Python Code Development for Gas Dynamics Problems

By:
Jaydip Patel

Abstract

This report details the development and application of Python codes to solve a series of problems in gas dynamics. The project focuses on three key areas: determining the location of a normal shock in a duct with friction, analyzing the interaction of oblique shock waves of opposite families, and calculating the aerodynamic properties of supersonic flow over a flat plate using shock-expansion theory. By leveraging numerical methods and libraries such as NumPy and SciPy, this work demonstrates the utility of computational tools in solving complex fluid dynamics problems that are often challenging to tackle analytically. The codes are validated against examples from established textbooks, and their limitations are also discussed.

Table of Contents

1. Introduction

Gas dynamics, a field of fluid mechanics, deals with the study of compressible fluid flow and its effects on physical systems. The governing equations are often complex and non-linear, making analytical solutions difficult for many practical scenarios. This project utilizes the Python programming language to develop computational tools for solving a range of gas dynamics problems, drawing upon examples from well-known textbooks in the field. The use of Python, with its powerful libraries like NumPy for numerical operations and SciPy for scientific computing, allows for the efficient implementation of iterative and numerical solution techniques.

2. Problem 1: Normal Shock in a Duct with Friction

Problem Statement

This problem is based on Example 8.3 from Yahya's textbook. It considers a more practical variation where the length of the duct and the supersonic inlet Mach number are given, with the outlet being sonic. The objective is to write a Python code to determine the location of a normal shock within the duct, if one exists.

Methodology

The Python code developed for this problem takes the following inputs:

- L: Length of the tube
- M1: Inlet Mach number
- g: Gamma (ratio of specific heats)
- D: Diameter of the tube

- f: Frictional factor of the tube

The code iteratively guesses the pre-shock Mach number (Mx) and calculates the corresponding lengths of the duct upstream (L1) and downstream (L2) of the shock. The core of the solution lies in minimizing the difference (tolerance or error) between the given total length L and the sum of the calculated lengths L1 + L2.

Results and Validation

The code successfully reproduces the solution of the example problem from Yahya's textbook. For an inlet Mach number of 2, a tube length of 10.8m, a diameter of 0.3m, a frictional factor of 0.003, and gamma of 1.3, the code calculates:

- L1 (length from inlet to shock) = 5.35 m
- L2 (length from shock to outlet) = 5.45 m
- Mach (pre-shock Mach number) = 1.469

Code Limitations

The code may fail to produce a physically possible answer under certain conditions:

- If the given Mach number and duct length do not allow for the formation of a shock.
- If the inlet Mach number is not supersonic.
- If the required L* for the given Mach number is less than the total length of the duct.

3. Problem 2: Interaction of Shocks of Opposite Families

Problem Statement

This problem is based on Example 6.5 from Oosthuizen’s textbook and involves the interaction of two oblique shock waves of opposite families. The goal is to develop a Python code to solve for the properties of the flow after the shock interaction, including the final pressure and the angle of the slipstream.

Methodology

The code utilizes the fsolve function from the SciPy library to iteratively solve the theta-beta-Mach number relationship for the shock angles. The inputs include the upstream Mach number, pressure, temperature, and the two wedge angles. The solution approach is as follows:

1. Calculate the initial shock angles (beta) for each wedge.
2. Determine the flow properties (Mach number, pressure ratio) across each oblique shock.
3. Iteratively solve for the change in angle (delta) of the slipstream required to equalize the pressures in the regions downstream of the reflected shocks (P42 = P43).

Results

For an upstream Mach number of 3, pressure of 30,000 Pa, temperature of 264 K, and wedge angles of 4 and 3 degrees, the code calculates:

- Pressure ratio across the first shock (4-degree wedge) ≈ 1.35
- Pressure ratio across the second shock (3-degree wedge) ≈ 1.26
- Change in angle (delta) for the slipstream ≈ 0.9995 degrees

4. Problem 3: Supersonic Flow Over a Flat Plate (Shock-Expansion Theory)

Problem Statement

This problem is based on Example 4.15 from Anderson (2003) and involves the calculation of supersonic flow over a flat plate at an angle of attack. The objective is to use shock-expansion theory to calculate the lift and drag coefficients (cl and cd) and the precise angle of the slip-line.

Methodology

The code applies the principles of oblique shock waves and Prandtl-Meyer expansion fans. The inputs are the free-stream Mach number, pressure, temperature, and the angle of attack (alpha). The methodology is as follows:

1. **Expansion Fan (Upper Surface):** The Prandtl-Meyer function is used to calculate the change in flow properties as the flow expands over the top surface. The fsolve function is employed to find the Mach number corresponding to the expanded flow.
2. **Oblique Shock (Lower Surface):** The theta-beta-Mach number relationship is solved to find the shock angle and the properties of the flow after the oblique shock on the bottom surface.
3. **Aerodynamic Coefficients:** The lift and drag coefficients are calculated based on the pressure difference between the upper and lower surfaces.
4. **Slip-Line Angle:** The code iteratively solves for the angle (delta) of the slip-line required to equalize the pressures in the regions downstream of the trailing edge.

Results

For a free-stream Mach number of 2.6, pressure of 101,325 Pa, temperature of 273 K, and an angle of attack of 5 degrees, the code calculates:

- Lift Coefficient (cl) ≈ 0.146
- Drag Coefficient (cd) ≈ 0.01
- Change in angle (delta) for the slip-line ≈ 0.003 degrees

5. Conclusion

This project successfully demonstrates the development and application of Python codes for solving a variety of challenging problems in gas dynamics. The use of numerical methods and scientific computing libraries has proven to be an effective approach for analyzing complex flow phenomena. The codes have been validated against textbook examples, confirming their accuracy and utility as educational and analytical tools.

6. References

- Anderson, J. D. (2003). *Modern Compressible Flow: With Historical Perspective*. McGraw-Hill.
- Oosthuizen, P. H., & Carscallen, W. E. (Year of Publication). *Compressible Fluid Flow*. McGraw-Hill.
- Rathakrishnan, E. (Year of Publication). *Gas Dynamics*. Prentice-Hall of India.
- Yahya, S. M. (Year of Publication). *Fundamentals of Compressible Flow*. New Age International.

(Note: The years of publication for some references were not available in the provided files and should be added for completeness.)