TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

An assignment report

on $\theta - \beta - M$ plot for oblique shockwaves

Submitted by:

Sandip Adhikari

076bas035

Submitted to:

Asst. Professor Sudip Bhattarai

Department of Aerospace and Mechanical Engineering,

IOE, Pulchowk Campus

July 16, 2023

Introduction

An oblique shockwave is a shockwave, inclined at an angle with respect to the upstream flow, that forms when a supersonic flow encounters an angled surface, such as a wedge or an airfoil. It occurs due to the abrupt change in flow direction and deceleration caused by the angled surface.

$\boldsymbol{\theta - \beta - M}$ Relation for an oblique shockwave

It describes the relationship between the incident flow Mach number (M), the shockwave angle (β), and the deflection angle of the flow (θ). This relation is based on the conservation of mass, momentum, and energy across the oblique shockwave.

$\theta - \beta - M$ relation is given by:

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

Where,

M_1 = Incident flow Mach number

θ = shockwave angle; represents the orientation of the shockwave with respect to the flow direction

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

$\gamma \ = \ \frac{c_p}{c_v}$ = ratio of specific heats

Equation (1) specifies θ as a unique function of M_1 and β. This relation is vital to the analysis of oblique shock waves, and results in a plot of wave angle (β) and deflection angle (θ), with Mach number (M) as a parameter. The objective of this assignment is to draw this plot using equation (1) by writing a python program in Visual Studio Code.

Plot

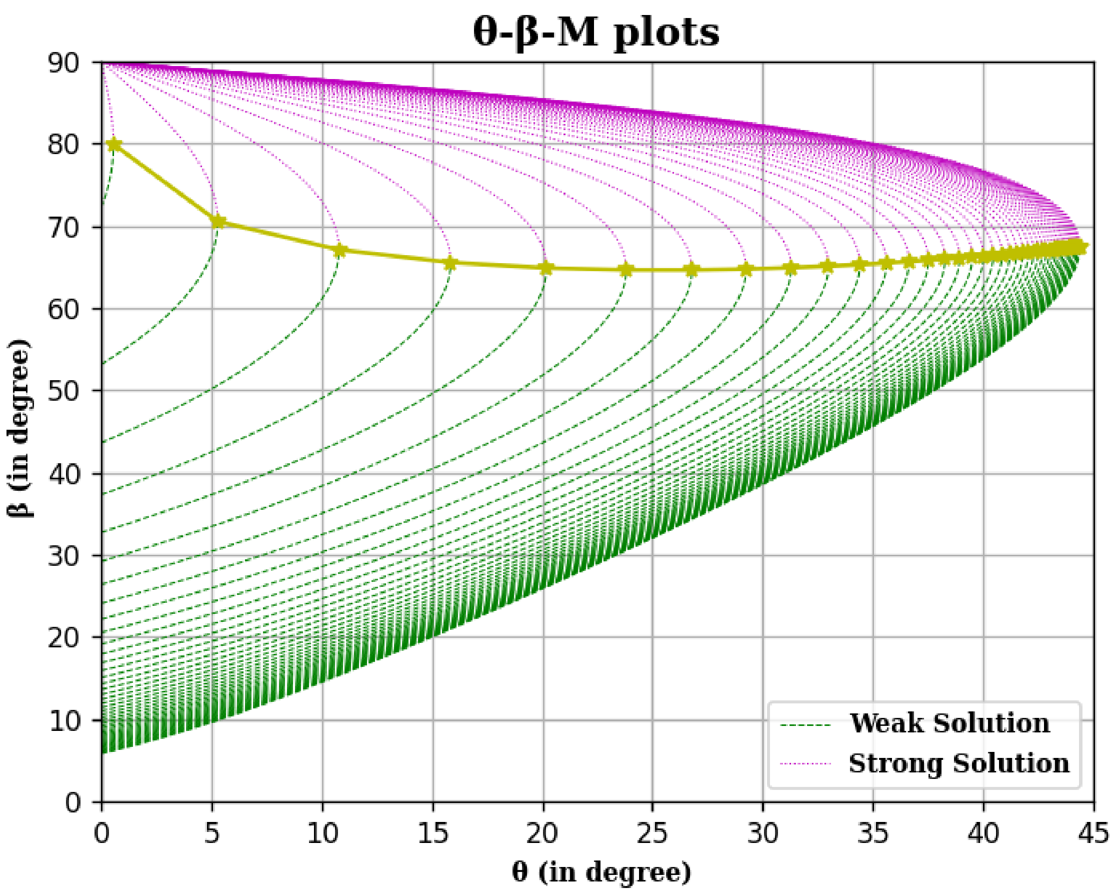

Figure 1: Wave angle (θ) vs deflection angle (β) plot at different upstream Mach numbers (M_1) (M_1 ranges from 1.05 to 10)

For a given Mach number, a curve is formed in the plot of θ vs β which is divided into two parts: weak (green color curve, representing low values of β) and (magenta color curve, representing high values of β) strong solutions, based on the maximum value of θ as shown in the figure 1.

The yellow curve represents the maximum values of θ for each θ vs β plot at the corresponding Mach number. If the geometry of the body, for given Mach number, is such that $\theta > \theta_{\max}$, then no solution exists for a straight oblique shockwave. Instead, nature establishes a curved shock wave, detached from the corner or the nose of the body.

As the Mach number increases, the solution for maximum θ also increases (the curve elongates more to the right side) whereas the value of β for the corresponding $\theta_{\max}$ decreases.

Conclusion

A python code for plotting the $\theta - \beta - M$ plot was written in Visual Studio code and the characteristics of the plot were studied to become more familiar with the concept of oblique shockwaves.

APPENDIX I

Python Source Code:

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

gamma = 1.4

M = np.array(np.arange(1.05, 10, 0.2))
batm = np.zeros(M.shape)
tmax = np.zeros(M.shape)

# loop for plots from M = 1.05 to 10
for i in range(0, M.size):

    #Finding beeta minimum and beeta array
    betamin = np.arcsin(1 / M[i])
    beeta = np.array(np.arange(betamin, np.pi / 2, np.pi / 20000))

    # Finding theeta using Theeta Beeta M relation
    theeta = np.arctan(
        (2 * ((M[i] * np.sin(beeta)) ** 2 - 1) / (np.tan(beeta) * (M[i] ** 2 *
              (gamma + np.cos(2 * beeta)) + 2))))

    # finding theeta max and its position in terms of theeta
    tmax[i] = 180 / np.pi * np.max(theeta)
    max_position = np.argmax(theeta)

    # slicing theeta array into weak and stron solutions
    weak_soln = 180 / np.pi * theeta[0:int(max_position)]
    strong_soln = 180 / np.pi * theeta[int(max_position) + 1:theeta.size]

    # slicing beeta array and finding beeta max
    beta1 = 180 / np.pi * beeta[0:int(max_position)]
    batm[i] = 180 / np.pi * beeta[beta1.size]
    beta2 = 180 / np.pi * beeta[int(max_position) + 1:theeta.size]

    # modified color and line style for strong solution
    plt.plot(weak_soln, beta1, "g--", linewidth=0.6)
    plt.plot(strong_soln, beta2, "m:", linewidth=0.6)

plt.plot(tmax, batm, "y*-")
```

```python
# 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'
}

plt.title('θ-β-M plots', fontdict = font_propt)
plt.xlabel(' θ (in degree)', fontdict = font_propl)
plt.ylabel(' β (in degree)', fontdict = font_propl)
plt.xlim([0, 45])
plt.ylim([0, 90])
plt.grid(True)
plt.legend(['Weak Solution', 'Strong Solution'], prop = font_propl)
plt.show()
```