TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

An assignment report on evaluation of pressure distribution over a 2D Tangent Ogive Profile using Shock Expansion Method

Submitted by:

Sandip Adhikari

076bas035

Submitted to:

Asst. Professor Dr. Sudip Bhattarai

Department of Aerospace and Mechanical Engineering,

IOE, Pulchowk Campus

February 18, 2024

Introduction

Among the various local surface inclination methods used to determine the surface pressure distribution over hypersonic bodies, the shock expansion method utilizes the Prandtl-Meyer expansion function to find the pressure at each point on the surface. The shock expansion method assumes a sharp-nosed body with an attached shockwave.

The steps to calculate the pressure coefficient (C_p) over a body (as shown in figure 1) travelling at hypersonic speeds are described below:

a. Assuming the nose of the body to be wedge with semi-angle θ_n and based on that deflection angle, calculating M_n & P_n behind the oblique shockwave at nose using oblique shock relations.

b. Calculating pressure P_i at a point i by assuming Prandtl-Meyer Expansion from n to i. To calculate the pressure P_i , M_i should be calculated using equation (i) which makes the use of Prandtl-Meyer Function. And the expansion angle $\Delta\theta$ should also be evaluated by determination of the local inclination angle θ_i.

Expansion angle,

$$\Delta\theta = \theta_n - \theta_i$$

$$or,\ \theta_n - \theta_i = \nu(M_i) - \nu(M_n) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (i)$$

Here,

$$\nu(M) = \sqrt{\frac{\gamma+1}{\gamma-1}} \tan^{-1}\sqrt{\frac{\gamma-1}{\gamma+1}(M^2-1)} - \tan^{-1}\sqrt{M^2-1}$$
$$= Prandtl - Meyer\ Function$$

Then, pressure P_i is calculated using equation (ii).

$$\frac{P_i}{P_n} = \left(\frac{1+\left[\frac{\gamma-1}{2}\right]M_n^2}{1+\left[\frac{\gamma-1}{2}\right]M_i^2}\right)^{\frac{\gamma}{\gamma-1}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (ii)$$

c. Finally, calculating C_p for the corresponding P_i value using equation (iii).

$$C_p = \frac{P_i - P_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (iii)$$

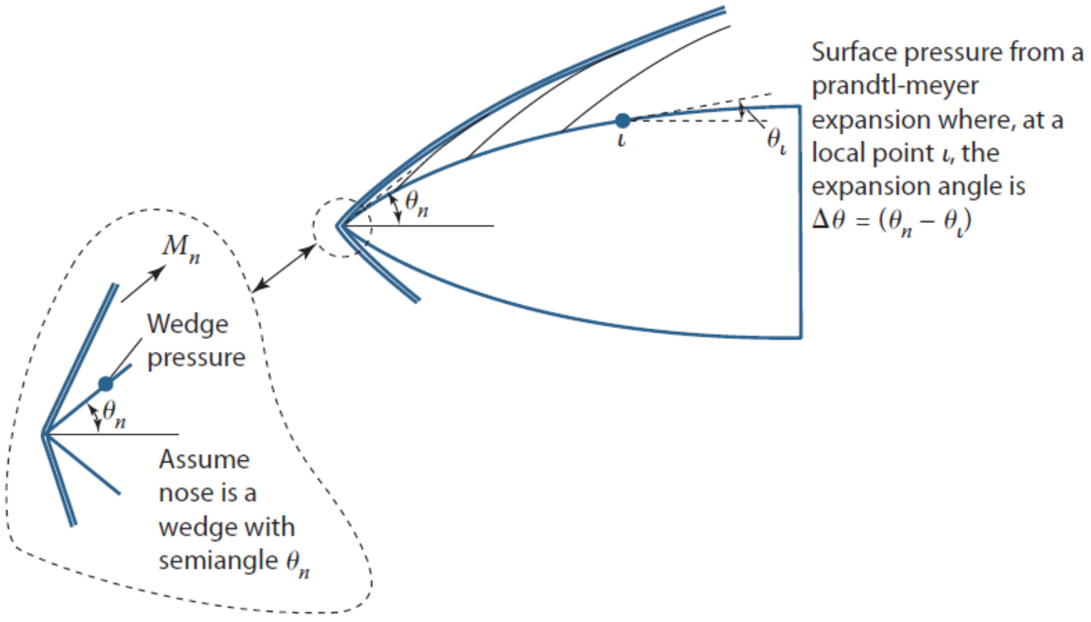
Surface pressure from a prandtl-meyer expansion where, at a local point ι, the expansion angle is $\Delta\theta = (\theta_n - \theta_\iota)$
ι
θ_ι
θ_n
M_n
Wedge pressure
θ_n
Assume nose is a wedge with semiangle θ_n

Figure 1: Shock Expansion Method

Geometry

A tangent ogive (a type of nose cone) 2D profile having construction circle radius ρ (also called Ogive radius) of 1m and length L of 1m was selected. The corresponding base radius R of the geometry is 0.381966m.

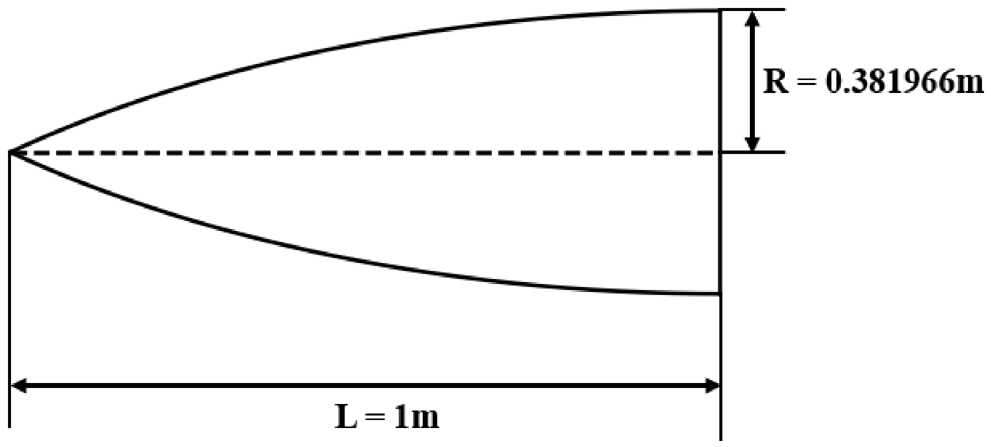

Figure 2: Tangent Ogive Profile Geometry

The profile of the geometry shown in figure 2 was plotted by writing a script in python (mentioned in appendix), with the implementation of following tangent ogive functions:

Ogive radius, $\rho^2 = \frac{R^2+L^2}{2R}$

The radius y at any point x, as x varies from 0 to L is:

$$y = \sqrt{\rho^2 - (L-x)^2} + R - \rho$$

Freestream Flow Conditions

The selected freestream Mach number was 6.2 and other atmospheric conditions were selected based on the altitude of the 30km.

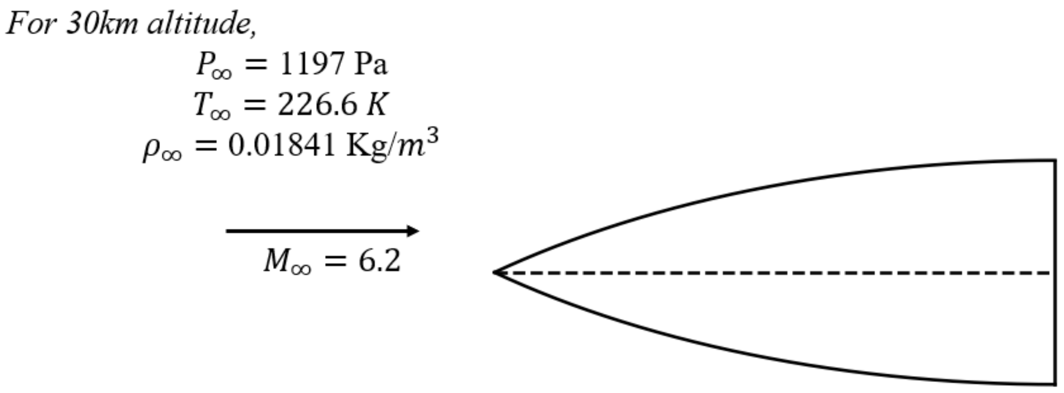

Figure 3: Freestream Flow Conditions

Plot

As shown in the obtained C_p vs x/L plot (figure 4), the pressure coefficient C_p is decreasing exponentially as we move downstream along the profile and the maximum C_p is obtained at the nose, which is equal to 1.24731.

At the nose, as the flow is hypersonic, an oblique shock is formed. From the code, the obtained deflection angle at nose is $\theta_n = 41.8103^\circ$ and the corresponding wave angle is $\beta = 61.64253^\circ$. Due to the formation of shock wave, the freestream flow properties changed behind the shock wave. The Mach number decreased to 5.45600 whereas the pressure increased to 41371.55476 Pa.

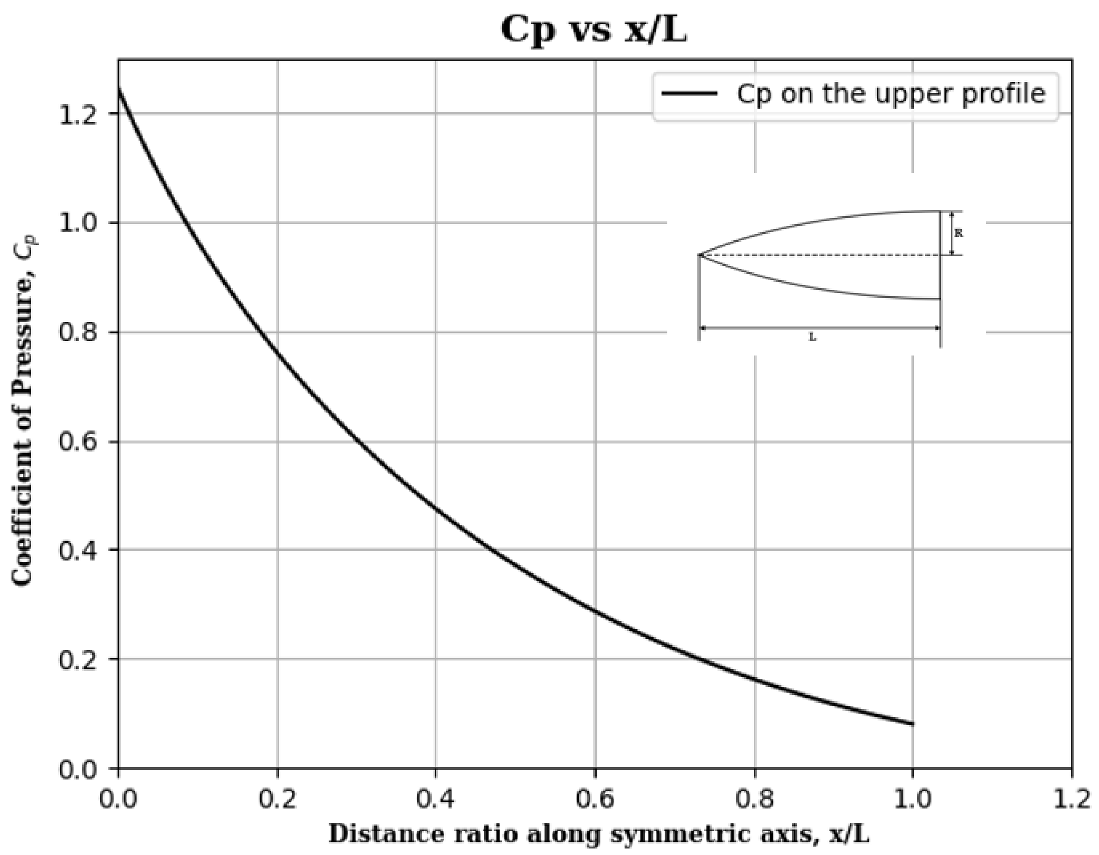

Figure 4: Cp vs x/L plot over the upper profile of tangent Ogive

After the change in properties of the flow due to the encounter of shockwave, the flow is turned away from itself because of the changing tangent ogive profile. This results in the formation of expansion waves and hence, the pressure over the profile decreased on moving from nose to the end of the profile (upto x/L =1). On the other hand, the Mach number had an increasing nature.

Conclusion

A python code to evaluate the pressure distribution over a tangent ogive profile, was written and compiled in VS Studio Code. The coefficient of pressure C_p was plotted against the symmetric axis distance ratio x/L, which showed an exponentially decreasing nature of C_p with increasing x/L (upto x/L =1).

APPENDIX I

Python Source Codes:

a. Tangent Ogive Profile Construction

```
import numpy as np
import matplotlib.pyplot as plt

rho = 1.5 # radius of the circle (in m)
L = 1 # length of the ogive profile (in m)

#base radius of the tangent ogive profile
R = rho - np.sqrt(rho**2 - L**2)
print(R)

#profile data calculation
x = np.linspace(0, L, 1000)
y = np.sqrt(rho**2 - (L-x)**2) + R-rho
print(len(x))


plt.plot(x, y, 'k', label = 'Tangent Ogive Profile', linewidth = 1.5)
plt.plot(x, -y, 'k',linewidth = 1.5)
plt.plot([x[999], x[999]], [y[999], -y[999]], 'k', linewidth = 1.5)
plt.plot([x[0], x[999]], [y[0], y[0]], 'k--', linewidth = 1.5)
plt.xlim([-0.2, 1.2])
plt.ylim([-1, 1])
#plt.grid(True)
plt.legend()
plt.show()
```

b. Cp vs x/L plot using Shock Expansion Method

```
# Now I am become Death, the destroyer of worlds.#

import numpy as np
import matplotlib.pyplot as plt
import scipy.optimize as optimize

#Tangent Ogive Profile Parameters
rho = 1.5 # radius of the circle (in m)
L = 1 # length of the ogive profile (in m)
R = rho - np.sqrt(rho**2 - L**2)#base radius of the tangent ogive profile
print(R)
```

```
#profile independent data calculation
x = np.linspace(0, L, 1000)
y = np.sqrt(rho**2 - (L-x)**2) + R-rho

#Freestream Flow Parameters
M1 = 6.2 #Mach number
g = 1.4 #specific heat ratio
#atmospheric properties corresponding to alt = 30km (from sea level)
T1 = 226.6 #Temperature in K (in celcius -46.64 )
P1 = 1197 #Pressure in Pa
rhoa1 = 0.01841 #density in kg/m^3
a = np.sqrt((g*P1)/rhoa1) #speed of sound
u1 = M1 * a #flow velocity

#Function to find the inclination angle(theta/deflection angle) at any point
on the profile
def theta_calculation(x):
    theta = np.arctan((L - x) / np.sqrt(rho**2 - (L - x)**2))
    return theta

#Function to find beta for a given theta
def beta_calculation(theta_val):
    theta_function = lambda beta_val, theta_val :
np.arctan(2*(M1**2*np.sin(beta_val)**2-
1)/(np.tan(beta_val)*(M1**2*(g+np.cos(2*beta_val))+2))) - theta_val
    beta_initial_guess = 0.3
    beta_solution = optimize.fsolve(theta_function, beta_initial_guess,
args=(theta_val), xtol = 1e-6)
    return beta_solution[0]

#Function to find the expansion angle (Prandtl_Meyer_Function)
def expansion_angle(del_theta):
    v = np.sqrt((g+1)/(g-1)) * np.arctan(np.sqrt((g-1)/(g+1) * (M[0]**2 - 1)))
- np.arctan(np.sqrt(M[0]**2 - 1))
    del_theta_function = lambda Mi, del_theta :  np.sqrt((g+1)/(g-
1))*np.arctan(np.sqrt((g-1)/(g+1) * (Mi**2 - 1))) - np.arctan(np.sqrt(Mi**2 -
1)) - v - del_theta
    Mi_initial_guess = 1.5
    Mi_solution = optimize.fsolve(del_theta_function, Mi_initial_guess, args =
(del_theta), xtol = 1e-6)
    return Mi_solution[0]

#Flow properties due to the oblique shockwave at nose and the shock parameters
theta_n = theta_calculation(x[0])  #deflection at nose
#print(theta_n, np.rad2deg(theta_n))
beta_n = beta_calculation(theta_n) # shockwave angle
#print(beta_n, np.rad2deg(beta_n))
```

```
Mn1 = M1 * np.sin(beta_n) # Mach normal component before shock
Mn2 = np.sqrt((1+ (g-1)/2 * Mn1**2)/(g*Mn1**2-(g-1)/2))# Mach normal component
behind the shock
M2_n = Mn2/np.sin(beta_n - theta_n)# Flow Mach Behind the shock
P2_n = (1 + (2*g)/(g+1)*(M1**2*np.sin(beta_n)**2-1)) * P1 # Pressure behind
the shock
Cp_n = (P2_n - P1)/ (0.5 * rhoa1 * u1**2) #coeff of pressure at nose
#print(Mn1, Mn2, Cp_n, M2_n, P2_n)


#Cp calculation along the profile
P = np.empty(len(x))
M = np.empty(len(x))
Cp = np.empty(len(x))
M[0] = M2_n # Mn
P[0] = P2_n # Pn
Cp[0] = Cp_n
for i in range(1, len(x)-1):
    del_theta1 = theta_n - theta_calculation(x[i])
    M[i] = expansion_angle(del_theta1)
    P[i] = (((1 + ((g-1)/2) * M[0]**2)/(1 + ((g-1)/2) * M[i]**2))**(g/(g-1)))
* P[0]
    Cp[i] = (P[i] - P1)/ (0.5 * rhoa1 * u1**2)

Cp[len(x)-1] = 2*Cp[len(x)-2] - Cp[len(x)-3]
#print(M, Cp)

plt.plot(x/L, Cp, 'k', label = 'Cp on the upper profile', linewidth = 1.5 )

# Set font properties for labels
font_propl = {
    'family': 'serif',
    'size': 9,
    'weight': 'bold'
}
# Set font properties for title
font_propt = {
    'family': 'serif',
    'size': 14,
    'weight': 'bold'
}

# Set plot title and labels
plt.title('Cp vs x/L', fontdict = font_propt)
plt.xlabel(' Distance ratio along symmetric axis, x/L', fontdict = font_propl)
plt.ylabel(' Coefficient of Pressure, $C_{p}$', fontdict = font_propl)
plt.xlim([0, 1.2])
plt.ylim([0, 1.3])
plt.grid(True)
plt.legend()
plt.show()
```