INTEGRAL BOUNDARY LAYER METHODS IN PYTHON

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Aerospace Engineering

by

Malachi Edland

August 2021

COMMITTEE MEMBERSHIP

TITLE: Integral Boundary Layer Methods in Python

AUTHOR: Malachi Edland

DATE SUBMITTED: August 2021

COMMITTEE CHAIR: David D. Marshall, Ph.D.
Professor of Aerospace Engineering

COMMITTEE MEMBER: Aaron Drake, Ph.D.
Professor of Aerospace Engineering

COMMITTEE MEMBER: Paulo Iscold, Ph.D.
Professor of Aerospace Engineering

COMMITTEE MEMBER: Colleen Kirk, Ph.D.
Professor of Applied Mathematics

ABSTRACT

Integral Boundary Layer Methods in Python

Malachi Edland

This thesis presents a modern approach to two Integral Boundary Layer methods implemented in the Python programming language. This work is based on two 2D boundary layer methods: Thwaites' method for laminar boundary layer flows and Head's method for turbulent boundary layer flows. Several novel enhancements improve the quality and usability of the results. These improvements include: a common ordinary differential equation (ODE) integration framework that generalizes computational implementations of Integral Boundary Layer methods; the use of a dense output Runge-Kutta ODE solver that allows for querying of simulation results at any point with accuracy to the same order as that of the solver; and an edge velocity treatment method using cubic spline interpolation that improves the simulation performance using only points from an inviscid edge velocity distribution. Both the laminar and turbulent methods are shown to benefit from smoothing of the edge velocity distribution. The choice of ODE solver alleviates the need to artificially limit step sizes. Comparisons against analytic solutions, experimental data and XFOIL results provide a wide varity of verification and validation cases with which to compare. The implementation of Thwaites' method in this thesis avoids simplifications made in other implementations of this method, which results in more robust results. The implementation of Head's method produces high-quality results typically found in other implementations while utilizing the common ODE integration framework. Utilizing the common ODE framework results in significantly less code needed to implement Thwaites' and Head's methods. In addition, the boundary layer solvers produce results in seconds for all results presented here. Boundary layer transition

and separation criteria are implemented as a proof of concept, but require future work.

ACKNOWLEDGMENTS

Thanks to:

- My parents, whose financial support, patience, and love have carried me through 23 years of life, 4 educational institutions, 2 degrees, and endless experiments and projects in the backyard

- My amazing girlfriend, for reassuring me when the hours are long and the work gets tough that I'm capable enough to make it through

- Kari Mobley and ACI Jet for giving me two exhilarating years in the hangar and the Python experience I needed to start this project

- Nick Brake and Cory Seubert for taking me on and providing invaluable work experience at Empirical Systems Aerospace, concurrent and complimentary to this project

- Finally, thank you to my wonderful advisor Dr. David Marshall for taking me under your wing and encouraging me to find new things! I think back often to when I walked into your office and first asked you about Python. Your first words were "sure Malachi, take a seat." The way you said it made me feel so included, and I knew that I'd come to the right person to learn from. More than a year later, I've learned so much, and am so glad that I took that seat.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

Latin

Re_θ Momentum thickness Reynolds number, page 2

c_f Skin friction coefficient, page 3

F_1 Head's method experimental fit to H_1, page 14

H Shape factor, page 3

H_1 Alternative shape factor used in Head's Method, page 13

H_{tr} Shape factor immediately after transition, page 24

m Falkner-Skan velocity distribution exponent, page 6

s Shear correlation, page 8

u Streamwise velocity component at a point in the flow field, page 5

u_∞ Freestream velocity, page 4

u_e Edge velocity, page 4

u_n transpiration velocity, page 42

v Transverse velocity component at a point in the flow field, page 5

Greek

β Falkner-Skan wedge angle parameter, page 5

δ^* Displacement thickness, page 2

δ Boundary layer thickness, page 2

θ Momentum thickness, page 2

ν Kinematic viscosity, page 2

Chapter 1

INTRODUCTION

1.1 Background

The exponential growth of computing power in recent years has made a powerful impact in the development of aeronautics modeling techniques. Many steps in aircraft design processes rely on computational tools that vary widely in precision and simulation times. Modern computational techniques for fluid flows have been driven by the capabilities of the hardware on which they run, becoming more and more complex and accurate, as computational costs decrease.

While high-accuracy solvers are the focus of research labs, lower fidelity models are appreciated in many design shops as a way of obtaining quick, accurate results. In design environments where simulation time savings, workflow, and modularity are extremely valuable, Integral Boundary Layer methods offer valuable results and can quickly return important boundary layer parameters needed for the analysis of a design.

1.2 Boundary Layers

In aerodynamics, the concept of boundary layers has been very important for understanding viscous flows, since the velocity of a low-viscosity fluid flowing around an object changes rapidly near the object's surface. Prandtl was the first to define a *Grenzschicht*, literally *boundary layer*, as the thin layer of viscous influence near a wall in 1904 [15]. While the Navier-Stokes equations for fluid flows already existed at this

point, their complexities had limited aerodynamicists to inviscid solutions before this development [17]. Prandtl's definition would result in a simplification of the Navier-Stokes equations applicable within the boundary layer. The result allowed the flow to be treated as inviscid outside the boundary layer and simplified the Navier-Stokes equations within the boundary layer.

Boundary layers are often characterized by their thicknesses. The boundary layer thickness, δ, is the distance at which the local velocity is equivalent to the freestream velocity, u_∞. The displacement thickness, δ^*, characterizes the displacement of the inviscid streamline from the surface and is calculated as the amount of freestream mass leaving the boundary layer in the direction normal to the surface:

$$\delta^*(x) = \int_0^\infty \left(1 - \frac{u(x,y)}{u_e x}\right) dy \tag{1.1}$$

The momentum thickness, θ, characterizes the amount of momentum lost within the boundary layer, and is calculated as the amount of equivalent freestream momentum lost:

$$\theta(x) = \int_0^\infty \frac{u}{u_e}\left(1 - \frac{u}{u_e}\right) dy \tag{1.2}$$

Other parameters are also used to characterize the boundary layer flow.

The momentum thickness Reynolds number is defined as:

$$\mathrm{Re}_\theta = \frac{u_e \theta}{\nu} \tag{1.3}$$

where ν is kinematic viscosity. Re_θ is a common term used to characterize the boundary layer, including in the methods used for predicting transition. The shape

factor H is the ratio of displacement thickness over momentum thickness:

$$H = \frac{\delta^*}{\theta} \tag{1.4}$$

It is typically used as a criteria for separation and for transition, and also appears in boundary layer approximation methods. The skin friction coefficient c_f quantifies the viscous stresses on the surface. It is defined as the wall shear stress τ_w nondimensionalized by dynamic pressure $q = (1/2)\,\rho u^2$:

$$c_f = \frac{\tau_w}{q} \tag{1.5}$$

The velocity used to calculate q has not always been defined consistently. Some references use the freestream velocity (far upstream of the surface) and some use the velocity at the edge of the boundary layer at the current streamwise location. In this work, c_f is nondimensionalized using the boundary layer edge velocity, u_e to calculated q.

Boundary layers in fluid flows undergo specific changes as they move along the surface. Boundary layers are described as laminar when flow travels smoothly and steadily [17]. Laminar boundary layers are prevalent in low Reynolds number flows. In contrast, boundary layers that are turbulent occur at higher Reynolds numbers and are characterized by random fluctuations in the instantaneous flow properties. They are often described on statistical or time-averaged bases [17].

As a flow moves along a surface, the boundary layer near the surface may change from having laminar qualities close to the stagnation point, to becoming turbulent further downstream. The process by which a laminar flow becomes turbulent is known as transition. Laminar or turbulent flows may also separate from the surface. These changes occur because of the growth of instabilities along the surface [17].

1.3 The Blasius Solution

In 1908, the work of Blasius presented a solution for boundary layer growth on a flat plate with no streamwise pressure gradient [4]. The inviscid velocity distribution for this type of flow is represented as:

$$u_e = u_\infty \tag{1.6}$$

Where at any x point along the surface, the velocity at the edge of the boundary layer, u_e , is equal to the freestream velocity, u_∞. This solution utilizes the similarity property of this type of boundary layer. The boundary layer equations can be transformed into a third order, non-linear ODE that represents a similarity solution for the laminar laminar flow field anywhere within the boundary layer, except at the start of the plate, $x = 0$. The solution at any streamwise location in the boundary layer can be found from solving for the dimensionless stream function $f(\eta)$:

$$ff'' + 2f''' = 0 \tag{1.7}$$

with boundary conditions

$$f(0) = 0, f'(0) = 0; f'(\infty) = 1 \tag{1.8}$$

where η is a dimensionless coordinate $\eta \sim {}^{y}/_{\delta^*}$:

$$\eta = y\sqrt{\frac{u_\infty}{\nu x}} \tag{1.9}$$

At the surface, $\eta = 0$. A value of $\eta = 1$ corresponds to the displacement thickness δ^*. Blasius' work described the velocity distribution along the surface, as well as through

the boundary layer, with the following relations:

$$u = u_\infty f'(\eta) \tag{1.10a}$$

$$v = \frac{1}{2}\sqrt{\frac{\nu_\infty}{x}}(\eta f' - f) \tag{1.10b}$$

where u and v are streamwise and transverse velocity, respectively. This solution provides the following expressions for displacement thickness, momentum thickness, and skin friction coefficient:

$$\delta^* = 1.7208\sqrt{\frac{\nu x}{u_\infty}} \tag{1.11a}$$

$$\theta = 0.664\sqrt{\frac{\nu x}{u_\infty}} \tag{1.11b}$$

$$c_f = \frac{0.664}{\sqrt{\mathrm{Re}_x}} \tag{1.11c}$$

1.4 The Falkner-Skan Solution

In 1930, Blasius' solution was generalized by the work of Falkner and Skan that applied a boundary layer to a wide variety of flows [2]. Potential flow theory was used to find the edge velocity expressions and the physical flows that they represent. One flow that may be represented using the Falkner-Skan equations is a wedge flow. In a wedge flow, a flat surface is positioned at an angle $\pi\beta/2$ to the flow, as shown in fig. 1.1.

The edge velocity, u_e, for Falkner-Skan flow is defined by the following expression:

$$u_e = u_\infty \left(\frac{x}{L}\right)^m \tag{1.12a}$$

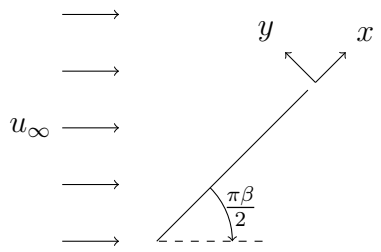

Figure 1.1: Falkner-Skan wedge flow. Note that if $\beta = 0$, the wedge is equivalent to a flat plate.

where

$$m = \frac{\beta}{2 - \beta} \tag{1.12b}$$

and where x is the distance along the surface from the stagnation point [18]. This velocity distribution can represent a number of different flows beyond wedge flows, so it is treated in this project as a general Falkner-Skan flow.

Once again, the boundary layer equations are transformed into a non-linear ODE:

$$f''' + ff'' + \beta[1 - (f')^2] = 0 \tag{1.13a}$$

with the boundary conditions:

$$f(0) = 0, f'(0) = 0; f'(\infty) = 1 \tag{1.13b}$$

A numerical solution must be used to obtain an approximation of f. Similarly to Blasius' solution, f and its derivatives may then be used to obtain δ^*, θ, and c_f. The displacement thickness, momentum thickness, and skin friction coefficient are given by:

$$\delta^* = \left(\frac{2}{m+1}\right)^{1/2} \left(\frac{\nu x}{u_\infty}\right)^{1/2} \int_0^\infty (1 - f')\, d\eta \tag{1.14a}$$

$$\theta = \left(\frac{2}{m+1}\right)^{1/2} \left(\frac{\nu x}{u_\infty}\right)^{1/2} \int_0^\infty f'(1 - f') d\eta \tag{1.14b}$$

$$c_f = \frac{2\sqrt{\frac{m+1}{2}}}{\sqrt{\mathrm{Re}_x}} \left(f''|_0\right) \tag{1.14c}$$

1.5 Integral Boundary Layer Methods

A different approach to analyzing the boundary layer equations came from Theodore Von Karman in 1921 [19]. He integrated the boundary layer equations through the boundary layer (essentially averaging over the thickness) [19] to arrive at:

$$\frac{d\theta}{dx} + \frac{\theta}{u_e}(2+H)\frac{du_e}{dx} = \frac{1}{2}c_f \tag{1.15}$$

This is known as the *momentum integral equation*. This equation forms the basis for the Integral Boundary Layer (IBL) methods. Empirical relations are needed to complete models based on this equation. Equation (1.15) is valid for both laminar and turbulent boundary layers with the appropriate empirical relations; however, to model the relationships between H, c_f, and θ, different empirical relations are needed based on the boundary layer characteristics. At a given streamwise location on the surface, these methods return the shape factor, momentum thickness, skin friction, and displacement thickness. Two specific Integral Boundary Layer methods were used in this project: Thwaites' method for laminar flows, and Head's method for turbulent flows. These methods are well studied and are generally simpler than other IBL methods, as their ODE's are mostly derived from experimental data or analytic solutions [8].

1.5.1 Thwaites' Method for Laminar Boundary Layers

A method for estimation of laminar boundary layers was presented by Bryan Thwaites in 1949 [3]. This method is 2-dimensional, but may be applied to axisymmetric bodies with the Mangler transformation [8]. In this work, it is used exclusively as a 2D method. Thwaites defined a parameter λ, where:

$$\lambda = \frac{\theta^2}{\nu}\frac{du_e}{dx} \tag{1.16}$$

By arranging the momentum integral equation as follows:

$$\frac{c_f}{2} = \frac{d\theta}{dx} + (2+H)\frac{\theta}{u_e}\frac{du_e}{dx} \tag{1.17}$$

and by multiplying both sides by ${}^{u_e\theta}/_{\nu}$:

$$\frac{\tau_w \theta \nu}{u_e} = \frac{u_e\theta}{\nu}\frac{d\theta}{dx} + \frac{\theta^2}{\nu}\frac{du_e}{dx}(2+H) = s(\lambda) \tag{1.18}$$

He found that two functions, a wall shear correlation, s, and shape factor, H, were approximated fairly well as functions only of λ.

Tabulated values correlating s and H to λ were also provided by Thwaites and were obtained from a number of solutions to the laminar boundary layer equations. Various fits to this tabulated data have been presented, such as those presented in Cebeci and Bradshaw [8] and White [17].

Solving eq. (1.16) for θ^2:

$$\theta^2 = \frac{\lambda\nu}{{}^{du_e}/_{dx}} \tag{1.19}$$

and given that:

$$\theta\left(\frac{d\theta}{dx}\right) = \frac{1}{2}\left(\frac{d\theta^2}{dx}\right) \tag{1.20}$$

Equation (1.18) may be rewritten as:

$$s(\lambda) = \frac{u_e}{2\nu}\left(\frac{d\theta^2}{dx}\right) + \frac{\theta^2}{\nu}\left(\frac{du_e}{dx}\right)(2+H) \tag{1.21a}$$

$$2s(\lambda) = u_e\frac{d}{dx}\left(\frac{\lambda}{{}^{du_e}/_{dx}}\right) + 2\lambda(2+H) \tag{1.21b}$$

$$u_e\frac{d}{dx}\left(\frac{\lambda}{{}^{du_e}/_{dx}}\right) = 2[s(\lambda) - \lambda(2+H)] = F(\lambda) \tag{1.21c}$$

To simplify solving this, Thwaites recognized that if $F(\lambda)$ was of the form $F(\lambda) = a - b\lambda$, then the following exact solution for θ^2 may be obtained:

$$\frac{\theta^2}{\nu} = au_e^{-b}\left(\int_0^x u_e^{b-1}d\zeta + c\right) \tag{1.22a}$$

Where the integration constant c must be zero so that the momentum thickness at the start of the surface, $x = 0$, is not infinite, ${}^{c}/_{0}$. Thwaites found the best fit values to be:

$$a = 0.45, b = 6.0 \tag{1.22b}$$

This yields the final expression of:

$$\theta^2 = \frac{.45\nu}{u_e^6}\int_0^x u_e^5 dx \tag{1.23}$$

Implementations of Thwaites' method typically carry out this integral on the velocity distribution (possible analytically with a known velocity distribution, or numerically with a simple quadrature technique). With θ calculated at a given point and ${}^{du_e}/_{dx}$ either solved for analytically or approximated in order to determine λ using eq. (1.16), s and H may be estimated using the aforementioned empirical relationships.

While the assumed relationship for F in equation eq. (1.21c) makes this method easier to solve, it also sacrifices the accuracy of the tabulated values Thwaites provided for s and H (as well as the fits provided by others) by introducing potential error in θ from the linearization of $F(\lambda)$.

A new method, using the empirical fits for $s(\lambda)$ and $H(\lambda)$, was derived during this project to avoid this linearization and the error it involves.

Presented in fig. 1.2 are the values of $F(\lambda)$ using the linear assumption and two different fits. The associated errors are shown in fig. 1.3. The linear assumption (the most common) produces the most error, as shown in fig. 1.3. The lowest error in F is achieved with the fits to s and H presented by Cebeci and Bradshaw [8] and eq. (1.21c) to solve for F:

$$s(\lambda) \approx \begin{cases} 0.22 + 1.57\lambda - 1.8\lambda^2 & \text{for } 0 \leq \lambda \leq 0.1 \\ 0.22 + 1.402\lambda + \frac{0.018\lambda}{0.107+\lambda} & \text{for } -0.1 \leq \lambda < 0 \end{cases} \tag{1.24a}$$

$$H(\lambda) \approx \begin{cases} 2.61 - 3.75\lambda + 5.24\lambda^2 & \text{for } 0 \leq \lambda \leq 0.1 \\ \frac{0.0731}{0.14+\lambda} + 2.088 & \text{for } -0.1 \leq \lambda < 0 \end{cases} \tag{1.24b}$$

This model, while accurate, applies to a specifically limited range compared to the White fits [17]:

$$s(\lambda) \approx (\lambda + 0.09)^{0.62} \tag{1.25a}$$

$$\begin{gathered} H(\lambda) \approx 2 + 4.14z - 83.5z^2 + 854z^3 - 3337z^4 + 4576z^5 \\ \text{where } z = (0.25 - \lambda) \end{gathered} \tag{1.25b}$$

Figure 1.3 shows that by using a linear approximation for F, significant error is introduced at certain values of λ. While splicing the different fits may have shown slight improvement, this project instead implements a cubic spline between Thwaites'

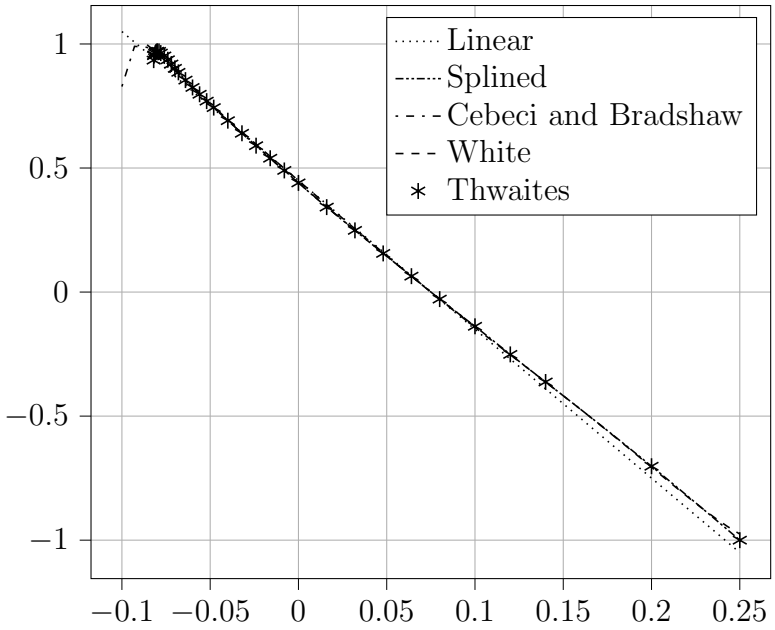

Figure 1.2: Linearized $F(\lambda)$, ($F(\lambda) = .45 - 6\lambda$), splined $F(\lambda)$ from Thwaites' original values, and $F(\lambda)$ using fits to $s(\lambda)$ and $H(\lambda)$ presented by Cebeci and Bradshaw [8] and White [17].

original values to return s and H as functions of λ. This ensures zero error in F at Thwaites' points, and a smooth derivative through the entire distribution.

Using a Runge-Kutta solver, this project's implementation of Thwaites' method eliminates the need to assume that F is linear. The details of this change are described more in section 2.2. For comparison with known analytical results, Thwaites' method may be applied analytically to Falkner-Skan flow. Given the Falkner-Skan velocity distribution described by eq. (1.12a), Thwaites' integral may be applied to calculate θ^2 at a given point:

$$\theta^2(x) = \frac{.45\nu}{u_\infty^6 \left(\frac{x}{L}\right)^{6m}} \int_0^x u_\infty^5 \left(\frac{x}{L}\right)^{5m} \tag{1.26}$$

which may be simplified to

$$\theta(x) = \sqrt{\frac{.45 L^m \nu x^{1-m}}{u_\infty (5m+1)}} \tag{1.27}$$

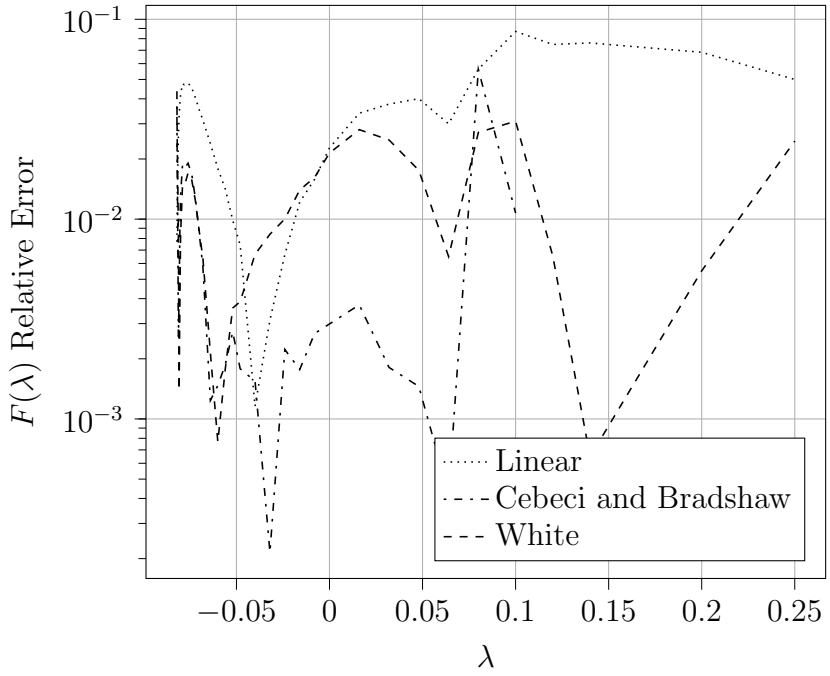

Figure 1.3: Error introduced using linearized $F(\lambda)$ ($F(\lambda) = .45 - 6\lambda$), fits presented by Cebeci and Bradshaw [8], and fits presented by White [17], against Thwaites' original tabulated values. Note that that no error for the spline (presented later) is shown, as its error is 0 at every one of Thwaites' points.

For various values of m, the error of this calculation is within 10% for the momentum thickness for Falkner-Skan flow, eq. (1.14b) [17].

1.5.2 Head's Method for Turbulent Boundary Layers

M. R. Head first described his method for solving for a turbulent boundary layer in 1958 [6]. This is a 2-dimensional method, but can be applied to axisymmetric bodies with the appropriate transformation. His method used a new shape factor, H_1, based on the entrainment velocity [14]. Similar to the definition of H in eq. (1.4),

$$H_1 = \frac{\delta - \delta^*}{\theta} \tag{1.28}$$

Given the momentum integral equation:

$$\frac{d\theta}{dx} + \frac{\theta}{u_e}(2+H)\frac{du_e}{dx} = \frac{1}{2}c_f \tag{1.15 revisited}$$

a relation for the entrainment velocity and the new shape factor, H_1, such as the experimental data created by Cebeci and Bradshaw:

$$F_1(H_1) = \frac{1}{u_e}\frac{d}{dx}(u_e\theta H_1) \approx 0.0306(H_1 - 3)^{-0.6169} \tag{1.29}$$

$$H_1(H) \approx \begin{cases} 3.3 + 0.8234(H-1.1)^{-1.287} & \text{for } H \leq 1.6 \\ 3.3 + 1.5501(H-0.6778)^{-3.064} & \text{for } H > 1.6 \end{cases} \tag{1.30}$$

and the Ludwieg-Tillman relation for c_f, θ, and H:

$$c_f \approx 0.246 \times 10^{-.678H}\,\mathrm{Re}_\theta^{-.268} \tag{1.31}$$

a system of partial differential equations with 2 unknowns, θ and H, may be obtained if u_e and $^{du_e}/_{dx}$ are known as functions of x. This process is described further with the implementation of Head's method in section 2.4.

Cebeci and Bradshaw [8] presented a Runge-Kutta implementation for solving Head's Method, using θ and H as the simulation variables. Given the relationship presented by Ludwieg and Tillman [5] in eq. (1.31), skin friction coefficient can be found as a function of θ and H to eiliminate it from the momentum integral equation, eq. (1.15). Finally, the derivative of H with respect to x can be determined from experimental fits, F_1 as a function of H_1 in eq. (1.29), and H_1 as a function of H in eq. (1.30).

Cebeci and Bradshaw used a simple approximation for $^{du_e}/_{dx}$ using the provided values of the edge velocity in order to obtain $^{dH}/_{dx}$ from eq. (1.29). Their implementation used the form of the fits to F_1 and H_1 to calculate the derivatives needed to implement the solution to the system of ODE's.

Note that $^{dH_1}/_{dH}$ may be obtained analytically when $H_1(H)$ is estimated as in eq. (1.30).

By applying the chain rule to eq. (1.29) and using the fits to $F_1(H_1)$ and $H_1(H)$, the derivative of H with respect to x may be found:

$$\frac{dH_1}{dx} = \left(\frac{dH_1}{dH}\right)\left(\frac{dH}{dx}\right) \tag{1.32a}$$

$$u_e F_1 = \frac{d}{dx}(u_e \theta H_1) = \theta H_1 \frac{du_e}{dx} + u_e H_1 \frac{d\theta}{dx} + u_e \theta \frac{dH_1}{dx} \tag{1.32b}$$

$$u_e \theta \frac{dH_1}{dx} = u_e \theta \left(\frac{dH_1}{dH}\right)\left(\frac{dH}{dx}\right) = u_e F_1 - \theta H_1 \frac{du_e}{dx} - u_e H_1 \frac{d\theta}{dx} \tag{1.32c}$$

$$\frac{dH}{dx} = \frac{u_e F_1 - u_e \frac{d\theta}{dx} H_1 - \frac{du_e}{dx}\theta H_1}{\frac{dH_1}{dH} u_e \theta} \tag{1.33}$$

1.6 Transition Modeling

Near the location of transition, the flow becomes unstable and undamped Tollmien-Schlichting waves appear [17]. They develop within the boundary layer until the flow becomes fully turbulent and transitions. As described in White [17], the transition from laminar flow to turbulent flow and the determination of its location is not built of fundamental *theory*. Rather, tests and fits try to predict the location of transition as a function of various parameters. While many parameters such as pressure gradient and wall roughness may be chosen for functional relationships to determine transition, many of the approximations use only one or two.

One such method is that of Michel [10]. This method uses only one parameter, the local momentum thickness Reynolds number (defined in eq. (1.3)). Re_θ is compared to the Michel criteria, defined by:

$$2.9 \left(\frac{x u_e(x)}{\nu} \right)^{0.4} \tag{1.34}$$

According to this model, transition has occurred at the location where Re_θ surpasses the Michel criteria:

$$\mathrm{Re}_\theta = 2.9 \left(\frac{x u_e(x)}{\nu} \right)^{0.4} \tag{1.35}$$

1.7 Runge-Kutta and Dense Output

Runge-Kutta solvers provide a numerical approximation for the solution to a first order ordinary differential equation (or system of first order differential equations) of the form $y' = f(t, y)$, where f is a function returning the derivative of y at the current value of the independent variable t and the simulation variable y [9]. Runge-

Kutta solvers are often referred to by their order, such as RK4, which indicates a 4^{th} order accurate solver. Solvers may have multiple schemes of different orders running concurrently to maintain an error tolerance between the separate orders.

Runge-Kutta solvers are a natural fit for use in IBL methods. Surface distance x may be used as the independent variable for the simulation. The y vector may be composed of one or more variables specific to each IBL method.

For example, a Runge-Kutta implementation of Thwaites method could implement the following expression for y:

$$y = \left[\theta^2\right] \tag{1.36}$$

resulting in the derivative expression

$$y' = \left[\frac{d\theta^2}{dx}\right] \tag{1.37}$$

These values can be found given u_e, $^{du_e}/_{dx}$, and values of s and H. A Runge-Kutta implementation of Head's method could implement the following expression for y:

$$y = \begin{bmatrix} \theta \\ H \end{bmatrix} \tag{1.38}$$

resulting in the y' vector:

$$y' = \begin{bmatrix} \frac{d\theta}{dx} \\ \frac{dH}{dx} \end{bmatrix} \tag{1.39}$$

where these relations may be obtained from eq. (1.15), eq. (1.29), eq. (1.30), and eq. (1.31).

Runge-Kutta simulations often determine results in as few solution points as possible to reduce simulation times, as long as the error remains within a user defined

tolerance. However, fewer simulation points result in fewer locations with known values of y when examining simulation results. Consider the IBL example above. The user may need a boundary layer parameters at several specific x points. Setting specific simulation points increases the time to run, and requires points of interest to be determined before running each simulation. Interpolating between result points introduces error unless an interpolant is used that is the same order of accuracy as the Runge-Kutta solver. This undermines the accuracy of the solver itself. In order to find accurate results at specific points without recomputing the solution, some solvers have a property known as dense output. Dense output returns a polynomial fit between each pair of simulation points. This polynomial has the same degree of accuracy as the solver. This feature allows for quick querying of results with minimal computational costs and without loss of accuracy.

Chapter 2 presents implementations of Thwaites' and Head's methods using a common Runge-Kutta solver between both implementations. The solver used in this work is a Runge-Kutta of the order 5-4, a 5^{th} order solver that uses a 4^{th} order solution as an error estimate in order to adjust the step size [21]. This chapter also covers various implementation details, such as a common IBL parent class intended for use with a variety of IBL methods, transition modeling, and separation. Chapter 3 discusses the results of the solvers compared against analytical solutions, experimental data, and XFOIL results. Finally, Chapter 4 offers takeaways from the models and opportunities for future work.

Chapter 2

IMPLEMENTATION

2.1 Structure

While Thwaites' Method and Head's method have both been implemented using computational methods (Cebeci and Bradshaw [8] present implementations of both), this work provides a Python implementation that makes use of a common parent class and user interface between the two that can be used as the basis for the implementation of other Integral Boundary Layer methods. Figure 2.1 shows the overall structure of the module. In addition, fig. 2.2 shows the classes implementing laminar separation, boundary layer transition, and turbulent separation. The structure shown in fig. 2.1 serves multiple purposes. There are two major divisions between the object classes in fig. 2.1 that may be made in order to better understand the implementation. The first is a vertical division between Thwaites' and Head's method. The second is a lateral division between the `SimData` classes and `Sim` classes.

2.1.1 Parent Classes

The division between Thwaites' and Head's methods is the more trivial of the two. Thwaites' and Head's methods are separate methods for laminar and turbulent analysis, so this division shows the classes associated with each method. This division also highlights their parallel implementations and common inheritances from the `IBLSim` and `IBLSimData` classes.

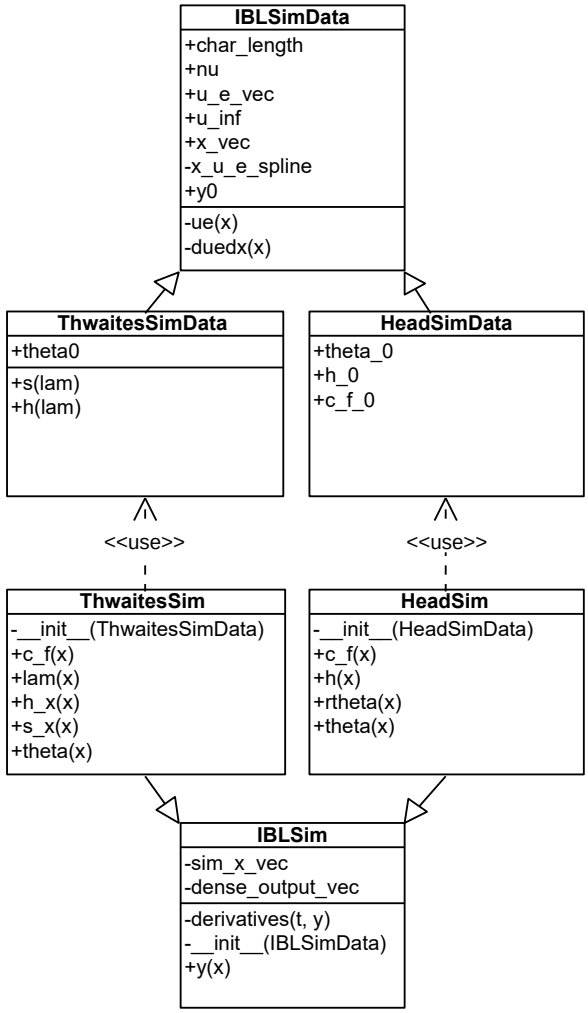

Figure 2.1: Structure of the object classes within PyBL.

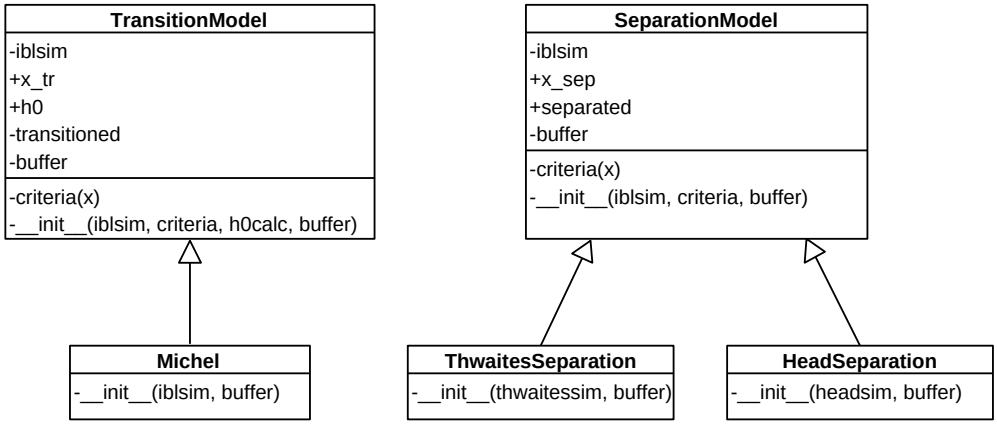

Figure 2.2: Separation and transition object classes within PyBL.

Integral Boundary Layer methods share common attributes; they manipulate the momentum integral equation, eq. (1.15), and relationships using an edge velocity distribution and other boundary layer parameters to arrive at a system of differential equations that may be solved for over x. When pursuing the problems computationally, it follows that certain details can be abstracted from any Integral Boundary Layer method and reused for other methods of the same category, in order to avoid having to implement the same features more than once.

Implementing these two methods concurrently demanded that this relationship be abstracted into a parent class. Both simulations need velocity distributions as functions of x and one or more boundary conditions to run, and need to return results. A user running a simulation through transition would make use of both models and should not have to learn two separate conventions in order to implement them. At the very least, a parent class maintains that the user interfaces between models are consistent. In addition, updates and revisions that would benefit both IBL models (or future models using the same parent classes) may be implemented in the parent classes, as opposed to updating every model.

As shown in section 1.7, both simulations require current estimates of u_e and $^{du_e}/_{dx}$ at any simulation point in order to complete their calculations of f'. In order to accomplish this when given only points from a velocity distribution, a cubic spline was implemented in the `IBLSimData` class [22]. The spline not only returns both the edge velocity and $^{du_e}/_{dx}$ as functions of x, but also ensures a smooth first derivative for optimal behavior of the Runge-Kutta solver. It uses an edge condition that sets the first and second segment at each curve end with the same polynomial. Internal testing showed that this edge condition was appropriate for these results.

2.1.2 User Interface

The division between the `Sim` and `SimData` classes is a user interface feature that uses object classes to differentiate between the setup and results of each simulation. The `SimData` classes receive and process the data, creating the velocity distribution spline and saving the appropriate initial conditions for access by the simulation. Any modification made from one simulation to the next can be made on this class. For example, changing the momentum thickness at the stagnation point does not require creating simulation data from scratch. This class does not represent any simulation results however, only the appropriate inputs.

The `Sim` classes accept only a `SimData` class as an input from the appropriate simulation. Creating a `Sim` class runs the simulation, storing dense output polynomials from between each pair of solution points. In order to query the results, the `IBLSim` parent class contains a method that takes one or more x values, finds the appropriate polynomial fit, and returns the appropriate y value or vector at that point. Other boundary layer parameters may be extracted from y by the simulation-specific objects.

Because the simulation inputs and the simulations are implemented as objects instead of dictionaries or arrays, their attributes have permissions that keep them from direct access to the user. This prevents values being edited erroneously, and tracks values that are left modifiable by the user.

2.2 Thwaites' Method

This implementation of Thwaites' method presents substantial modifications to the traditional implementations. An implementation using a Runge-Kutta solver to solve

Thwaites' method improves the predictive results by eliminating assumptions that were previously needed to find a solution, with similar computational costs.

Having defined the Thwaites parameter λ: eq. (1.16),

$$\lambda = \frac{\theta^2}{\nu}\frac{du_e}{dx} \qquad \text{(1.16 revisited)}$$

the momentum integral equation can be written as:

$$\frac{u_e}{\nu}\frac{d\theta^2}{dx} = 2(-[H(\lambda)+2]\lambda + s(\lambda)) = F(\lambda) \approx .45 - 6\lambda \qquad (2.1)$$

Although s and H are treated as functions of λ when querying results and are calculated from empirical fits, traditional implementations of Thwaites' method approximate the relationship $2(-[H(\lambda)+2]\lambda + s(\lambda))$, or $F(\lambda)$, as a linear relationship. This linear approximation allows for the Thwaites integral, eq. (1.23) (restated below) to be solved by hand for known velocity distributions:

$$\theta^2 = \frac{.45\nu}{u_e^6}\int_0^x u_e^5 dx \qquad \text{(1.23 revisited)}$$

When arranging this method for the Runge-Kutta integrator, eq. (2.1) may be solved for $d\theta^2/dx$ instead:

$$f' = \frac{d\theta^2}{dx} = \frac{\nu(.45 - 6\lambda)}{u_e} \qquad (2.2)$$

Using θ^2 as the simulation variable y, λ can be obtained from θ^2 and the velocity spline, and this can be propagated downstream. Implemented in this way, this method solves the same problem as traditional Thwaites method implementations.

Using the same method to estimate λ, and using fits to s and H, the assumption of linearity for F may be instead removed so that eq. (2.1) then becomes:

$$f' = \frac{d\theta^2}{dx} = \frac{2\nu(s(\lambda) - (2 + H(\lambda))\lambda)}{u_e} \tag{2.3}$$

This uses more detail on the front end, solving for θ using s and H.

In addition, a cubic spline was used to estimate s and H using Thwaites' original tabulated values. Fits introduced by Cebeci and Bradshaw [8] and White [17] introduce up to 10% error at some points and are limited in their applicable ranges. In addition, their piecewise nature does not ensure a smooth derivative for the entirety of λ. Fitting a cubic spline to Thwaites' values eliminates both of these issues.

With more care taken in modeling θ, Thwaites' method was implemented in a more consistent form to its derivation, and with more direct influence from Thwaites' values of s and H.

The module assumes that the starting point for `ThwaitesSim`, `x_vec[0]`, is a stagnation point, unless `theta0` is given. The momentum thickness is determined using the following stagnation condition presented in Moran [14]:

$$\theta_0 = \sqrt{\frac{.075\nu}{\frac{du_e}{dx}\Big|_0}} \tag{2.4}$$

Note that the slope of the edge velocity distribution is used here and can have a significant impact on the boundary layer predictions. In addition, this relationship requires the edge velocity derivative at zero to be infinite for zero momentum thickness.

For the special case of a flat plate, θ_0 may be set equal to 0. If the velocity distribution does not begin at the stagnation point, θ_0 must be known and set with `theta0`. The

user may specify a starting point, `x0`, if not starting at the first point of the velocity distribution, but must also specify `theta0`.

2.3 Michel Transition

In order to provide transition analysis for Thwaites' method and accompanying initial conditions for Head's method, this implementation utilizes the Michel transition criterion introduced in section 1.6. The Michel criteria is a satisfactory approximation and is a sufficient proof of concept to show that the laminar and turbulent solvers could be tied together to simulate transition. In this respect, more focus was given to the laminar and turbulent solvers themselves.

The transition model uses a parent class, `TransitionModel`, in order to make some of the transition features available for other models. `Michel` is a class that takes a completed simulation (`SimData`) as an input, and has methods and attributes that find and describe the transition point. Taking advantage of the laminar solver's dense output, the implementation of the Michel criteria inherits an iterative method from its parent class to find the point of transition, where

$$\mathrm{Re}_\theta = 2.9 \left(\frac{x u_e(x)}{\nu} \right)^{.4} \qquad \text{(1.35 revisited)}$$

After this point, the `Michel` class will return an estimated value of the shape factor after transition, H_{tr}, . The method uses the following relationship presented by Coles [11]:

$$H_{tr} = \frac{1.4754}{\ln(\mathrm{Re}_\theta |_{x_{tr}})} + .9698 \qquad (2.5)$$

The `Michel` object returns this value after solving for x_{tr}.

2.4 Head's Method

This implementation of Head's method follows the implementation presented by Cebeci and Bradshaw [8] fairly closely, using the same empirical fits. $H_1(H)$ is fit with a piecewise function, restated below:

$$H_1(H) \approx \begin{cases} 3.3 + 0.8234(H - 1.1)^{-1.287} & \text{for } H \leq 1.6 \\ 3.3 + 1.5501(H - 0.6778)^{-3.064} & \text{for } H > 1.6 \end{cases} \quad \text{(1.28 revisited)}$$

In order to determine ${}^{dH_1}/_{dH}$, it is useful to note that both segments are of the form:

$$H_1(H) = 3.3 + a(H - b)^c \tag{2.6}$$

and that their derivatives are of the form:

$$\frac{dH_1}{dH} = ac(H - b)^{c-1} \tag{2.7}$$

${}^{d\theta}/_{dx}$ and ${}^{dH}/_{dx}$ may then be obtained as described in section 1.5.2.

2.5 Laminar and Turbulent Separation

Two separation models were implemented in this project for comparison with test cases, based on the possibility that nonphysical or inconsistent results could be the product of separation. A `SeparationModel` parent class was created similarly to the `TransitionModel` class, with methods to optimize for the x value of separation using the solvers' dense output capabilities.

The criteria used for laminar separation using Thwaites method is a λ value of:

$$\lambda < -.0842 \tag{2.8}$$

Which corresponds to a value of $s(\lambda) = 0$. This value was noted by Moran [14] along with correlation formulas for λ, and the assertion that the wall shear stress is 0 at separation.

For Head's method, the criteria used for separation was instead the value of H. This was due to the Ludwieg-Tillman relation for $c_{fLT} = 0.246 * 10^{-.678H} R_\theta^{-.268}$ tending to 0 as H grows. Shape factor was shown to have sufficient correlation with turbulent separation when:

$$1.8 < H < 2.4 \tag{2.9}$$

The specificity of the value was not seen as critical since the derivative of H was also very high at separation. This relationship was also prescribed by Moran [14].

No criteria was implemented to simulate reattachment. If the boundary layer was found to have separated at any point, results downstream of that point were said to be invalid, since the solvers had no implementation to account for a separated boundary layer.

Chapter 3

TEST CASES AND RESULTS

3.1 Testing Thwaites' Method Against Falkner Skan

Because Falkner-Skan flow can be solved analytically with Thwaites' method, it was chosen as a test case for the current method. This series of tests allowed for comparison between the Falkner-Skan solution, Thwaites' method solved analytically, a Runge-Kutta implementation with linearized $F(\lambda)$, and finally, a Runge-Kutta implementation with non-linearized $F(\lambda)$, interpolating Thwaites' tabulated values to determine s and H as functions of λ. For these flows, $d\theta^2/dx$ is infinite at zero, so these simulations begin at an offset from zero. Chapter A describes the reason for this choice in more detail.

These various cases produce results with accuracies consistent with Thwaites' method. Thwaites' method is known to be within 10% of exact results before separation [17]. The analytic solution using Thwaites' method matches the Falkner-Skan solution closely, but not exactly. Section 1.5.1 shows how Thwaites' method may be used to calculate the momentum thickness for Falkner-Skan flow. In addition, the simulations with non-linearized $F(\lambda)$ were expected to differ from the numerical solutions using the linearized $F(\lambda)$, since different equations are be solved.

Even the solution between the numerical and analytical solutions using linearized $F(\lambda)$ (posed to solve mathematically equivalent cases) were expected to differ, as a result of using a numerical solver.

The solver was tested at various values of m. Three are presented here. As noted earlier, Falkner-Skan flow with a value of $m = 0$ reduces to Blasius flow. Each of the numerical methods was given a starting value of θ_0 from the analytic Falkner-Skan result.

The three simulations are meant to compare against known results and validate the performance of the algorithm in controlled conditions. The last solution in this section tested the resilience of this implementation and made a naive attempt at solving without the analytical solution for θ_0, instead implementing the Moran stagnation condition. While it is important that this algorithm perform well with specific initial conditions, the fourth case is intended to show its robustness in cases where the initial conditions are not known.

The results were compared against the Falkner-Skan solution and the analytical Thwaites solution for the corresponding m and u_∞ values. While the derivative of the velocity profile may be derived analytically, the Runge-Kutta solver was given only points from the velocity distribution in order to show the ability of the cubic spline to sufficiently represent the situation.

3.1.1 Falkner-Skan Flow, m=0

At $m = 0$, the Falkner-Skan solution simplifies to the Blasius result. This results in a uniform edge velocity distribution $u_e = u_\infty$. Figure 3.1 demonstrates that the solver results match very closely with the analytical solution using Thwaites.

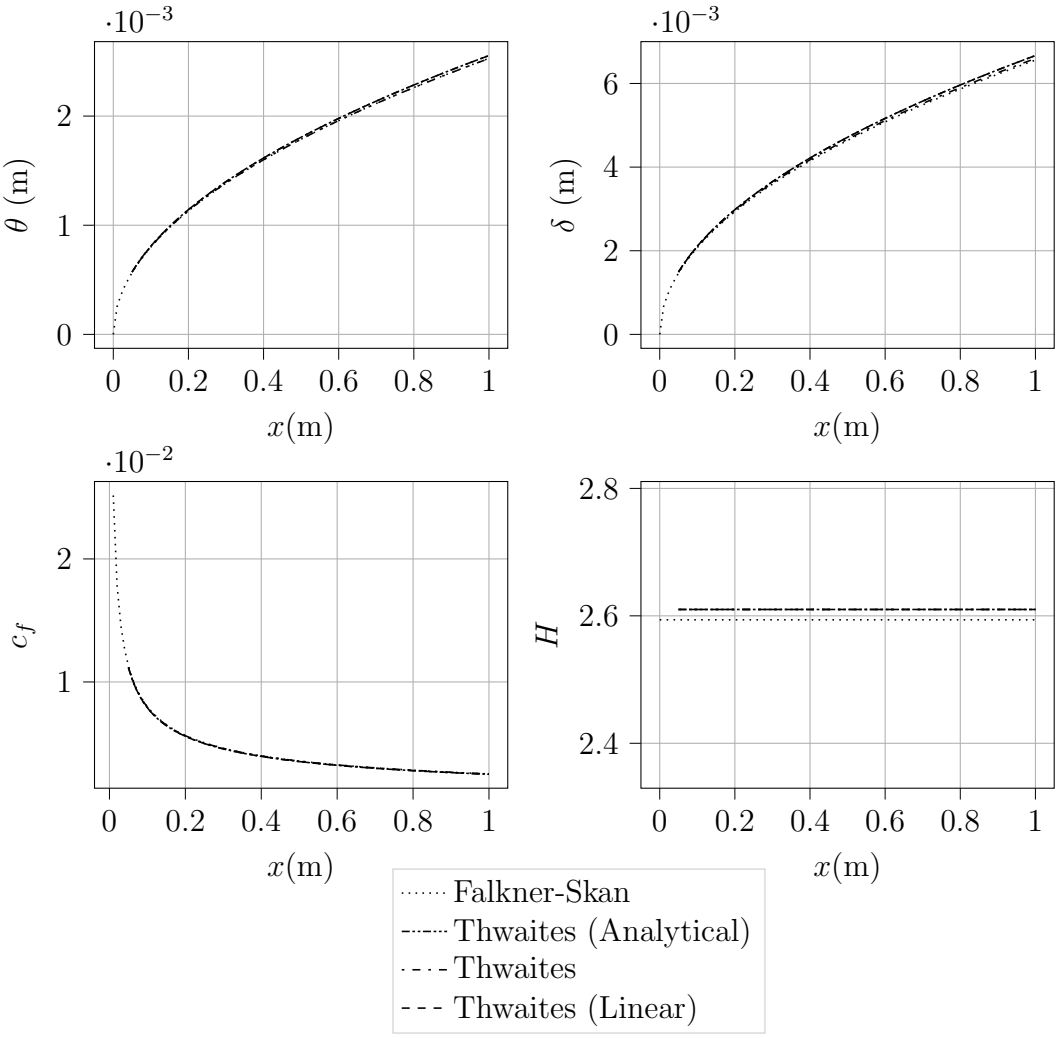

Figure 3.1: Falkner-Skan Flow, m=0 Results. For this flow, the analytical, linear, and interpolated Thwaites method are very consistent.

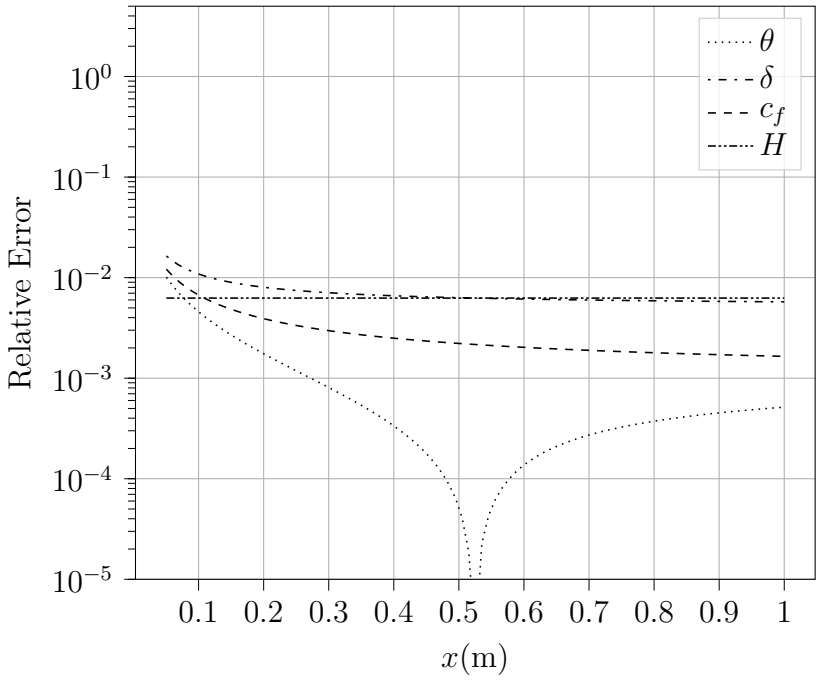

Figure 3.2: Falkner-Skan Flat Plate Flow Error.

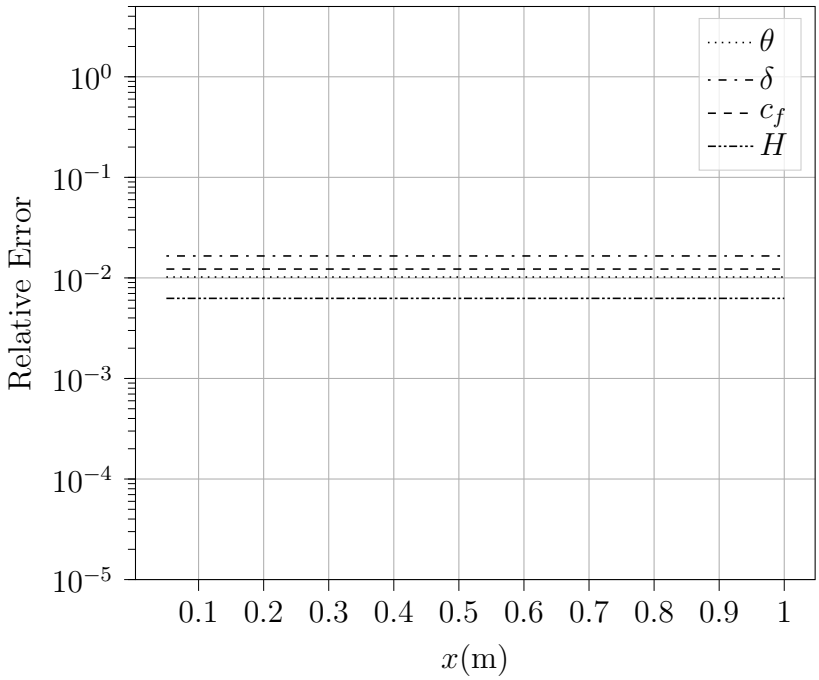

Figure 3.3: Falkner-Skan Flat Plate Flow - Linear Method Error.

3.1.2 Falkner-Skan Flow, m=1

For this flow, the Falkner-Skan velocity distribution simplifies to:

$$u_e = u_\infty \left(\frac{x}{L}\right) \tag{3.1}$$

with the velocity derivative simplifying to:

$$\frac{du_e}{dx} = \frac{u_\infty}{L} \tag{3.2}$$

This simulation's results are presented in fig. 3.4, and the errors for each simulation are presented in fig. 3.5 and fig. 3.6. This result shows that the linearized and interpolated Thwaites produce similar results for this condition to the analytic solution. While the solution begins at the given value of θ_0, it quickly gravitates towards the Thwaites result and has similar error.

3.1.3 Falkner Skan Flow, m=1/3

While the $m = 0$ and $m = 1$ examples both gave useful results for quantifying the effectiveness of the various new implementations of Thwaites, they were both fairly special cases of Falkner-Skan flow. The case with $m = 0$ results in constant edge velocity, and both $m = 0$ and $m = 1$ have constant derivatives.

A value of $m = 1/3$ was chosen as an in-between value to display testing against a more average Falkner-Skan flow. The results for a simulation at this condition are displayed in fig. 3.7.

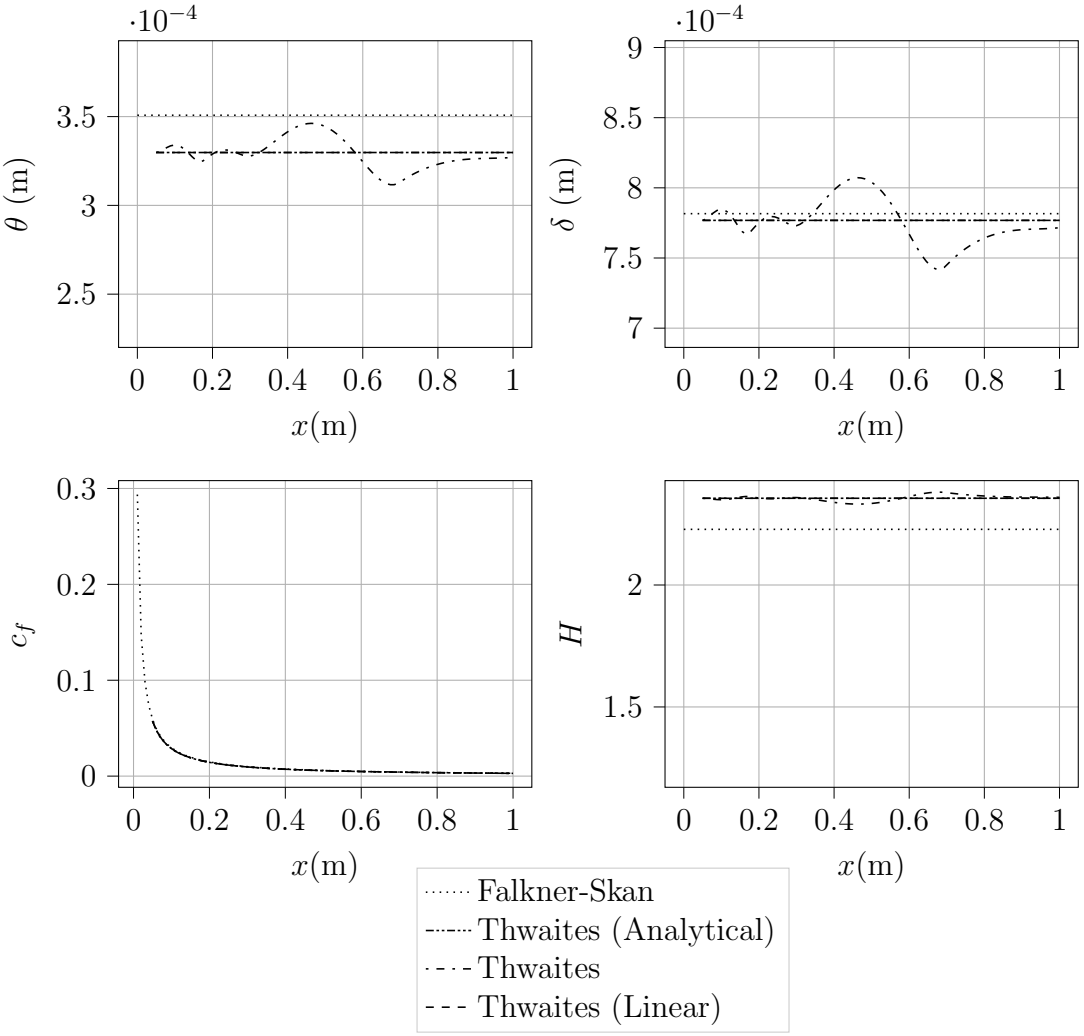

Figure 3.4: Results for Thwaites' method on Falkner-Skan Flow, m=1. This result shows more fluctuation in the results.

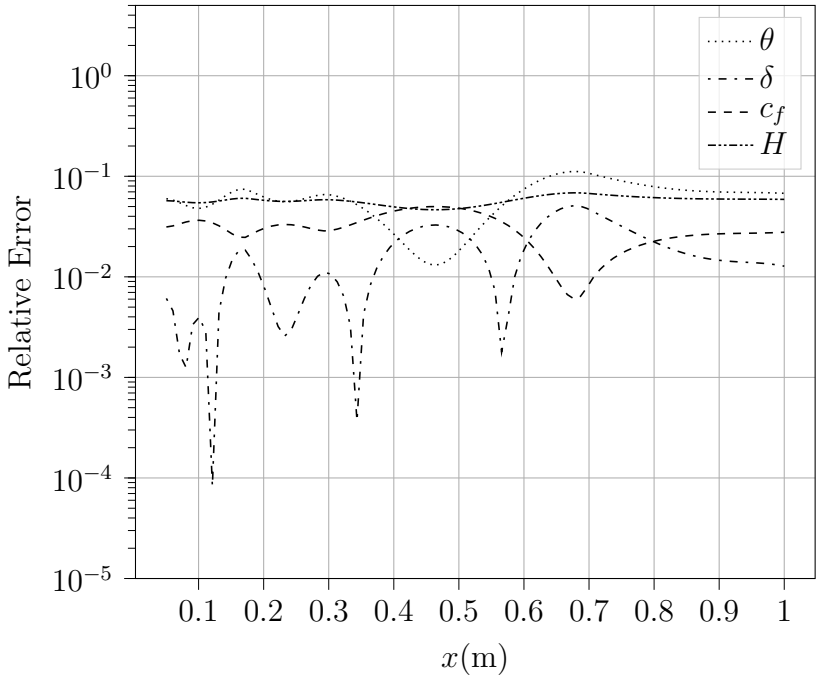

Figure 3.5: Falkner-Skan Flow Error, m=1. This result was generated using the full estimation of $F(\lambda)$ with cubic fits for s and H.

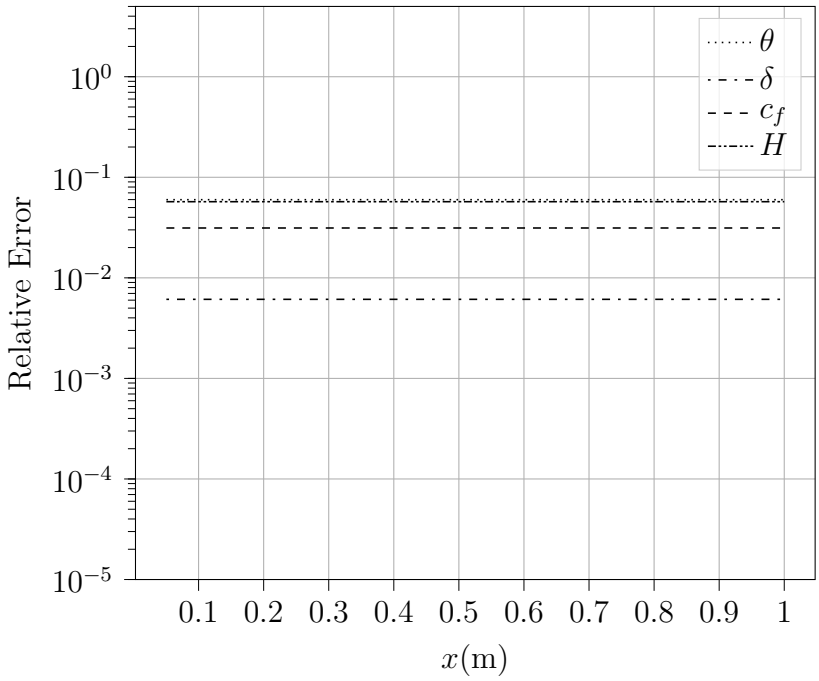

Figure 3.6: Falkner-Skan Flow Error. This result was generated using the linear estimation of $F(\lambda)$ with the cubic fits for s and H used only after θ had been solved.

The spline for the $m = 0$ and $m = 1$ cases was given a relatively simple velocity distribution to approximate for the other two cases, given that their m values both eliminated the exponential qualities of their velocity distributions. In comparison, this simulation left an exponential function to be estimated by a cubic spline.

Figure 3.7, fig. 3.8 and fig. 3.9 show the total results and associated errors for this simulation. These results show similar errors to the $m = 0$ and $m = 1$ cases, regardless of the increased complexity of the edge velocity distribution. Testing showed that the interpolated values from Thwaites' paper maintained similar error to the linearized method.

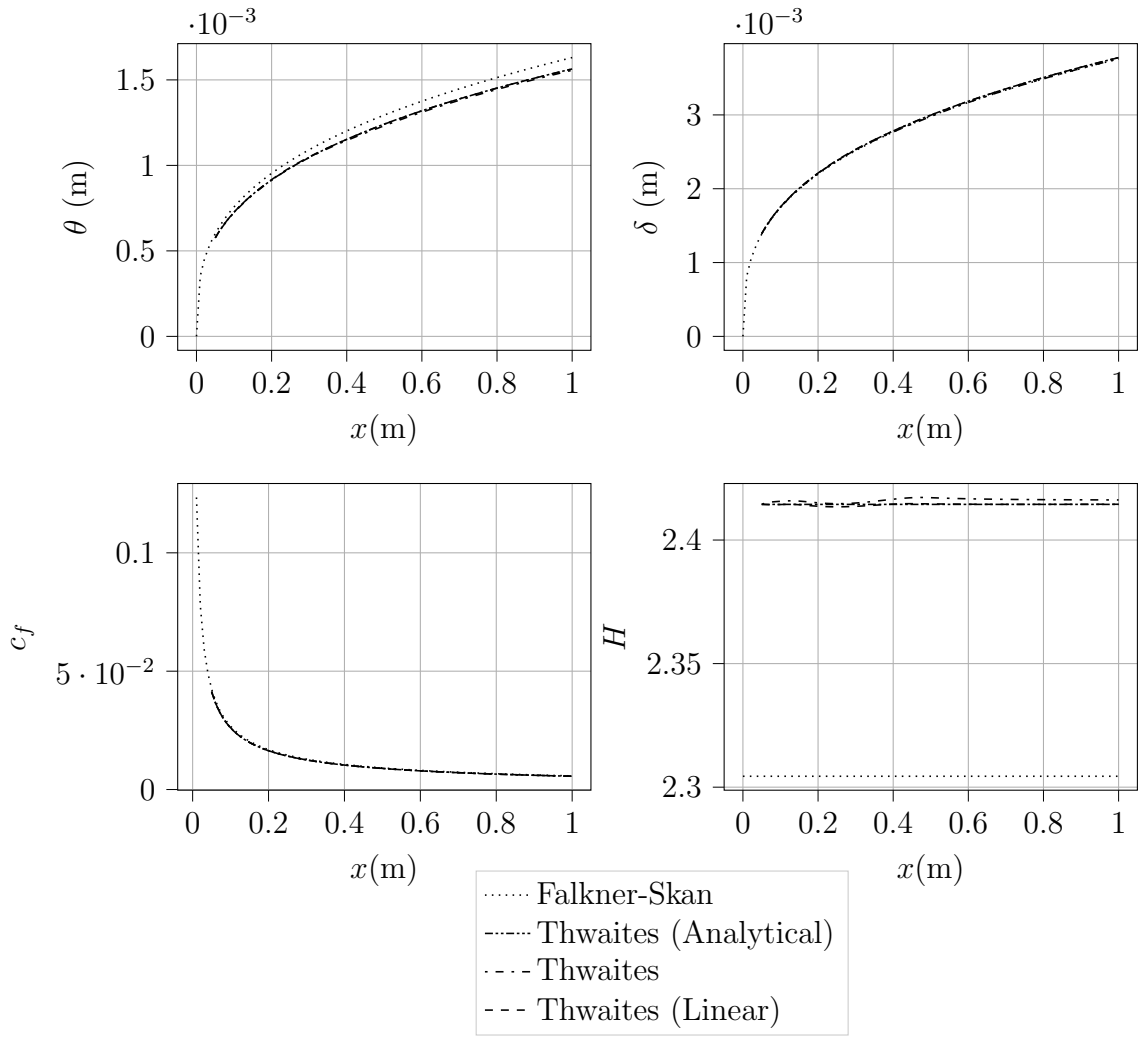

Figure 3.7: Falkner-Skan Flow Results, m=1/3. These results are the result of a more complex velocity distribution.

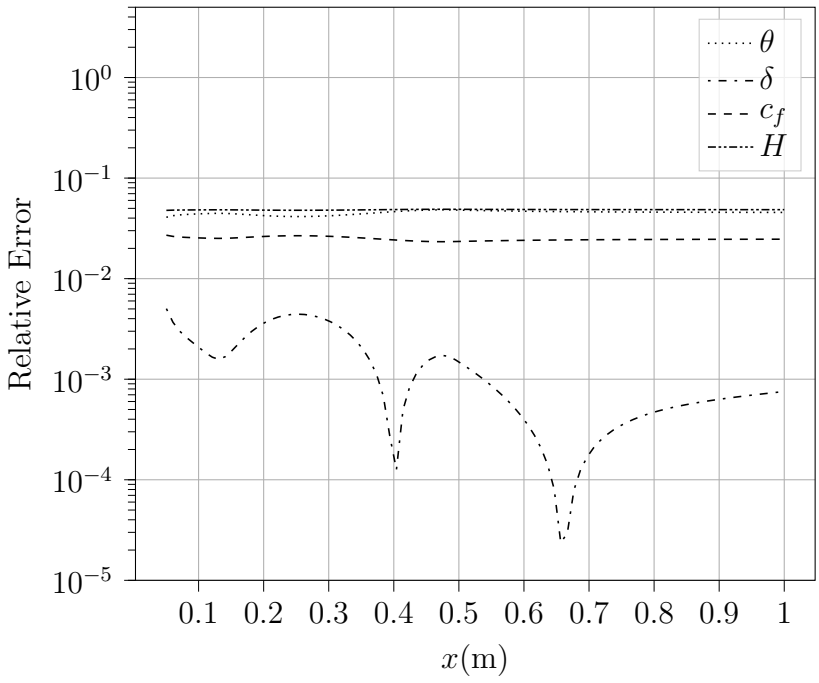

Figure 3.8: Falkner-Skan Flow Error, m=1/3. This error was calculated using the Falkner-Skan result as reference. These results are well within the 10% error reported for the analytic solution.

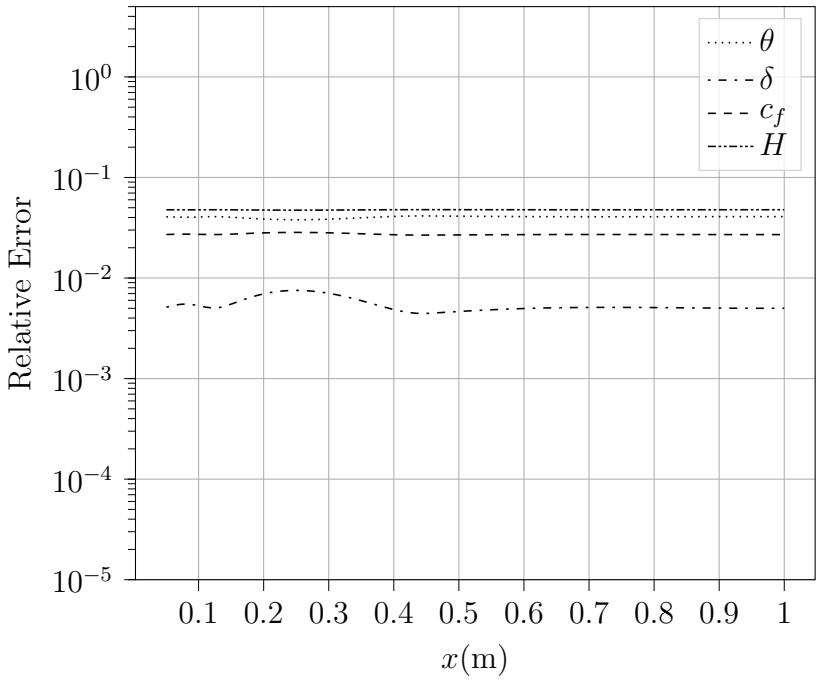

Figure 3.9: Falkner-Skan Flow Error, m=1/3, using linearized F. This simulation was analogous to the analytical solution, which also had an offset error from the Falkner-Skan solution.

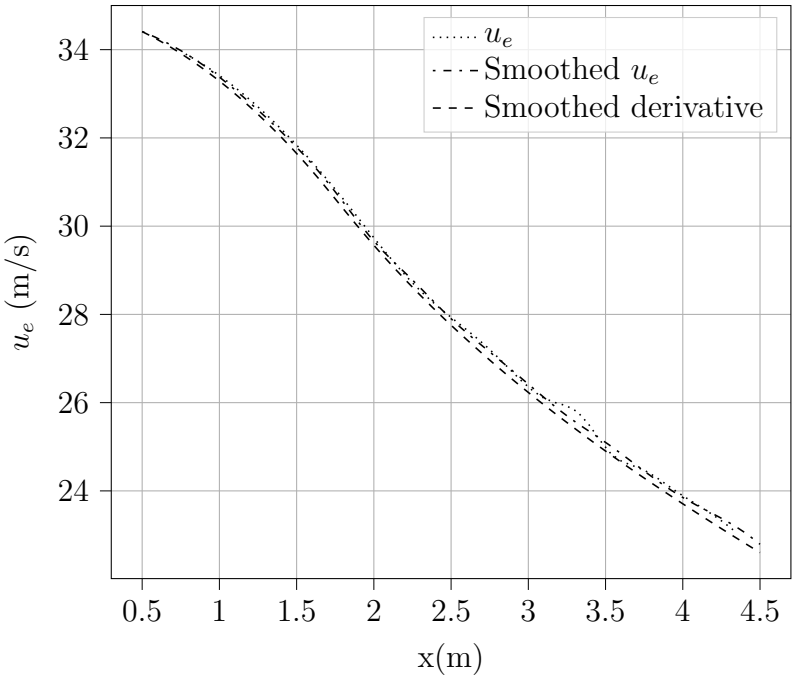

Figure 3.10: Comparison of different velocity inputs to Head's method for a flat plate with mild adverse pressure gradient

3.2 Head's Method on Flat Plate with Mild Pressure Gradient

In order to test the accuracy of the Head's method implementation, simulations were run to compare results with a test case presented by Ludwieg and Tillman [5]. The test case was a flat plate with a mild adverse pressure gradient.

Verification against this specific set of experimental results demonstrated the relationship between the smoothness of the Head implementation's inputs and its ability to produce useful outputs. Experimental data presented by Ludwieg and Tillman [5] (and, notably, tabulated in [7]) was used to run the turbulent simulation. The newer, tabulated data was presented with two plotted results: the edge velocity, u_e as a function of x, and the derivative of edge velocity, $^{du_e}/_{dx}$. Both papers note that this data had been smoothed, but the exact method by which it was smoothed was not described.

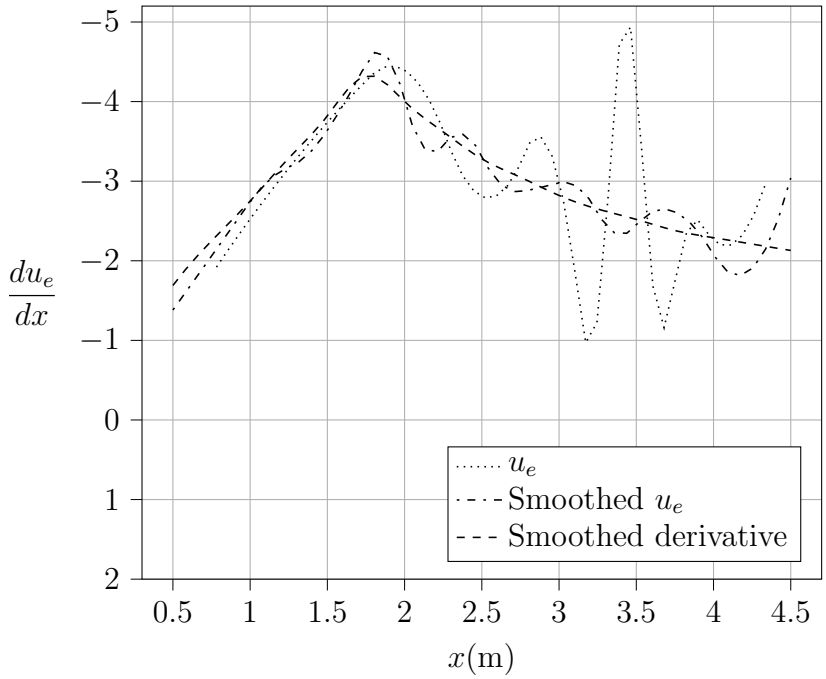

Figure 3.11: Comparison of different velocity derivatives for each spline input to Head's method for a flat plate with mild adverse pressure gradient

In order to run this simulation, points for edge velocity were used from the tabulated data, extracted from the smoothed plot, and extrapolated from $^{du_e}/_{dx}$ values taken from the smoothed derivative plot. Figure 3.10 shows the three splines that were generated from points taken from the tabulated values and smoothed u_e and derivative plots. Note that the smoothing omitted a small discrepancy near $x = 3.3$.

While the splines in fig. 3.10 seem to match fairly closely, the derivatives of the splines (shown in fig. 3.11) show significant differences. The splines from the tabulated u_e and smoothed u_e were generated fitting only the points without consideration for the smoothness of the derivative. The third spline, because it was built from the smoothed, derivative-extrapolated points, provided an additional degree of smoothness. Its antiderivative spline (and appropriate integration constant) were used to run a third simulation.

These different velocity conditions had significant impacts on the boundary layer results. Comparing a variety of results for different boundary layer parameters in fig. 3.12, it is clear that smoothing the derivative made a significant difference in the accuracy of the program and produced results more closely matching the experimental results compared to the other edge velocity treatments. Boundary layer data shown are the reported values from Kline et al [7].

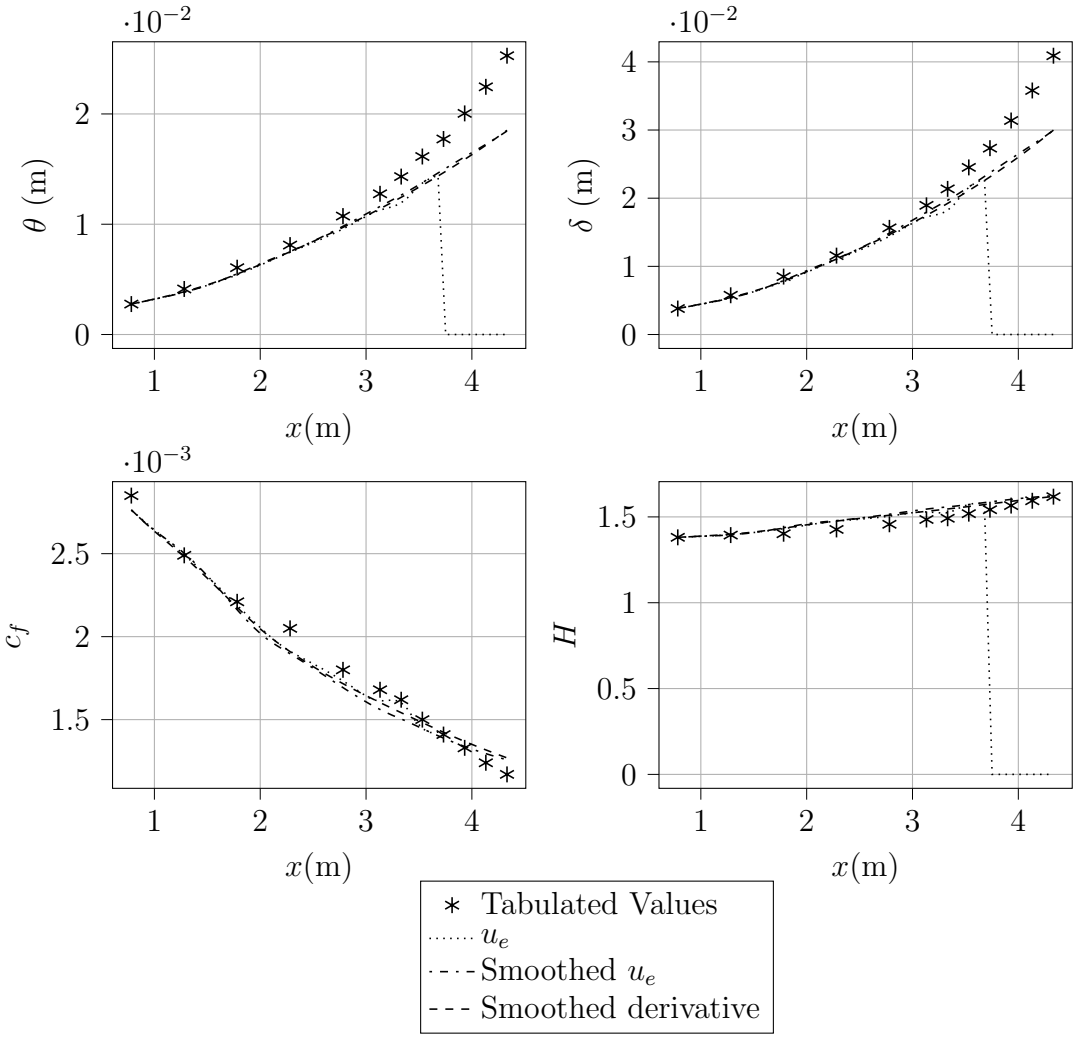

Figure 3.12: Flat plate with mild adverse pressure gradient, Head's Method. The simulation results using different velocity distrubutions, alongside tabulated experimental results.

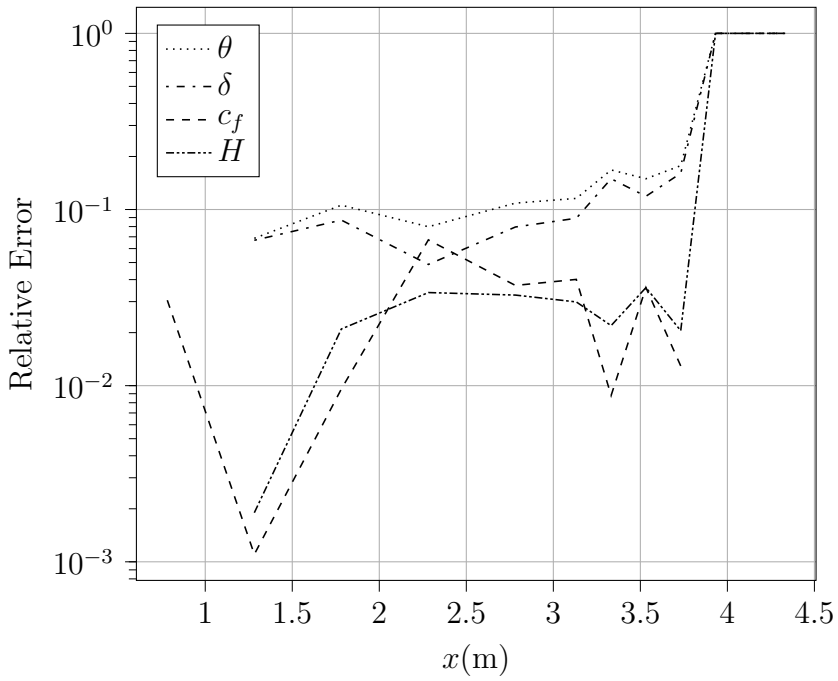

Figure 3.13: Head's method results for a flat plate, mild adverse pressure gradient. Error for each boundary layer parameter. Without smoothing implemented, the simulation failed near $x = 3.6$.

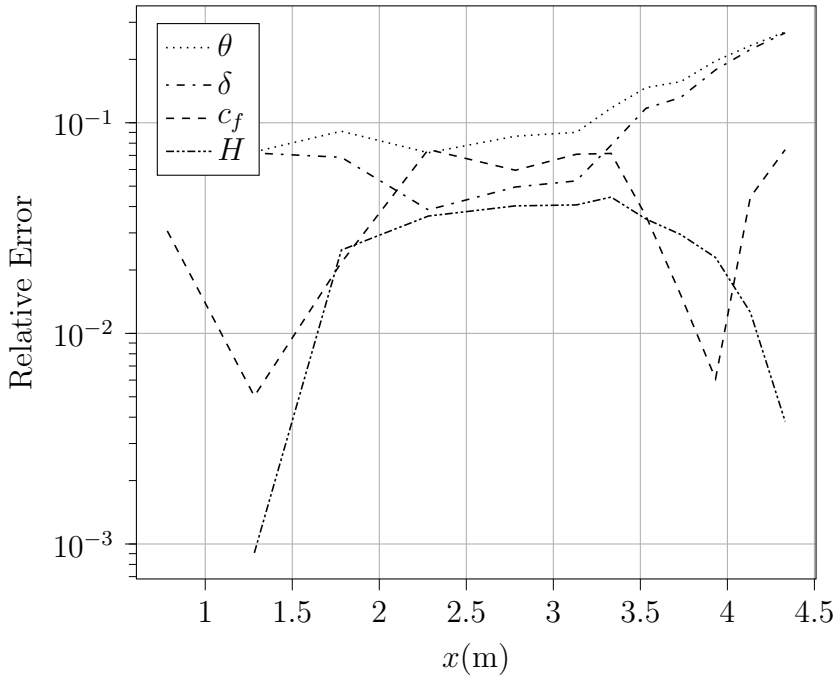

Figure 3.14: Head's method results for a flat plate, mild adverse pressure gradient with smoothed u_e data. Error for each boundary layer parameter.

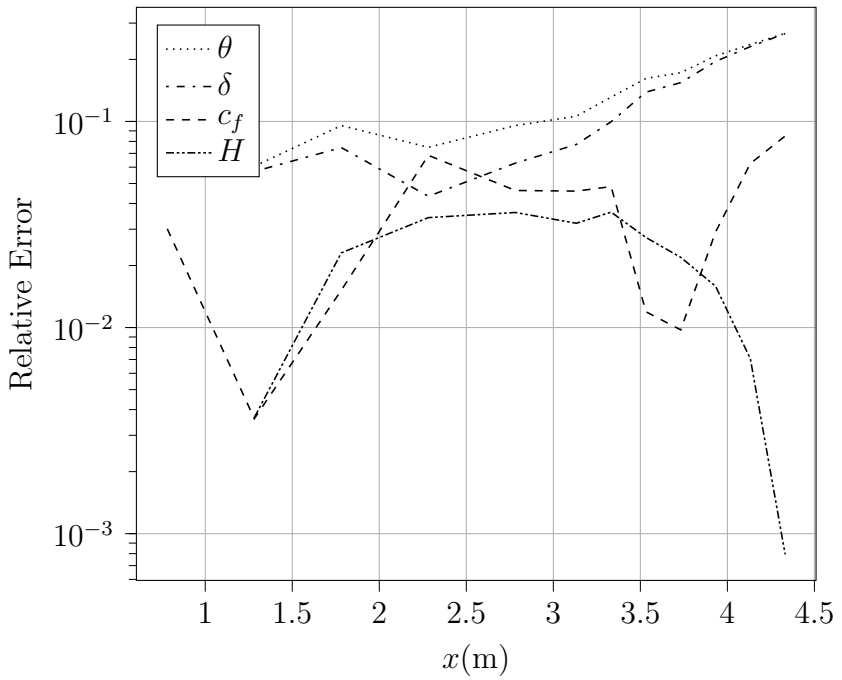

Figure 3.15: Head's method results for a flat plate, mild adverse pressure gradient using smoothed velocity derivative distribution. Error for each boundary layer parameter. The error improves slightly from the smoothed u_e result.

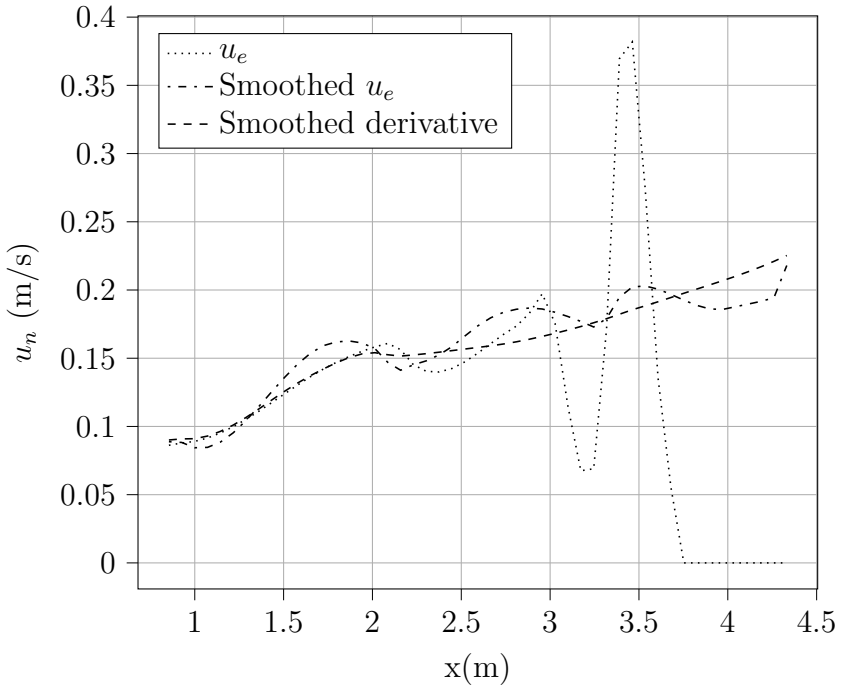

Figure 3.16: Flat Plate with mild adverse pressure gradient, Head's method - transpiration velocity results.

This case shows that in some instances, providing edge velocity data may result in non-physical results when $^{du_e}/_{dx}$ is not sufficiently smooth. When possible, smoothed data should be used.

This test shows the capabilities of this Head's method implementation to model turbulent boundary layers. When practical, smoothing of the edge velocity derivative seems to return more accurate results. For example, in a case where an inviscid solver is to be paired with Head's method, the transpiration velocity, u_n, would be very important in generating new panel strengths and running a new inviscid solution. Figure 3.16 shows calculated transpiration velocities from each simulation's results. Transpiration velocity may be calculated using the following relationship:

$$u_n = \frac{du_e}{dx}\theta H + u_e\theta\frac{dH}{dx} + u_e H\frac{d\theta}{dx} \tag{3.3}$$

Clearly, these results show that the smoothness of the velocity derivative has a significant impact on the resulting transpiration velocity. Smoothness of the data in a use case like this would be very important. However, in a case where the user is looking for the momentum thickness at a certain point on the surface, smoothness of the edge velocity profile or its derivative may not be an issue.

3.3 Testing Against XFOIL

In order to better demonstrate the results of the different methods incorporated into the module, the results from this work were compared to XFOIL against the current method. Note that since XFOIL tightly couples the Integral Boundary Layer solution to its panel strength solution iteration and uses a more sophisticated transition

criteria, these results will not match exactly[1]. This tight coupling normally involves several iterations to obtain the final edge velocity and boundary layer results. To minimize the effects of this tight coupling, care was taken to run only one viscous iteration of XFOIL for comparison of PyBL and XFOIL results. Even with this consideration, XFOIL's viscous method is admittedly different and equal results between these methods should not be expected, nor is XFOIL the gold standard for boundary layer results. Integral Boundary Layer methods stand in validity on their own [8], but XFOIL provides another tool to compare against.

XFOIL was used to create the inviscid flow over the airfoil to be analyzed. From this solution the edge velocities and stagnation point were found and used as the initial conditions for PyBL. XFOIL was then run in viscous mode, taking care to limit viscous iterations to one, to obtain the results needed for the comparisons. This process allowed the ability to control iterations of XFOIL's viscous solver and settings for transition method and location. For comparison against the fully turbulent solution with Head's method, this was important for forcing transition as early as possible.

In order to minimize the influence XFOIL's tightly coupled viscous solver has on its results, its number of iterations was set to one. However, this does not eliminate the effect of the boundary layer solution being closely coupled with its panel algorithm. This results in the data returned from XFOIL not being a true one-way coupling of the inviscid solution and the Integral Boundary Layer method as in this work. However, the comparisons are still instructive for examining the predicition capabilities of this work. Finally, it is recognized that the Michel transition criteria used for this work is not as capable as XFOIL's implementation of the e^N method; however, its inclusion in this work is presented here as a proof of concept. The introduction of a transition

[1]Since the Integral Boundary Layer solution affects the inviscid solution, via the transpiration velocity term, XFOIL uses an interative solution process to converge on the solution and solves for the panel strengths and Integral Boundary Layer terms simultaneously.

criteria parent class provides a building block for future authors to implement their own transition criteria using the spline and dense output capabilities of the module.

3.3.1 Laminar Results

In order to compare laminar results for as much chord length as possible, the laminar-only test case was chosen at a low Reynolds number, with an airfoil at 0° angle of attack. The airfoil chosen was a NACA 0003 airfoil, which is an airfoil with zero camber and maximum thickness of 3% of the chord length. This yielded an XFOIL prediction that did not transition or separate until nearly the trailing edge. The results are displayed in fig. 3.17. Figure 3.18 shows the associated differences against XFOIL.

This case did not trigger the module's transition criteria, but did trigger the separation criteria at $x = .956$. This chord location is where there is very high velocity derivative.

As seen most prominently in the displacement thickness and shape factor, an instability in the result arises near $x = 0.6$ and grows downstream. This fluctuation can be attributed to the non-smooth edge velocity profile in the XFOIL inviscid results at the same location, as shown in fig. 3.19. While this may not provide a large percent difference for velocity, the velocity derivative is significantly affected, also shown in fig. 3.19. This is another indication, along with the Head's method example, that smoothing of the edge velocity and its derivative should be investigated further.

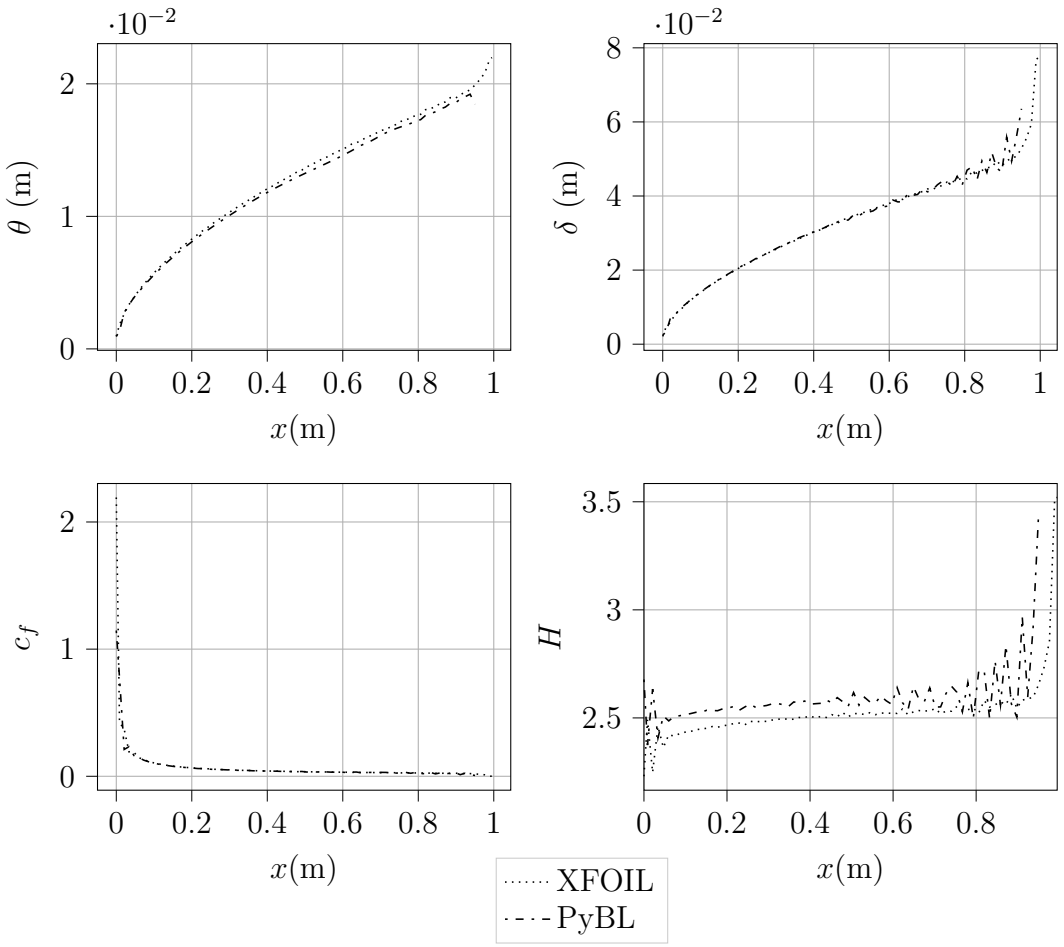

Figure 3.17: Comparison against the fully laminar PyBL result and XFOIL. PyBL reported laminar separation at $x = .956$.

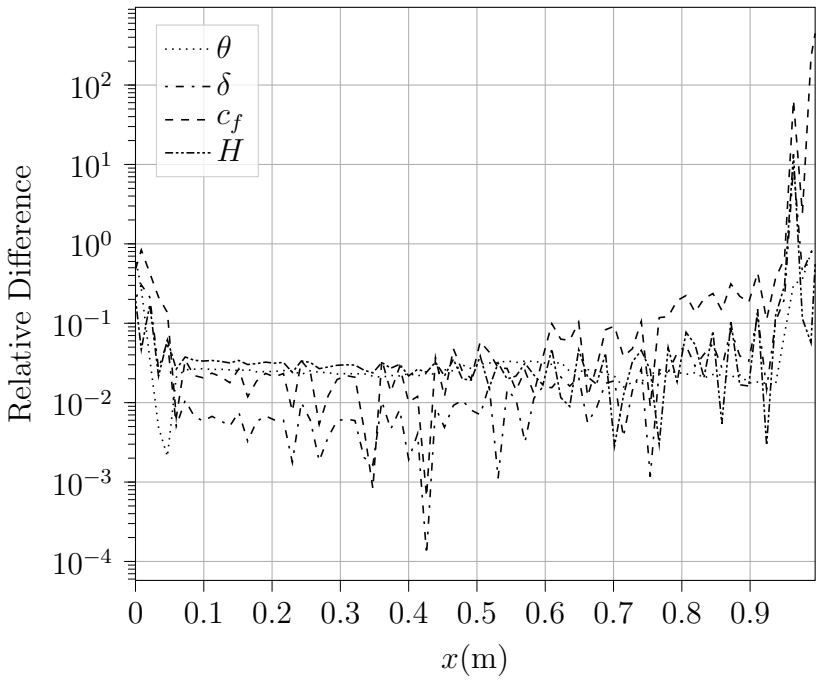

Figure 3.18: Difference between the fully laminar PyBL result and XFOIL. Note that PyBL reported separation at $x = .956$, so the large spike in difference near $x = 1$ is not guaranteed accuracy by the method.

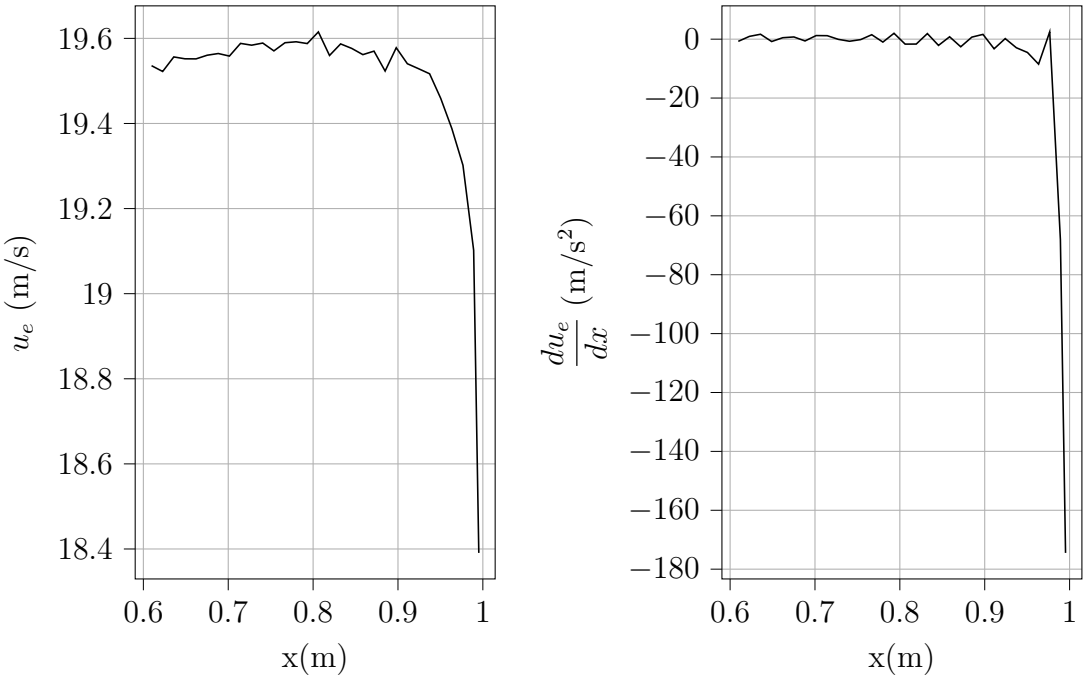

Figure 3.19: The velocity distribution from XFOIL provided difficulty in derivative approximation due to its jagged nature.

3.3.2 Turbulent Results

For the turbulent results, the choice of airfoil and conditions was less restrictive. In order to emulate a fully turbulent boundary layer, XFOIL was forced to transition near the leading edge. A value of $x = 0.01$ was used as the point at which to transition. Because $x = 0$ was the point chosen for initial conditions for the turbulent solver, this introduced a difference in the result at the leading edge, as seen in fig. 3.20 and fig. 3.21. Note that there is a discrepancy at the leading edge because of XFOIL's transition criteria, which smooths out the transition into a range of chord locations.

After this initial difference, the turbulent results show strong agreement with XFOIL for the rest of the domain, up until the trailing edge. For this case, the turbulent separation criteria was met by the Python module very close to the trailing edge at $x = 1.02$, due to the large edge velocity velocity derivative at that point. This value corresponds to the sharp increase in difference with XFOIL for all values at the trailing edge.

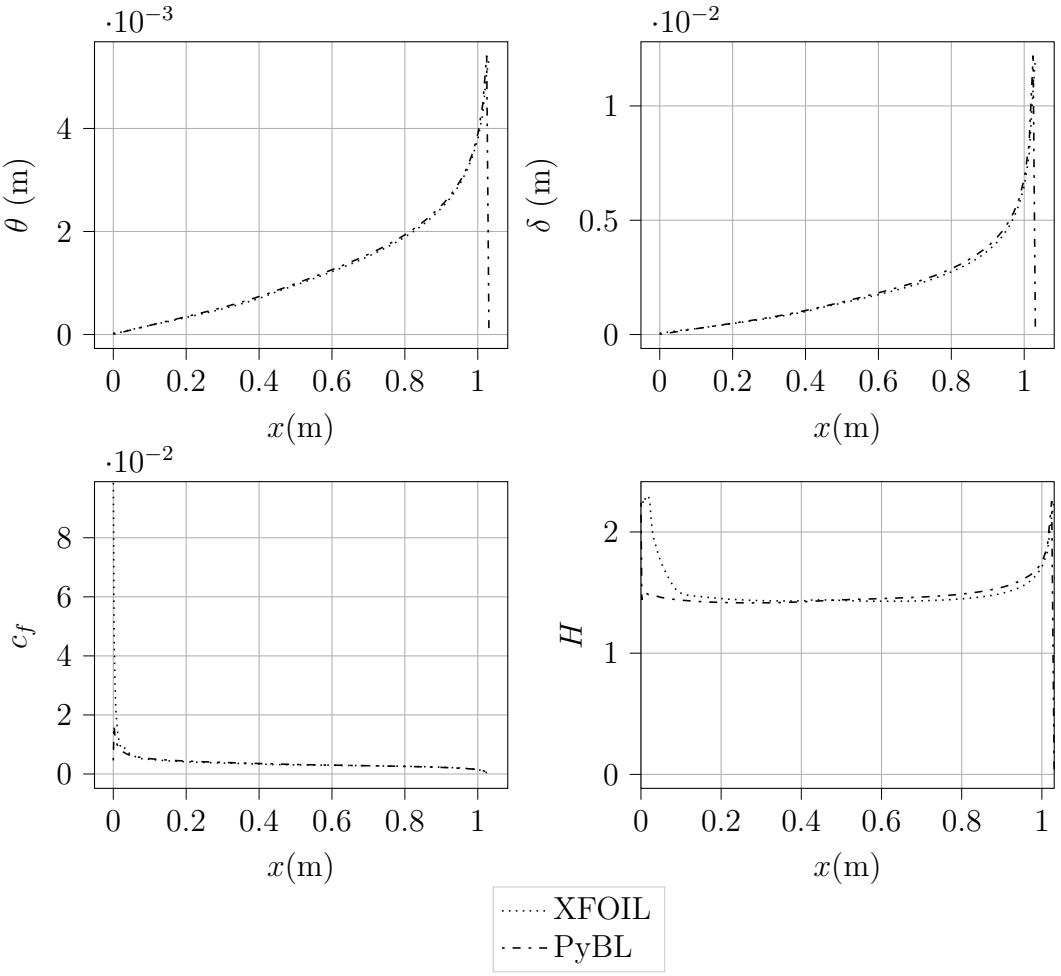

Figure 3.20: Results from fully turbulent XFOIL (forced transition at $x = .01$) and fully turbulent PyBL simulation (Head's Method only) using XFOIL's initial conditions. XFOIL's more gradual transition implementation bled remnants of laminar shape factor into the turbulent results.

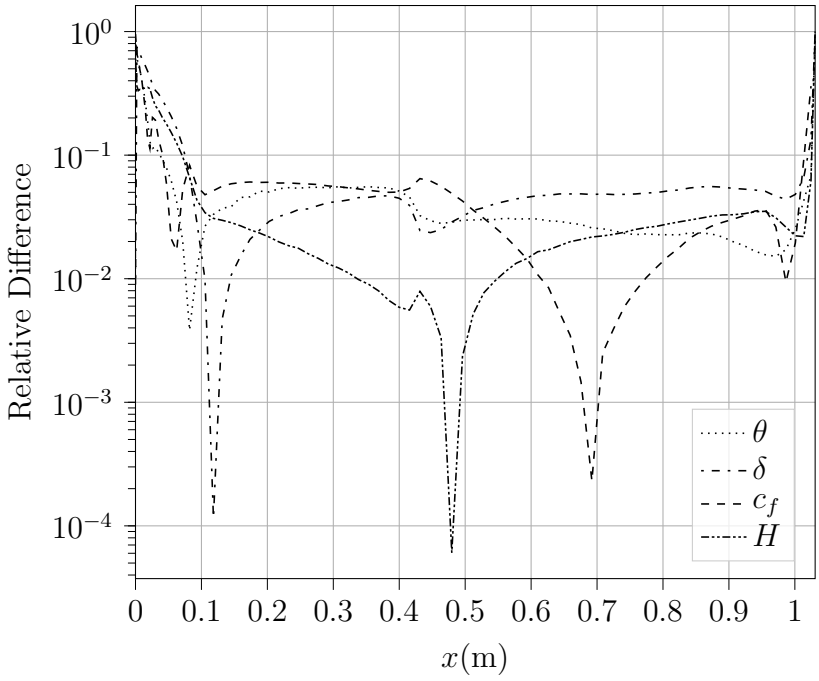

Figure 3.21: The differences of Head's method with respect to XFOIL results where transition was forced at $x = 0.01$.

3.3.3 Combined Results

This simulation was a comparison against XFOIL with a full implementation of stagnation criteria, the laminar Thwaites solver, transition criteria, the turbulent Head solver, and separation criteria for both the laminar and turbulent simulations. A different airfoil and flow conditions were tested in order demonstrate the versatility of the solver. The following result is a NACA 0009 airfoil, run at a Reynolds number of 2×10^6 and angle of attack of $0°$. The boundary layer parameters for this result are displayed in fig. 3.22. The Python module predicted transition at $x = 0.492$. This differed from the XFOIL prediction of $x = .66$. The results showed noticeable differences in this region, with the largest differences occurring at PyBL's transition location. The discrepancies propagated downstream, leaving the relative difference fairly high for this result. Figure 3.23 shows the differences for variables of interest for this simulation. While the differences are significant, this is a challenging test case

where errors will accumulate as they are propogated downstream. Note that the skin friction results matched once the XFOIL boundary layer transitioned. Figure 3.22 shows that the trend of the shape factor was not dissimilar to the XFOIL result, except for noticeable differences near transition.

Overall, there is much better matching between the XFOIL and PyBL results compared to the natural transition case. The laminar region matches quite well, and noticeable differences arise at the transition location because of XFOIL's treatment of the transition. The smoothing of the the transition region impacts the predicted properties at the transition location as well as downstream. However, this effect is much less pronounced than in the natural transition case.

Figure 3.25 shows the difference between XFOIL and PyBL where the differences in the laminar region are much less than 10%. The differences in displacement thickness, skin friction coefficient, and shape factor are highest at the transition location and then generally decrease downstream. For the momentum thickness, those differences do not spike at the transition location because both implementations keep the momentum thickness constant at the transition location because both implementations keep the momentum thickness constant at the the transition location. However, the XFOIL smoothing does impact the differences downstream as expected.

While XFOIL results are not to be accepted as truth, they provide valuable comparison for implementation of IBL methods.

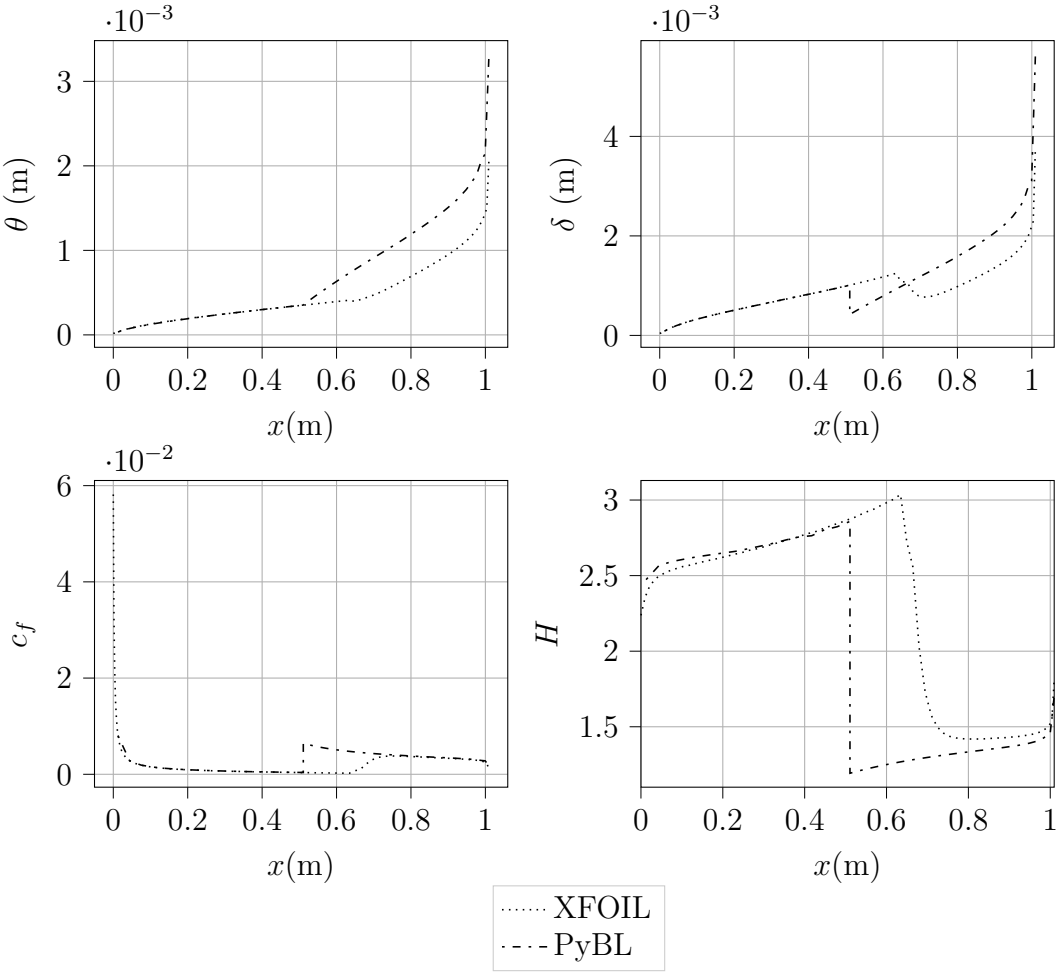

Figure 3.22: Results for full implementation of Michel transition criteria, laminar and turbulent solvers, against XFOIL. The curves showed similarities, even with transition location disagreement. While sufficient as a proof of concept for transition criteria, major differences existed between the two solutions.

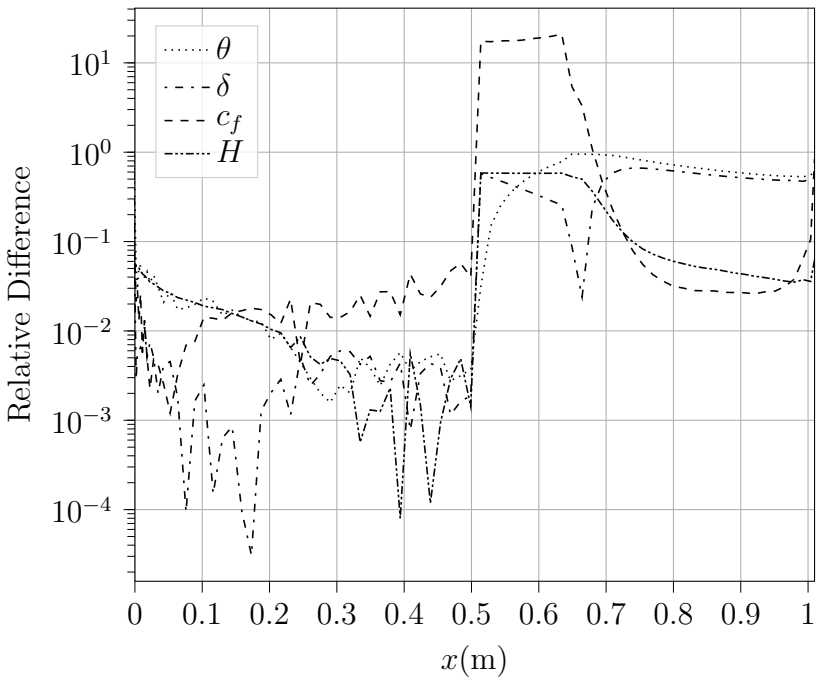

Figure 3.23: Full simulation result difference against XFOIL. Note that the largest difference occurs between the two solvers' estimates of x_{tr}. The H_{tr} value estimated by Michel did not represent the same gradual transition as XFOIL's e^N method.

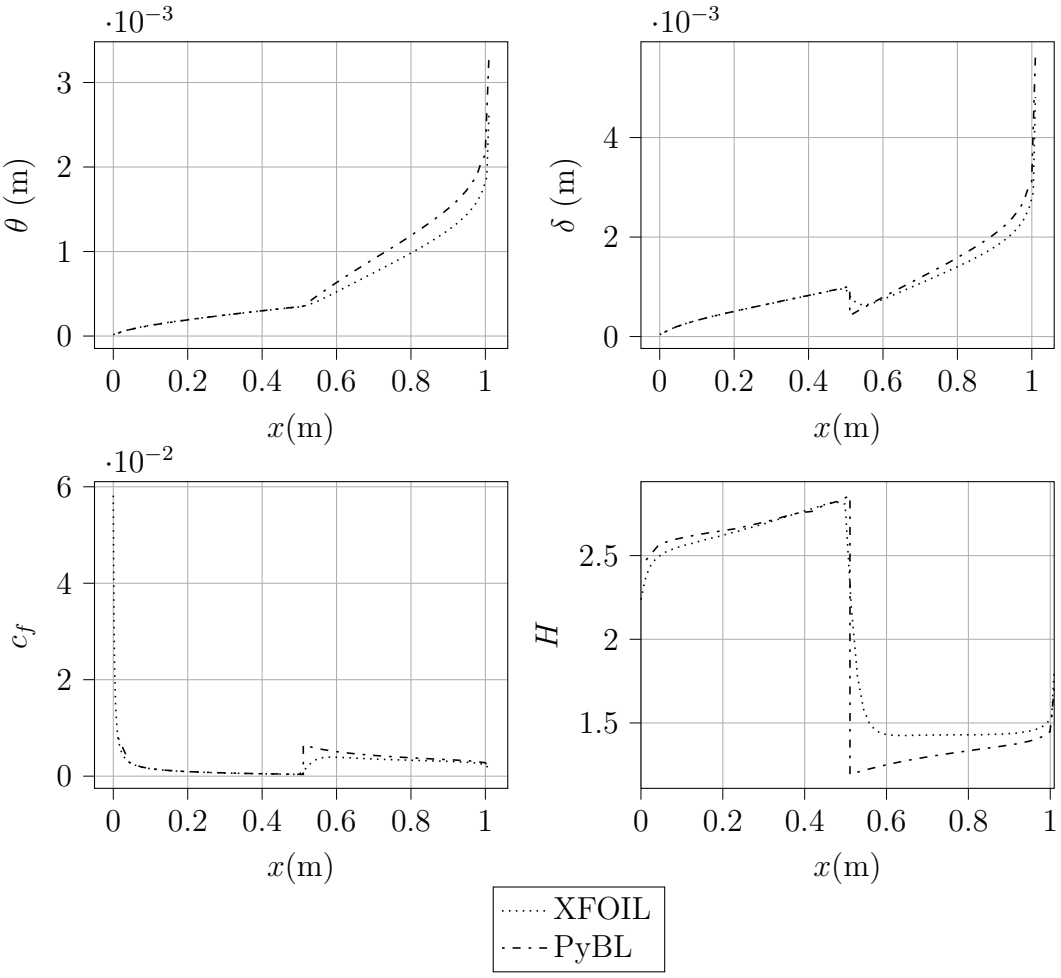

Figure 3.24: Results after forcing XFOIL to transition at the point predicted by Michel criteria. The instantaneous change in shape factor defined by the Michel method showed a clear difference from the gradual change in the XFOIL result.

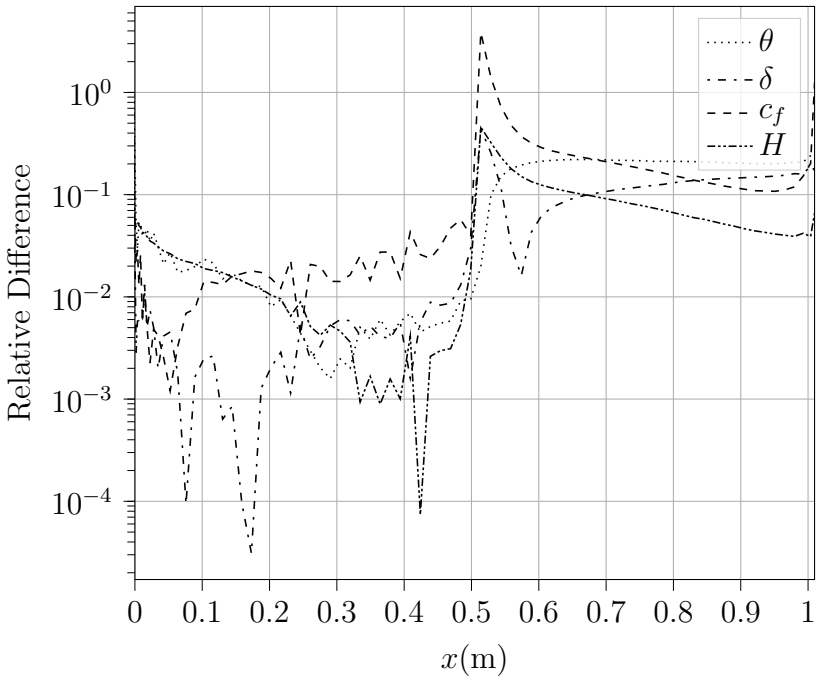

Figure 3.25: Simulation result difference after forcing XFOIL to transition at the same point as PyBL. A significant offset was still present in the results near transition.

Chapter 4

CONCLUSION AND FUTURE WORK

This project shed new light on implementations for Thwaites' and Head's methods. The results have shown to have sufficient agreement for various test cases, and were implemented such that a user could query an accurate solution value at any point along the surface. Cases run naively with only points from an inviscid velocity distribution matched very closely to other viscous simulations, and the dense output functionality resulted in a streamlined implementation of transition and separation criteria.

In terms of future work to be done, more accurate transition criteria should be implemented in order to produce more realistic results for transition and downstream. Further, a deeper change to Head's method (similar to removing the linear assumption of Thwaites' method for the s and H functions) could add to its fidelity. Better fits to the empirical relations of Head's method could improve the quality of the results. More complex integral boundary layer models can be implemented to improve the results and provide a wider selection of modeling options. This could include transformations for axisymmetric bodies or by integrating along streamlines for 3-dimensional geometries.

Not all features of the flow field may be well represented by the cubic spline method used in this work. For example, surfaces with abrupt steps or gaps may not be accurately represented by a smooth spline. This framework can be enhanced to encompass these cases. Finally, more work may be done to investigate the error introduced by using a cubic spline to interpolate within a 5-4 order Runge-Kutta

solver. It is possible that lower or higher order solvers may be a better fit, or that large step sizes decrease accuracy since portions of the spline may be stepped around entirely. More work should be done to ensure that this grouping of tools does not present inaccuracies by promoting instability and oscillations in the results.

As a general lesson from this project, the empirical and experimental fits and splines were less accurate in the extreme cases and reduced the quality of the results from the solver in those regions. This also affected the solution process with the ODE solver taking large spatial steps, and resulted in nonphysical values or leaving the domain for the fits. This typically resulted in poor quality solutions or the solutions diverging.

Finally, creating more comparisons against experimental data would greatly improve the definition of limits for these solvers and provide a valuable resource for additional models to be developed. Testing done in this work provided a good baseline comparison, but there are a wide range of cases to be covered that are of importance to aerodynamicists.

Revisiting these methods decades later has proven to be fruitful in refining their implementations. Only by continued effort will they reach their fullest adoption.

Bibliography

[1] H. Blasius. "Grenzschichten in Flussigkeiten mit kleiner Reibung, 2. angew". In: *Math. Phye* 56 (1908).

[2] V.M. Falkner and S.W. Skan. Aeronautical Research Council Report 1314. 1930.

[3] B. Thwaites. "Approximate Calculation of the Laminar Boundary Layer". In: *Aeronautical Quarterly* 1.3 (Nov. 1, 1949), pp. 245–280. ISSN: 0001-9259. DOI: 10.1017/S0001925900000184.

[4] H Blasius. *The Boundary Layers in Fluids with Little Friction*. NACA technical memorandum 1256. Washington, Feb. 1950, p. 58.

[5] Hubert Ludwieg and W. Tillmann. *Investigations of the Wall-Shearing Stress in Turbulent Boundary Layers*. Report 1285. Number: NACA-TM-1285. May 1950.

[6] M. R. Head. *Entrainment in the Turbulent Boundary Layer*. Aeronautical Research Council Report 3152. London: Her Majesty's Stationary Office: Ministry of Aviation, Sept. 1958.

[7] Hubert Ludwieg and W. Tillmann. "Ludwieg and Tillman, mild adverse pressure gradient (1100)". In: *Computation of turbulent boundary layers: 1968 AFOSR-IFP-Stanford Conference*. Ed. by S. J. Kline et al. Vol. II. Meeting Name: Conference on Computation of Turbulent Boundary Layers. [Stanford] Calif., Thermosciences Division, Stanford University, 1969, pp. 55–70.

[8] Tuncer Cebeci. *Momentum transfer in boundary layers*. In collab. with P. Bradshaw and Long Beach Faculty publications California State University. Series in thermal and fluids engineering. Washington: Hemisphere PubCorp, 1977. ISBN: 978-0-07-010300-9.

[9] J. R. Dormand and P. J. Prince. "A family of embedded Runge-Kutta formulae". In: *Journal of Computational and Applied Mathematics* 6.1 (Mar. 1, 1980), pp. 19–26. ISSN: 0377-0427. DOI: 10.1016/0771-050X(80)90013-3.

[10] R. Michel, D. Arnal, and E. Coustols. "Stability Calculations and Transition Criteria on Two- or Three-Dimensional Flows". In: *Laminar-Turbulent Transition*. Ed. by V. V. Kozlov. International Union of Theoretical and Applied Mechanics. Berlin, Heidelberg: Springer, 1985, pp. 455–461. ISBN: 978-3-642-82462-3. DOI: 10.1007/978-3-642-82462-3_55.

[11] Alec David Young. *Boundary Layers*. Google-Books-ID: _n1GAAAAYAAJ. American Institute of Aeronautics and Astronautics, 1989. 296 pp. ISBN: 978-0-930403-57-7.

[12] Stephen J. Pollard. "Evaluation of the CMARC panel code software suite for the development of a UAV aerodynamic model". Accepted: 2013-04-30T21:56:07Z. Thesis. Monterey, California. Naval Postgraduate School, 1997. URL: https://calhoun.nps.edu/handle/10945/31947 (visited on 08/16/2021).

[13] Giovanni P. Galdi. "An Introduction to the Navier-Stokes Initial-Boundary Value Problem". In: *Fundamental Directions in Mathematical Fluid Mechanics*. Ed. by Giovanni P. Galdi, John G. Heywood, and Rolf Rannacher. Advances in Mathematical Fluid Mechanics. Basel: Birkhäuser, 2000, pp. 1–70. ISBN: 978-3-0348-8424-2. DOI: 10.1007/978-3-0348-8424-2_1.

[14] Jack Moran. *An introduction to theoretical and computational aerodynamics*. Mineola, N.Y: Dover Publications, 2003. 464 pp. ISBN: 978-0-486-42879-6.

[15] John D. Anderson. "Ludwig Prandtl's Boundary Layer". In: *Physics Today* 58.12 (Dec. 1, 2005). Publisher: American Institute of Physics, pp. 42–48. ISSN: 0031-9228. DOI: 10.1063/1.2169443.

[16] Tuncer Cebeci and Jean Cousteix. *Modeling and computation of boundary-layer flows*. 2., rev. and extended ed. Berlin Heidelberg: Springer, 2005. 502 pp. ISBN: 978-0-9668461-9-5 978-3-540-24459-2.

[17] Frank White. *Viscous Fluid Flow*. 3rd edition. New York, NY: McGraw Hill, Jan. 1, 2011. ISBN: 978-1-259-00212-0.

[18] Hermann Schlichting and Klaus Gersten. *Boundary-Layer Theory*. 9th ed. Berlin Heidelberg: Springer-Verlag, 2017. ISBN: 978-3-662-52917-1. DOI: 10.1007/978-3-662-52919-5.

[19] Arthur E. P. Veldman. "Entrainment and boundary-layer separation: a modeling history". In: *Journal of Engineering Mathematics* 107.1 (Dec. 2017), pp. 5–17. ISSN: 0022-0833, 1573-2703. DOI: 10.1007/s10665-017-9930-x.

[20] Tuncer Cebeci and Jean Cousteix. *Modeling and Computation of Boundary-Layer Flows*. ISBN: 0-9668461-9-2.

[21] *scipy.integrate.RK45 — SciPy v1.7.0 Manual*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.RK45.html (visited on 07/02/2021).

[22] *scipy.interpolate.CubicSpline — SciPy v1.7.0 Manual*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.CubicSpline.html (visited on 07/07/2021).

APPENDICES

Appendix A

FALKNER-SKAN FLOW NEAR ZERO

While Falkner-Skan flow is a well-established laminar flow case, it provided difficulty for numeric implementation of Thwaites' method starting at zero due to its behavior near zero. For this reason, the results provided in this thesis avoid starting near zero, starting instead at $x = 0.05$. Depending on values of $m \in [0, 1]$, the derivative of θ^2 with respect to x can take a variety of values near zero. These values might be well-behaved analytically, but this term is often problematic numerically. For these cases, the assumptions of both the cubic spline derived from the velocity distribution and the Runge-Kutta solver leave the program with no ability to accurately represent the behavior of the initial conditions. For some cases, the slope is infinite, and the cubic spline cannot represent thst. This results in the polynomial fits being poorly suited for the solver.

At each time step, this Runge-Kutta implementation calculates λ:

$$\lambda = \frac{\theta^{2 du_e/dx}}{\nu} \qquad \text{(1.16 revisited)}$$

then solves for the derivative of θ^2 with respect to x:

$$\frac{d\theta^2}{dx} = \frac{2\nu(s - [2 + H\lambda)]}{u_e} \qquad \text{(A.1)}$$

At the first solver step, the values of u_e and θ (given that the exact value of θ from the Falkner-Skan solution is used) are exact values. Note that if $u_e = 0$, then this term will usually be ∞ (depending on the value of the numerator), and the ODE solver will not be able to proceed. Even if the derivative is finite for this case, special treatment would be needed for the numerical implementation to handle the indeterminate $0/0$ result. In addition, du_e/dx is a cubic spline estimate, and this makes the evaluation of λ problematic in some situations. The Falkner-Skan flow edge velocity can directly calculated as an input given:

$$u_e = u_\infty \left(\frac{x}{L}\right)^m \qquad \text{(1.12a revisited)}$$

The analytic edge velocity derivative, which the solver is only estimating, has the form:

$$\frac{du_e}{dx} = \frac{m u_\infty}{L} \left(\frac{x}{L}\right)^{m-1} \qquad \text{(A.2)}$$

At $x = 0$, u_e has the values:

$$u_e(0) = \begin{cases} u_\infty & \text{for } m = 0 \\ 0 & \text{for } 0 < m < 1 \\ 0 & \text{for } m = 1 \end{cases} \qquad \text{(A.3a)}$$

Immediately after $x = 0$, u_e has the values:

$$u_e(x > 0) = \begin{cases} u_\infty & \text{for } m = 0 \\ u_\infty \left(\frac{x}{L}\right)^m & \text{for } 0 < m < 1 \\ u_\infty \left(\frac{x}{L}\right) & \text{for } m = 1 \end{cases} \qquad \text{(A.3b)}$$

The value of ${}^{du_e}/_{dx}$ at $x = 0$ (found analytically) take the forms:

$$\frac{du_e}{dx}(0) = \begin{cases} 0 & \text{for } m = 0 \\ \infty & \text{for } 0 < m < 1 \\ \frac{u_\infty}{L} & \text{for } m = 1 \end{cases} \tag{A.3c}$$

Immediately after $x = 0$:

$$\frac{du_e}{dx}(x > 0) = \begin{cases} 0 & \text{for } m = 0 \\ \frac{mu_\infty}{L}\left(\frac{x}{L}\right)^{m-1} & \text{for } 0 < m < 1 \\ \frac{u_\infty}{L} & \text{for } m = 1 \end{cases} \tag{A.3d}$$

Examination of eq. (A.1) and eq. (A.3a)-eqs. (A.3a) to (A.3d) shows that for $x = 0$, ${}^{d\theta^2}/_{dx}$ is not finite for $m > 0$. Further, eq. (A.3c) and eq. (A.3d) show that ${}^{du_e}/_{dx}$ is infinite for some values of m at $x = 0$. For most cases, a cubic spline cannot accurately represent this function. For these reasons, the Falkner-Skan flows presented in this thesis begin at $x = 0.05$.