of your choice: "))
Mn1_lower = M_infi*sin((np.pi/180)*beta_array[d])
Mn2_lower = down_mach(Mn1_lower)
M2_lower = Mn2_lower / (sin((np.pi/180)*(beta_array[d] - theta_array[d])))
print("M2_lower is : " + str(M2_lower))
p2_lower = p_infi*pressure_ratio(Mn1_lower)
print("p2_lower is :" + str(p2_lower))
nu_M3_lower = prandtl_meyer(M2_lower) + 5
f = lambda M: (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1))) - nu_M3_lower
M3_lower = newton(f,1.5)
print("M3_lower is : " + str(M3_lower))
p3_lower = p2_lower*((isen_stag_PR(M2_lower))/isen_stag_PR(M3_lower))
print("p3_lower is :" + str(p3_lower))

phi = float(input("Assume flow angle: "))
comp_angle_trailing_edge = phi + comp_angle
expan_angle_trailing_edge = expan_angle + phi
M = M3_upper
beta_array1 = beta(M)                           #np.empty(1001,float) # this is better
theta_array1 = np.empty(n,float)
for i in range(0,n):
    theta_array1[i] = np.arctan((2/tan(beta_array1[i]))*((M*sin(beta_array1[i]))**2 - 1) /
( (M**2)*(k+ cos(2*beta_array1[i])) + 2 ))    # M is M3_upper
```

```
for i in range(0,n):
    beta_array1[i] = beta_array1[i] * (180/np.pi)
    theta_array1[i] = theta_array1[i] * (180/np.pi)
for i in range(0,n):
    if abs(theta_array1[i] - comp_angle_trailing_edge) <= 0.001:
        print(i, theta_array1[i], beta_array1[i])
d1 = int(input("Enter a beta array index of your choice: "))
Mn3_upper = M3_upper*sin((np.pi/180)*beta_array[d1])
p4_upper = p3_upper*pressure_ratio(Mn3_upper)
print("p4_upper is :" + str(p4_upper))
nu_M4_lower = prandtl_meyer(M3_lower) + expan_angle_trailing_edge
f = lambda M: (180/np.pi)* (sqrt(a/b) *np.arctan(sqrt((b/a)*((M**2)-1)))  -
np.arctan(sqrt((M**2)-1))) - nu_M4_lower
M4_lower = newton(f,1.2)
print("M4_lower is : " + str(M4_lower))
p4_lower = p3_lower * ((isen_stag_PR(M3_lower))/isen_stag_PR(M4_lower))
print("p4_lower is : " + str(p4_lower))

p_upper = [p2_upper, p2_upper,p3_upper,p3_upper]
cp_upper = np.empty(4,float)
x = [0,1/2,1/2, 1]
p_lower = [p2_lower, p2_lower,p3_lower,p3_lower]
cp_lower = np.empty(4,float)
for i in range(0,4):
    cp_upper[i] = (2/(9*1.4*287))*(p_upper[i] - 1)*1000
for i in range(0,4):
    cp_lower[i] = (2/(9*1.4*287))*(p_lower[i] - 1)*1000
print(cp_upper)
print(cp_lower)
A, = plt.plot(x,cp_upper, color = 'blue', label = 'Upper')
B, = plt.plot(x,cp_lower, color = 'red', label = 'Lower')
plt.xlabel("Chord Length ")
plt.ylabel("Cp *(1000)")
plt.legend(['Upper', "Lower"])
plt.title("Cp variation wrt x for alpha = 5 deg")
plt.grid()
plt.show()
```