Shape, arrow

Description automatically generated

**TRIBHUVAN UNIVERSITY**

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

**An assignment report**

**on determination of condition for**

**formation of Mach stem using shock polar plot**

S**ubmitted by:**

Sandip Adhikari

076bas035

**Submitted to:**

Asst. Professor Sudip Bhattarai

Department of Aerospace and Mechanical Engineering,

IOE, Pulchowk Campus

February 7, 2024

**Introduction**

Shock polar, also known as pressure-deflection shock polar, is the graphical representation of the pressure ratio (between flow downstream and upstream of an oblique shockwave) and the flow deflection angle, , for a given upstream Mach number. The plot represents the locus of all the possible flow states after an oblique shockwave formation. Shock polars are useful in understanding shock reflection phenomena as they enable the analysis of reflection patterns and prediction of transitions (from regular to irregular reflection).

Relations used to draw the plot are mentioned below:

1. **Possible range of shockwave angle,**

The minimum possible value of corresponds to the value for Mach wave, which is equal to . Here, M1 is the upstream flow Mach number. And the maximum possible value corresponds to the value for normal shock wave;

Hence, the range of is given by

1. **relation for oblique shockwave**

Where,

= deflection angle; represents the deflection of the flow across the shockwave

= ratio of specific heats

1. **Pressure Ratio**

Here, P1 and P2 are the pressures upstream and downstream the oblique shockwave.

**Upstream Flow and Wedge Conditions**

Both the wedge angles and are taken to be (as per sign convention, is considered negative) whereas the upstream flow Mach number, is equal to 5.5. Figure 1 shows the representation of the conditions set.

The condition for the formation of Mach stem is determined from the shock polar plot, through trail and error, by keeping constant and varying .

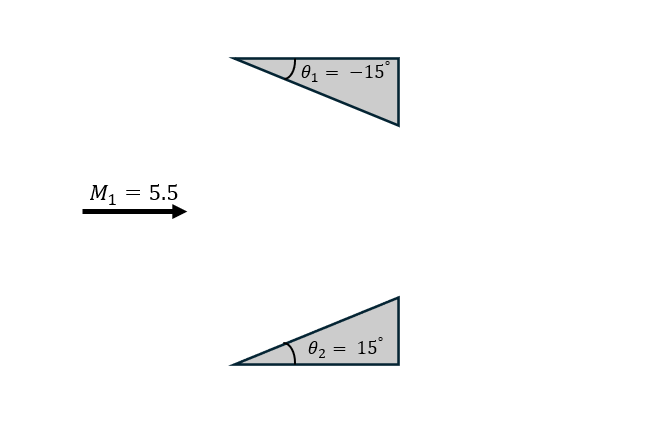


Figure 1: Representation of the upstream flow and initial wedge conditions

**Plot**

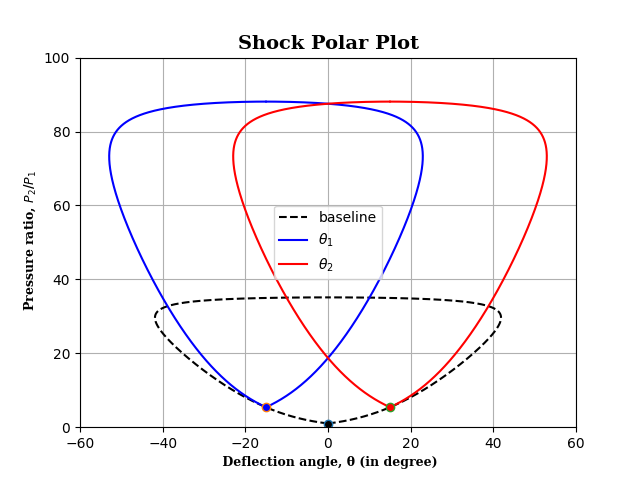
Figure 2 shows the baseline plot, which represents the locus of all the possible flow states after an oblique shockwave formation for . Along with the baseline plot, it shows the possible states for starting wedge angles and .

Figure 2: Shock Polar Plot for and

On keeping the constant and varying the , the two curves just separated (no intersection between curves) at (correct to one decimal place), which is shown in Figure 4, signifying the formation of Mach stem. Because the curves do not intersect, there is no common reflection point for the oblique shockwaves generated at the two wedges separately. So, in that case, Mach reflection occurs, resulting in Mach stem.

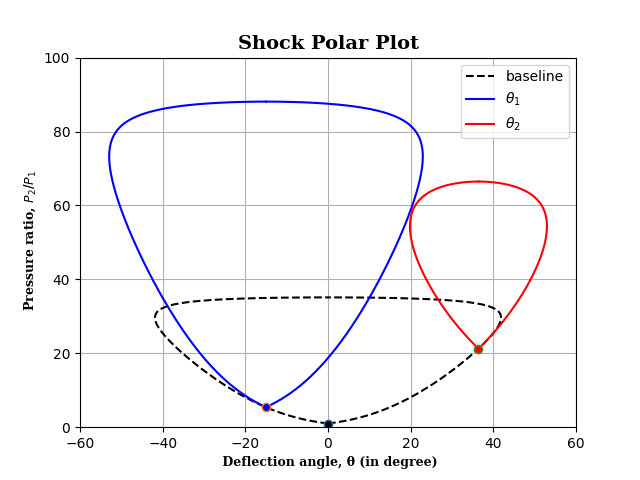
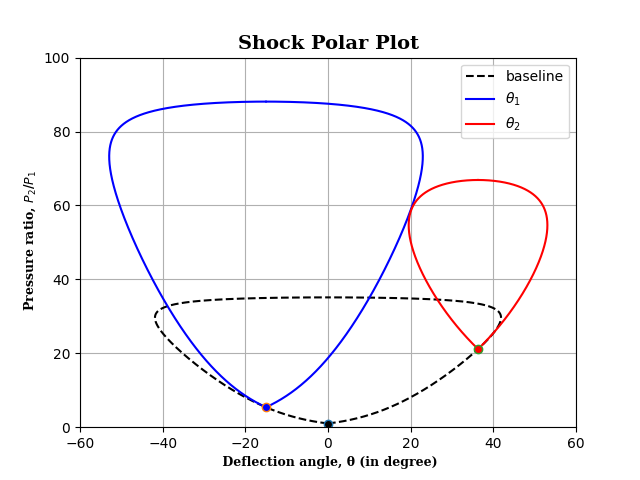


Figure 3: Shock Polar Plot for and

Figure 4: Shock Polar Plot for and

**Conclusion**

A python code to plot the shock polar was written and compiled in VS Studio Code. And through the plot, the condition for the formation of Mach stem, in two wedge configurations, was determined.

**APPENDIX I**

Python Source Code:

# “All that had gone before was not a thousandth of what was yet to come; the story of this star had barely begun.”

import numpy as np

import scipy.optimize as optimize

import matplotlib.pyplot as plt

g = 1.4 # specific heat ratio (gamma)

M = 5.5 # upstream mach number

theta\_1 = 15 # negative angle

theta\_2 = 36.4

# Possible range of beta and theta for M | Baseline plot | locus of all the possible shock wave formations for M

beta = np.arange(np.arcsin(1/M), np.pi/2, np.pi/10000)

theta = np.rad2deg(np.arctan(2\*(M\*\*2\*np.sin(beta)\*\*2-1)/(np.tan(beta)\*(M\*\*2\*(g+np.cos(2\*beta))+2))))

PR = 1 + (2\*g)/(g+1)\*(M\*\*2\*np.sin(beta)\*\*2-1)

plt.plot(theta, PR, 'k--', label='baseline' )

plt.plot(-theta, PR, 'k--', 0, 1, "o", markerfacecolor='k', linewidth=1.5)

#Function to find beta for a given theta

def calculate\_beta(theta\_val):

    theta\_function = lambda beta\_val, theta\_val:np.arctan(2\*(M\*\*2\*np.sin(beta\_val)\*\*2-1)/(np.tan(beta\_val)\*(M\*\*2\*(g+np.cos(2\*beta\_val))+2)))\*180/np.pi - theta\_val

    beta\_initial\_guess = 0

    beta\_solution = optimize.fsolve(theta\_function, beta\_initial\_guess, args=(theta\_val,))

    return beta\_solution[0]

#Calculation of flow and shock wave properties for theta\_1

beta\_1 = calculate\_beta(theta\_1)

print(np.rad2deg(beta\_1))

Mn1\_1 = M \* np.sin(beta\_1)

Mn2\_1 = np.sqrt((1+(g-1)/2\*Mn1\_1\*\*2)/(g\*Mn1\_1\*\*2-(g-1)/2))

M2\_1 = Mn2\_1/np.sin(beta\_1-np.deg2rad(theta\_1))

PR\_1 = 1 + (2\*g)/(g+1)\*(M\*\*2\*np.sin(beta\_1)\*\*2-1)

beta1 = np.arange(np.arcsin(1/M2\_1), np.pi/2, np.pi/10000)

theta1 = np.rad2deg(np.arctan(2\*(M2\_1\*\*2\*np.sin(beta1)\*\*2-1)/(np.tan(beta1)\*(M2\_1\*\*2\*(g+np.cos(2\*beta1))+2))))

PR1 = (1 + (2\*g)/(g+1)\*(M2\_1\*\*2\*np.sin(beta1)\*\*2-1))\*PR\_1

plt.plot(-(theta\_1+theta1), PR1, 'b', label = '$θ\_{1}$')

plt.plot(-(theta\_1-theta1), PR1, 'b',-theta\_1, PR\_1, "o", markerfacecolor='b', linewidth=1.5)

#Calculation of flow and shock wave properties for theta\_2

beta\_2 = calculate\_beta(theta\_2)

print(np.rad2deg(beta\_2))

Mn1\_2 = M \* np.sin(beta\_2)

Mn2\_2 = np.sqrt((1+(g-1)/2\*Mn1\_2\*\*2)/(g\*Mn1\_2\*\*2-(g-1)/2))

M2\_2 = Mn2\_2/np.sin(beta\_2-np.deg2rad(theta\_2))

PR\_2 = 1 + (2\*g)/(g+1)\*(M\*\*2\*np.sin(beta\_2)\*\*2-1)

beta2 = np.arange(np.arcsin(1/M2\_2), np.pi/2, np.pi/10000)

theta2 = np.rad2deg(np.arctan(2\*(M2\_2\*\*2\*np.sin(beta2)\*\*2-1)/(np.tan(beta2)\*(M2\_2\*\*2\*(g+np.cos(2\*beta2))+2))))

PR2 = (1 + (2\*g)/(g+1)\*(M2\_2\*\*2\*np.sin(beta2)\*\*2-1))\*PR\_2

plt.plot(theta\_2+theta2, PR2, 'r',label = '$θ\_{2}$')

plt.plot(theta\_2-theta2, PR2, 'r', theta\_2, PR\_2, "o", markerfacecolor='r', 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('Shock Polar Plot', fontdict = font\_propt)

plt.xlabel(' Deflection angle, θ (in degree)', fontdict = font\_propl)

plt.ylabel(' Pressure ratio, $P\_{2}$/$P\_{1}}$', fontdict = font\_propl)

plt.xlim([-60, 60])

plt.ylim([0, 100])

plt.grid(True)

plt.legend()

plt.show()