Evaluation of a Displacement-Body Model for Hypersonic Shock-Wave/Boundary-Layer Interaction

by

Lee Lopez

B.E. Mechanical Engineering
The Cooper Union for the Advancement of Science and Art, 2016

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2018

© 2018 Massachusetts Institute of Technology. All rights reserved.

Author: Signature redacted
Department of Aeronautics and Astronautics
May 24, 2018

Certified by: Signature redacted
Wesley L. Harris
C.S. Draper Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by: Signature redacted
Hamsa Balakrishnan
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
JUN 28 2018
LIBRARIES
ARCHIVES

MITLibraries

77 Massachusetts Avenue
Cambridge, MA 02139
http://libraries.mit.edu/ask

DISCLAIMER NOTICE

Due to the condition of the original material, there are unavoidable flaws in this reproduction. We have made every effort possible to provide you with the best copy available.

Thank you.

The images contained in this document are of the best quality available.

Evaluation of a Displacement-Body Model for Hypersonic Shock-Wave/Boundary-Layer Interaction

by

Lee Lopez

Submitted to the Department of Aeronautics and Astronautics
on May 24, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

In this thesis, a displacement-body model for strong-interaction shock-wave/boundary-layer interaction (SBLI) is presented and evaluated. The model considers 2-D flow over a flat plate with an adiabatic wall. The separation bubble is modeled as a displacement body with constant surface pressure, the value of which is set equal to the value of plateau pressure given by free-interaction theory. A shock-fitting method of characteristics is employed to numerically compute quantities in the inviscid outer flow. Boundary conditions that satisfy physical requirements at shock waves, slip lines, and solid walls are enforced. Accuracy of the model is shown for both laminar and turbulent flow regimes, as well as for Mach numbers in the hypersonic regime. Additionally, the model provides a physical explanation for the pressure drop observed downstream of reattachment in hypersonic flows.

Thesis Supervisor: Wesley L. Harris
Title: C.S. Draper Professor of Aeronautics and Astronautics

Acknowledgements

I would first like to extend my gratitude to my advisor, Professor Wesley L. Harris. Professor Harris has been a constant source of both academic and personal guidance for me over these past two years. I thank him for the freedom that he gave me to work on a project that interests me, and for his valuable input, which helped me to see that project through.

I would also like to thank all of my professors, for teaching me both the tools and the mindset to tackle new problems.

Last but not least, I would like to thank my family for their love and support over the years. I could not be where I am today without them.

Contents

List of Figures

List of Tables

Nomenclature

Symbols

β	Shock angle measured from x-axis
γ	Ratio of specific heats
δ	Boundary layer 99% thickness
δ^*	Boundary layer displacement thickness
θ	Flow angle measured from x-axis (i.e., $tan^{-1}\left(\frac{v}{u}\right)$)
μ	Mach angle (i.e., $sin^{-1}\left(\frac{1}{M}\right)$)
ρ	Density
τ	Viscous shear stress (i.e., τ_{xy})
ν	Prandtl-Meyer function
ϕ	A general flow quantity
C_f	Skin friction coefficient
c_p	Specific heat at constant pressure
k, l	Iteration resolution parameters
M	Mach number
N_A, N_B	Grid resolution parameters
p	Pressure
q	Incompressible dynamic pressure (i.e., $\frac{1}{2}\rho V^2$)
R	Gas constant
Re	Reynolds number (subscript denotes length scale)
s	Specific entropy
T	Temperature
u	x-component of velocity
v	y-component of velocity
V	Magnitude of $\boldsymbol{V}$
$\boldsymbol{V}$	Velocity vector (i.e., $u\hat{\boldsymbol{x}} + v\hat{\boldsymbol{y}}$)
x	Coordinate along wall
$\hat{\boldsymbol{x}}$	Unit normal vector along wall
y	Coordinate normal to wall
$\hat{\boldsymbol{y}}$	Unit normal vector normal to wall

Special Non-dimensional Parameters

$$\bar{\tau} = \frac{\tau}{\tau_{w0}}$$

$$\bar{x} = \frac{x - x_0}{x_s - x_0}$$

$$\tilde{x} = \frac{x - x_s}{x_r - x_s}$$

$$\bar{y} = \frac{y}{\delta_0^*}$$

$$\tilde{y} = \frac{y - y_w}{x_r - x_s}$$

Subscripts

0	Evaluated upstream of the interaction region
1	Evaluated immediately upstream of a shock wave
2	Evaluated immediately downstream of a shock wave
f	Evaluated immediately downstream of expansion fan past reattachment point
incident	Evaluated for incident shock wave
p	Plateau value
r	Evaluated at reattachment point
s	Evaluated at separation point
w	Evaluated at wall

Chapter 1

Introduction

1.1 Background

Shock-wave/boundary-layer interaction (SBLI) is a fundamental feature of fluid flow in the supersonic and hypersonic regimes. SBLI is ubiquitous in supersonic internal flows. Supersonic and hypersonic speeds admit the presence of shock waves, and since internal flows are contained within an enclosure, boundary layers form on the enclosure surfaces. Thus, any shock waves that form and extend to the enclosure wall interact with the boundary layer at those wall locations.

In external flow, SBLI is perhaps most commonly encountered in transonic flight. In such a situation, subsonic flow is accelerated above $M = 1$ as it traverses an airfoil, and a normal shock wave develops in order to compress flow back to the subsonic regime. The normal shock wave penetrates into the boundary layer that forms on the airfoil surface, imposing a dramatic adverse pressure gradient that may cause boundary layer separation. In many transonic aircraft, wing sweep is employed in order to delay the onset of shock waves and mitigate the negative effect that SBLI can have on the aerodynamic efficiency of the aircraft.

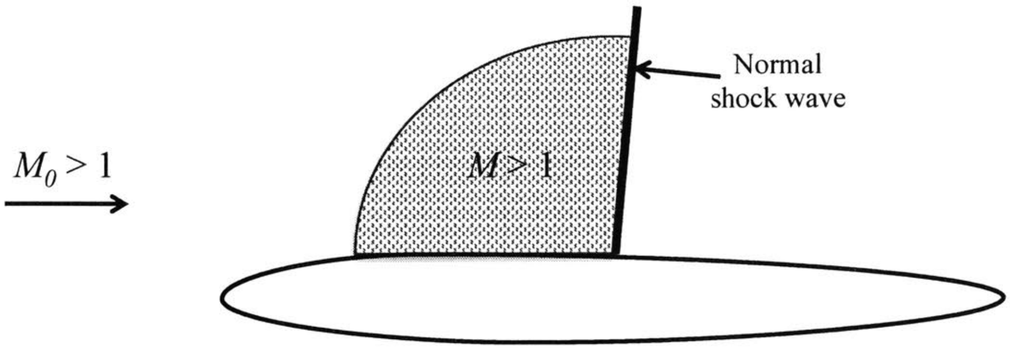

Figure 1-1: Flow regimes around a transonic airfoil, and the normal shock wave that forms and interacts with the boundary layer.

Meanwhile, in supersonic flight, the generation of shock waves is unavoidable. In supersonic aircraft configurations, SBLI occurs wherever a shock wave is incident upon another surface of the aircraft. SBLI also occurs at compression corners, where an oblique shock wave forms and interacts with the boundary layer on the compression corner surface. SBLI can have dramatic consequences on aerodynamic performance (i.e., aerodynamic efficiency), and thermodynamic heating. The latter is particularly a problem at hypersonic speeds, where exceedingly high temperatures can be observed when a shock-induced separation bubble reattaches. SBLI is therefore a crucial aspect of supersonic and hypersonic flight design.

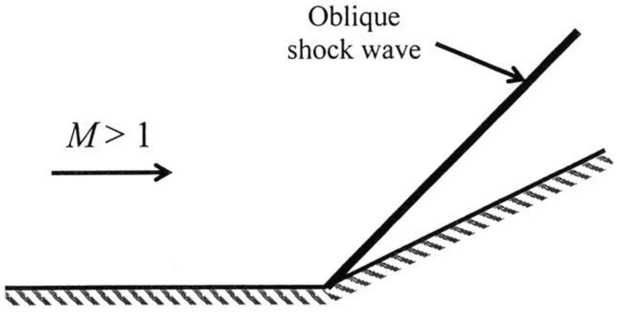

Figure 1-2: The oblique shock wave that forms at a supersonic compression corner.

SBLI can be classified into weak interactions and strong interactions. A weak interaction is characterized by minimal influence of the viscous boundary layer on the outer inviscid flow, and vice versa. The overall shock structure in a weak interaction resembles that of a purely inviscid shock wave reflection off a wall. However, the discontinuous rise in surface pressure that occurs in a purely inviscid case (an approach in which the boundary layer is neglected) is not observed with the presence of a boundary layer. Instead, the subsonic channel of the boundary layer allows pressure waves to propagate upstream of the purely inviscid reflection point, resulting in a more gradual rise in surface pressure due to the impinging shock wave and a region of upstream influence which cannot be predicted by a purely inviscid approach.

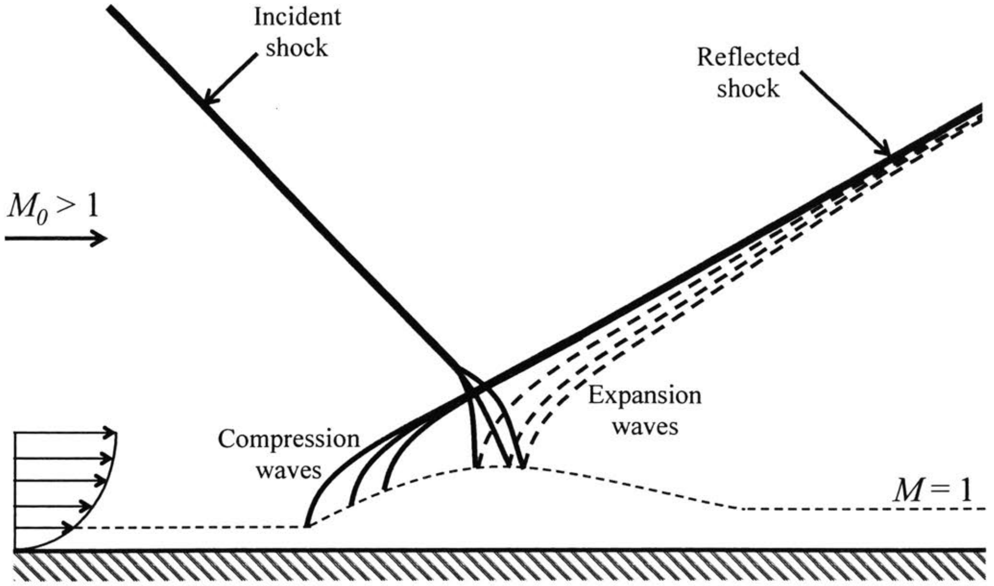

Figure 1-3: Weak interaction between an oblique shock wave and a boundary layer. The boundary layer thickens upstream of the incident shock wave, due to upstream influence below the sonic line. Outside the boundary layer, the two-shock structure resembles an inviscid shock wave reflection off a wall.

If the impinging shock wave imposes a sufficiently strong adverse pressure gradient upon the boundary layer, the boundary layer will separate. When separation occurs, the interaction is said to be a strong interaction. Such a strong interaction is the focus of this thesis.

In the case of a strong-interaction SBLI, there is a rapid thickening of the boundary layer in the separation region. Thickening of the boundary layer results in the formation of compression waves, which extend into the outer flow and coalesce to form a separation shock wave. The separation shock wave interacts with the incident shock wave, and from the intersection point of the two shock waves originates a slip line, or vortex sheet, across which pressure and flow direction are continuous but other flow properties (such as M

and s) are not. Upon reaching the boundary layer, the incident shock wave refracts, resulting in a system of reflected expansion waves that extends back into the outer flow [1]. Further downstream, the boundary layer reattaches, and a series of compression waves coalesce to form the reattachment shock wave.

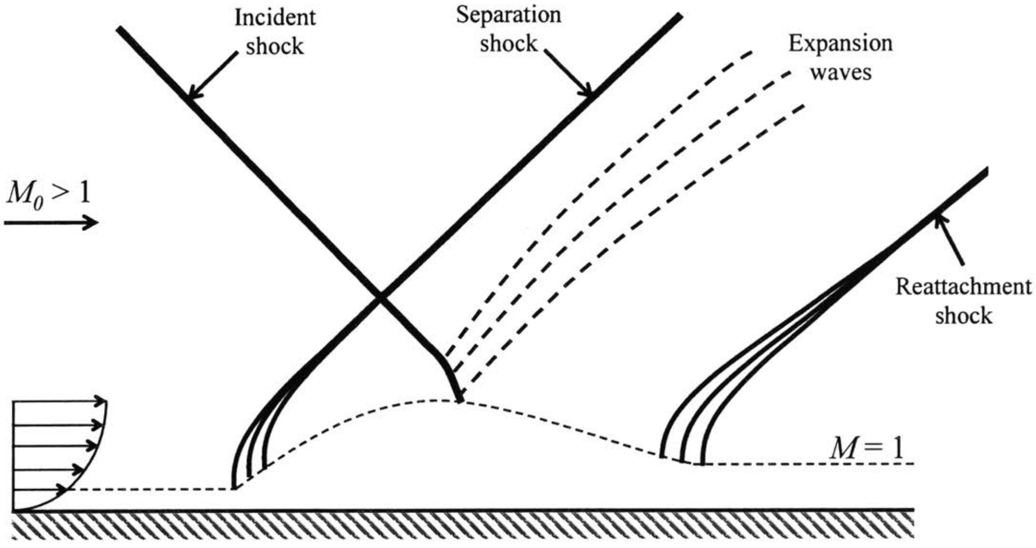

Figure 1-4: Strong interaction between an oblique shock wave and a boundary layer. In contrast to the weak interaction, there is a clear three-shock structure outside the boundary layer consisting of: incident shock wave, separation shock wave, and reattachment shock wave.

1.2 Prior Work

Early investigations of SBLI focused on laminar boundary layers, and were primarily theoretical or experimental in nature. Among the earliest theoretical contributions to the study of SBLI were those of Lees [2], in 1949, and Chapman et al. [3], in 1958. The development of free-interaction theory by Chapman et al. was particularly significant, and the theory still forms the basis for developing analytical intuition into SBLI. In the 1950s, experimental investigations of SBLI with laminar boundary layers were performed [4, 5], and they confirmed the free-interaction theory proposed by Chapman et al.

Computational approaches to SBLI followed a similar trajectory, beginning with a focus on laminar boundary layers. In 1964, Lees and Reeves proposed a boundary layer integral method for SBLI, which showed excellent agreement with experimental data for adiabatic, laminar flows [6]. More recent endeavors in computational fluid dynamics (CFD) have shifted to focus on turbulent boundary layers, the dynamics of which under the influence of shock waves are still largely not well understood [7, 8]. These endeavors include recent work in large-eddy simulation (LES) and direct numerical simulation (DNS), as well as a recent shift from the more traditional finite volume method to the finite element method.

1.3 Motivation for a Simplified Model

In the literature, there is a lack of efficient computational tools for predicting the characteristics of turbulent SBLI. Such a tool would be particularly useful for the validation of newer CFD approaches, which are typically validated against either experimental data or DNS results. The limited availability of existing data, and the long turnaround time for new experiments or DNS, limits the range of flow conditions and configurations over which new CFD results can be validated. An efficient means of checking CFD results over a range of arbitrary flow conditions is therefore quite attractive.

A simple yet accurate computational tool for SBLI also has the potential to be a time-saving design tool. A designer with access to such a tool might be encouraged to forego a more time-consuming simulation or experiment altogether. There is a particular case to be made when the quantities of interest to the designer are largely determined by inviscid mechanisms. It should not be necessary perform an expensive simulation or an experiment if, for example, a designer is only interested in the surface pressure associated with SBLI.

1.4 Thesis Outline

This thesis endeavors to provide that efficient computational tool for strong-interaction SBLI, covering both laminar and turbulent boundary layers. In the following sections, a two-dimensional, inviscid model of an oblique shock wave impinging on a flat-plate boundary layer is presented. First, the physical model and the computational details of performing calculations with the model are developed. Later, numerical results are presented and compared with experimental data. Finally, concluding remarks on the model are offered, along with possibilities for future work to address the model's limitations.

Chapter 2

Development of Model

2.1 Displacement-Body Model for the Boundary Layer

2.1.1 Global Shock Structure

The global shock structure of a strong-interaction SBLI (see Figure 1-4) suggests a simplified inviscid model, in which a displacement body produces an outer flow similar to the one that results from the viscous boundary layer [9]. In this thesis the displacement body takes the shape of a two-sided ramp. From the perspective of the outer inviscid flow, this geometry results sequentially in a compression, an expansion, and finally a second compression.

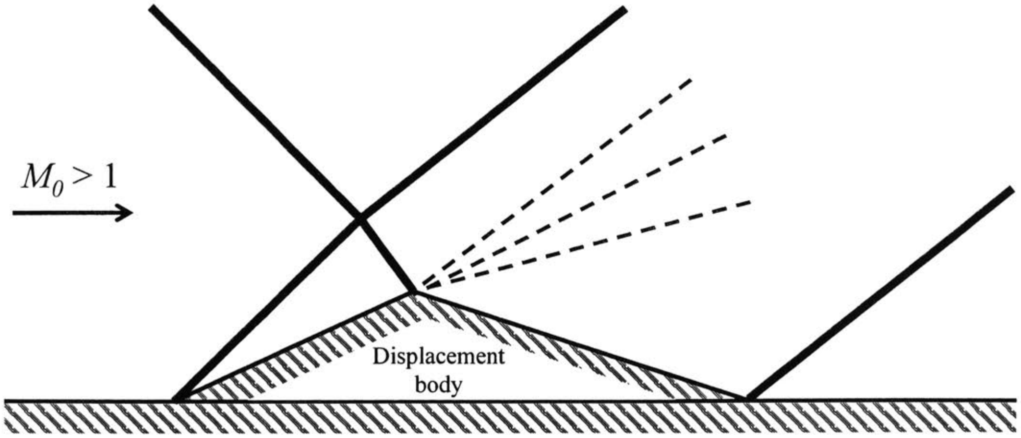

Figure 2-1: The displacement-body model of a strong-interaction SBLI. The outer flow structure shares many features with the viscous case (see Figure 1-4).

The requirements that are imposed on the displacement body are that: (a) the incident shock wave impinges on the apex of the ramp, and (b) the pressure distribution along the surface of the displacement body matches the surface pressure distribution inside a shock-induced separation bubble. Requirement (a) eliminates the possibility of a reflection of the incident shock wave off the displacement body, and also fixes the position of the displacement body relative to the incident shock wave. Requirement (b) fixes the shape of the displacement body by setting the strength of the first compression and that of the expansion. In order to impose (b), information of the surface pressure distribution inside a shock-induced separation bubble is required. This information is obtained from the insights of free-interaction theory, which is summarized below.

2.1.2 Free-Interaction Theory

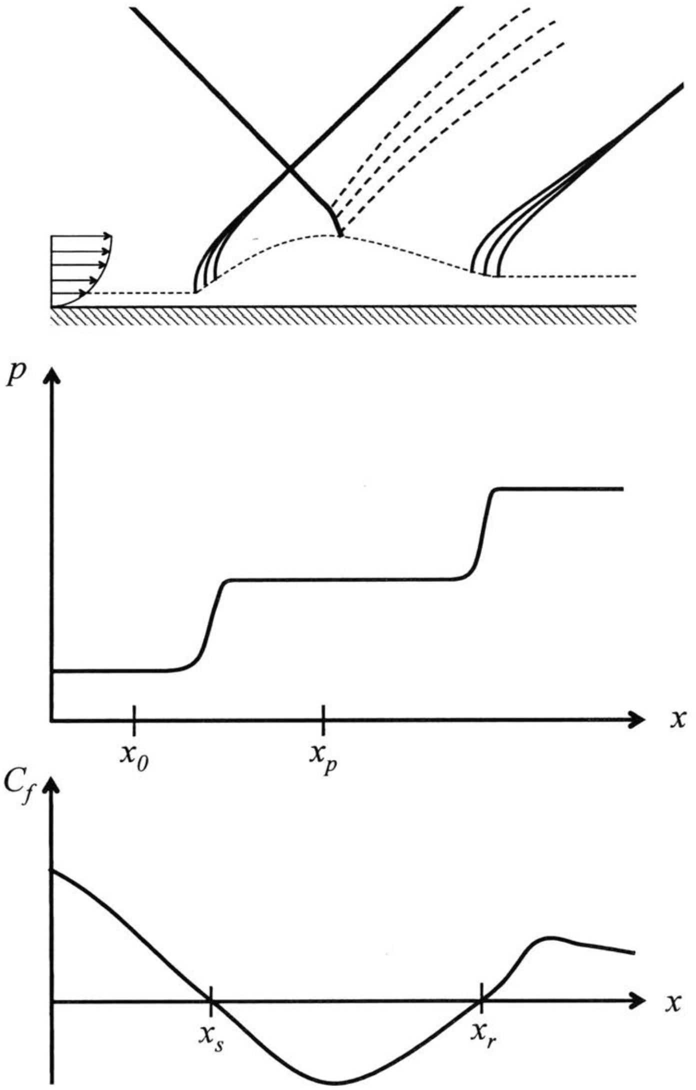

Figure 2-2: Distributions of surface pressure and skin friction coefficient for a strong-interaction SBLI. Several streamwise locations of interest in free-interaction theory are labeled.

A theoretical description of two-dimensional, strong-interaction SBLI is provided by free-interaction theory, first proposed by Chapman et al. [3]. The relevant equations for developing the theory are the streamwise boundary layer equation evaluated at the wall, and the equation for wave deflection of a supersonic flow due to the thickening of the boundary layer:

$$\frac{dp}{dx} = \left(\frac{\partial \tau}{\partial y}\right)_w \tag{2.1}$$

$$\frac{p - p_0}{q_0} = \frac{2}{\sqrt{M_0^2 - 1}} \frac{d\delta^*}{dx} \tag{2.2}$$

Note that (2.2) is linearized about conditions upstream of the interaction region, and therefore it is implicit that perturbations in the linearized parameters (p, q, and M) are assumed to be small.

Non-dimensionalization of the parameters in (2.1) and (2.2), an x-wise integration of (2.1) starting from x_0, and the combination of the result of that integration with (2.2), yields the following relationship:

$$\frac{p - p_0}{q_0} = F(\bar{x}) \sqrt{\frac{2C_{f_0}}{\left(M_0^2 - 1\right)^{1/2}}} \tag{2.3}$$

The function $F(\bar{x})$ is defined as:

$$F(\bar{x}) = \sqrt{\frac{d\bar{\delta}^*}{d\bar{x}} * \int_{\bar{x}_0}^{\bar{x}} \left(\frac{\partial \bar{\tau}}{\partial \bar{y}}\right)_w d\bar{x}} \tag{2.4}$$

For an adiabatic wall, $F(\bar{x})$ is a universal correlation function that monotonically increases from location x_0 through separation, before leveling off to a plateau value inside the separation bubble. These features of $F(\bar{x})$ for strong-interaction SBLI have been confirmed by experiment [10], and the separation and plateau values of $F(\bar{x})$ for laminar and turbulent boundary layers are stated in Table 2.1.

Table 2.1: Experimentally determined values [10] of the universal correlation function $F(\bar{x})$ at selected streamwise locations, for laminar and turbulent boundary layers.

	$F(\bar{x}_s)$	$F(\bar{x}_p)$
Laminar	0.81	1.47
Turbulent	4.22	6.00

An alternative correlation [11] for predicting the plateau pressure for a turbulent boundary layer, which eliminates the dependence on *Re* through skin friction in (2.3), is:

$$\frac{p_p}{p_0} = 1 + 0.5M_0 \tag{2.5}$$

Although elegant, the results of free-interaction theory are limited in their applicability. Free-interaction theory relies on the assumption that, compared to momentum, viscosity plays a relatively important role in the overall boundary layer dynamics of strong-interaction SBLI. This assumption is generally valid for laminar boundary layers and for turbulent boundary layers at low to moderate Re_δ, but begins to break down for turbulent boundary layers above $Re_\delta \sim 10^5$. This breakdown of free-interaction theory is attributed to the thinning of the viscous sublayer compared to the thickness of the subsonic layer at larger Re_δ, and therefore a lessening influence of viscosity in the overall boundary layer dynamics [9]. For very large Re_δ, an inviscid treatment of the rotational boundary layer is more appropriate than the viscous approach of free-interaction theory.

Nonetheless, for laminar boundary layers and turbulent boundary layers below $Re_\delta \sim 10^5$, free-interaction theory reveals important features of strong-interaction SBLI. For example, examination of (2.3) suggests that the surface pressure distribution within the separation bubble is determined entirely by the initial flow conditions upstream of the interaction. It is not dependent on the impinging shock wave in any way, provided that the impinging shock wave is strong enough to induce separation. Alternative combinations of (2.1) and (2.2) demonstrate that other features of strong-interaction SBLI are also dependent only on upstream conditions. If, for example, (2.1) and (2.2) are combined to eliminate p, the length scale of upstream influence (i.e., $x_s - x_0$) for a strong interaction can be similarly shown to be independent of the impinging shock wave strength.

Despite its many insights, free-interaction theory does not provide information about the reattachment region. In fact, reattachment is heavily dependent on the impinging shock wave strength, and so are quantities associated with reattachment, such as the overall length scale of the separation bubble and the pressure rise at reattachment. An increase in impinging shock wave strength will cause a larger separation bubble to form, since more momentum must be transferred from the outer flow down through the boundary layer to overcome the reversed near-wall flow [12]. A stronger impinging shock wave will also cause a larger rise in surface pressure to occur at reattachment. The reason for this dependence is that the total pressure rise over the interaction region is dictated by the strength of the impinging shock wave, but the pressure rise at separation is limited by the initial flow conditions according to free-interaction theory. Therefore, the difference between the total pressure rise, as necessitated by the impinging shock wave, and the pressure rise up to the plateau value, limited by initial flow conditions, must occur at reattachment.

Free-interaction theory provides the missing piece of information needed to fully develop the inviscid model of 2-D, strong-interaction SBLI that is treated in this thesis. The existence of a plateau pressure within the separation bubble suggests that the two-sided ramp in this study have a constant surface pressure equal to the plateau pressure. Equation (2.3) combined with the results of Erdos & Pallone [10] is used to set the plateau pressure for a laminar boundary layer, and equation (2.5) is used for a turbulent boundary layer. For given upstream flow conditions and a given incident shock wave, the setting of the surface pressure on the displacement body surface to a constant value determines a unique geometry for the displacement body.

In applying the results of free-interaction theory, it is assumed that the relevant assumptions hold, namely that if the boundary layer is turbulent, it be limited to a low to moderate Re_δ. Furthermore, in assuming that the surface pressure is constant and equal to the plateau pressure value for the entire streamwise extent of the separation bubble, an additional constraint on the validity of the model is introduced. The assumption of a constant surface pressure distribution within the separation bubble is most accurate when the separation bubble is large, i.e., $\frac{x_r - x_s}{x_s - x_0} \gg 1$. Since the length scale of the separation bubble is correlated positively with impinging shock wave strength, the isobaric two-ramp model most closely resembles a shock-induced separation bubble for an impinging shock wave strength far greater than that required for insipient separation.

2.2 Model of Fluid

For numerical calculations, the fluid is taken to be air with $c_p = 1.005$ kJ/kg/K and $R = 287$ J/kg/K. The air is assumed to be a calorically perfect gas; thus, specific heats are assumed to be constant, and the state of the gas is described by the ideal gas equation:

$$p = \rho RT \tag{2.6}$$

Furthermore, the gas is assumed to be in thermodynamic and chemical equilibrium always. Neither dissociation nor ionization of air molecules is modeled. Rarefied gas effects are also neglected; the gas is treated as a continuum.

2.3 Governing Equations

The laws governing fluid motion are conservation of mass, momentum, and energy. For steady, adiabatic flow of an inviscid fluid in the absence of body forces, the conservation laws may be written as:

$$\nabla \cdot (\rho V) = 0 \tag{2.7}$$

$$\nabla\cdot(\rho VV) = -\nabla p \tag{2.8}$$

$$\nabla\cdot\left(\rho V\frac{p}{\rho^{\gamma}}\right) = 0 \tag{2.9}$$

The boundary conditions for (2.7)-(2.9) are the user-specified freestream flow quantities and the no-flux condition, which is applied at solid walls. The no-flux boundary condition is equivalent to enforcing that the flow direction at a solid wall be tangent to the wall.

While (2.7) and (2.8) are conventional statements of conservation of mass and momentum, respectively, special attention is drawn to (2.9). Equation (2.9) is a statement of conservation of entropy, which replaces the conservation of energy equation. For the displacement-body model, it is valid throughout the flow except across a shock wave, where entropy is generated.

Since (2.9) cannot be applied across a shock wave, the conditions just upstream and just downstream of a shock wave must be related by another method. Applying conservation of mass, momentum, and energy across an oblique shock wave yields the following relationships between the upstream and downstream conditions:

$$tan(\theta_2 - \theta_1) = 2\,cot(\beta - \theta_1)\,\frac{M_1^2\,sin^2(\beta - \theta_1) - 1}{M_1^2(\gamma + cos[\,2\{\beta - \theta_1\}\,]) + 2} \tag{2.10}$$

$$M_2^2\, sin^2(\beta-\theta_2) = \frac{M_1^2\, sin^2(\beta-\theta_1) + \frac{2}{\gamma-1}}{\frac{2\gamma}{\gamma-1} M_1^2\, sin^2(\beta-\theta_1) - 1} \tag{2.11}$$

$$\frac{p_2}{p_1} = 1 + \frac{2\gamma}{\gamma+1}\left(M_1^2\, sin^2(\beta-\theta_1) - 1\right) \tag{2.12}$$

$$\frac{\rho_2}{\rho_1} = \frac{(\gamma+1)\, M_1^2\, sin^2(\beta-\theta_1)}{(\gamma-1)\, M_1^2\, sin^2(\beta-\theta_1) + 2} \tag{2.13}$$

The formulation of the governing equations set out here is particularly suited for the shock-fitting calculation procedure outlined in Chapter 3. Equations (2.7)-(2.9) are the building blocks for a method of characteristics, which is applied in shock-free regions of the flow. The shock-free regions, which are separated from one another by shock waves, are then linked by application of equations (2.10)-(2.13).

Chapter 3

Implementation

3.1 Method of Characteristics Solution Procedure

The technique known as the method of characteristics is employed in order to compute the flow within shock-free regions. The method of characteristics seeks characteristic curves within the computational domain, along which the partial differential equations to be solved can be transformed into ordinary differential equations. For supersonic flow, the governing equations (2.7)-(2.9) can be reformulated in such a manner as follows:

$$\frac{dp}{p} + \frac{\gamma}{2}\frac{d\left(M^2\right)}{1 + \frac{\gamma - 1}{2}M^2} = 0 \qquad \text{on } \frac{dy}{dx} = tan\,\theta \tag{3.1}$$

$$\frac{dp}{p} \pm \frac{\gamma M^2}{\sqrt{M^2 - 1}}d\theta = 0 \qquad \text{on } \frac{dy}{dx} = tan(\theta \pm \mu) \qquad (3.2), (3.3)$$

Equations (3.1)-(3.3) are ordinary differential relations, valid only along the specified characteristic curves in the computational domain.

The equations are solved numerically by marching along the characteristic curves, starting from an initial data line, according to the solution procedure outlined by Zucrow and Hoffman [13]. The calculation begins from an initial line, aligned approximately orthogonal to the flow direction, along which the values of all flow quantities are known. For simplicity, this description considers a data line consisting of only two points, denoted *a* and *b*, at which flow quantities are known. At a third point, *c*, flow quantities are not known but may be interpolated by the known values at points *a* and *b*. A fourth point, *d*, is the point at which flow quantities are to be computed.

The geometry of the grid stencil is as follows: *c* is located along the straight line segment that connects *a* and *b*; *a* and *d* lie along the same characteristic curve, which is denoted C_+; *b* and *d* lie along the same characteristic curve, which is denoted C_-; and *c* and *d* lie along a streamline, which is denoted C_0.

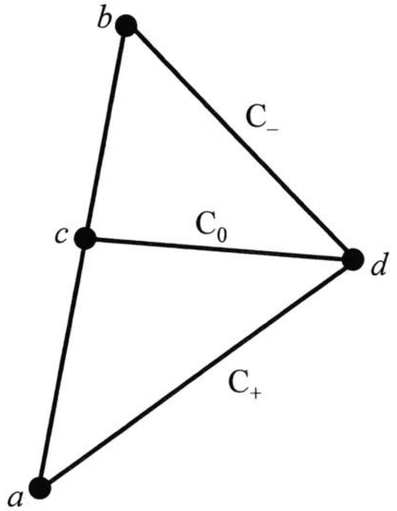

Figure 3-1: The stencil of grid points corresponding to the description of the solution procedure. Points *a* and *b* represent points at which flow quantities are known, point *c* a point at which flow quantities are interpolated from values at *a* and *b*, and point *d* the point at which flow quantities are to be computed. C_+, C_-, and C_0 all denote characteristic curves, with C_0 in particular corresponding to a streamline of the flow.

3.1.1 Predictor Step

The predictor step of the solution procedure produces a preliminary estimate of flow quantities at *d*. First, initial estimates of the coordinates of *c* and *d* are computed by simultaneously solving the following system of equations:

$$y_d - y_a = (x_d - x_a)\, tan\left(\theta_a + \mu_a\right) \tag{3.4a}$$

$$y_d - y_b = (x_d - x_b)\, tan\left(\theta_b - \mu_b\right) \tag{3.5a}$$

$$y_d - y_c = (x_d - x_c)\, tan\left(\frac{\theta_a + \theta_b}{2}\right) \tag{3.6}$$

$$y_c - y_a = (x_c - x_a)\, \frac{y_b - y_a}{x_b - x_a} \tag{3.7}$$

Once initial estimates for the locations of *c* and *d* are obtained, the estimated coordinates of *c* are improved by iteration. Equation (3.8), with θ inserted in place of ϕ, estimates the value of θ_c by linear interpolation between *a* and *b*. Equations (3.7) and (3.9) are then solved simultaneously for improved estimates of x_c and y_c. This process of solving equations (3.7)-(3.9) for improved estimates of θ_c, x_c, and y_c is repeated until a desired level of convergence is achieved.

$$\phi_c - \phi_a = \left(y_c - y_a\right)\, \frac{\phi_b - \phi_a}{y_b - y_a} \tag{3.8}$$

$$y_d - y_c = (x_d - x_c)\, tan\, \theta_c \tag{3.9}$$

Next, the updated values of x_c and y_c are used to estimate M_c, p_c, and ρ_c by linear interpolation between a and b. The interpolation is again given by equation (3.8), with ϕ replaced by the relevant flow quantity (M, p, or ρ).

Having computed a preliminary estimate of the flow quantities at c, the governing equations in characteristic form are solved numerically. First, (3.1)-(3.3) are cast in a form appropriate for a numerical solver. Integration of equation (3.1) along the streamline from c to d yields the exact expression (3.10), whereas discretization of (3.2) and (3.3) yields (3.11a) and (3.12a):

$$\frac{p_d}{p_c} = \left[\frac{1 + \frac{\gamma - 1}{2} M_c^2}{1 + \frac{\gamma - 1}{2} M_d^2}\right]^{\frac{\gamma}{\gamma - 1}} \tag{3.10}$$

$$ln(p_d) - ln(p_a) + \frac{\gamma M_a^2}{\sqrt{M_a^2 - 1}}(\theta_d - \theta_a) = 0 \tag{3.11a}$$

$$ln(p_d) - ln(p_b) - \frac{\gamma M_b^2}{\sqrt{M_b^2 - 1}}(\theta_d - \theta_b) = 0 \tag{3.12a}$$

Equations (3.10), (3.11a), and (3.12a) are solved simultaneously for the unknowns M_d, p_d, and θ_d. The additional unknown flow quantity ρ_d is then computed by the isentropic relation:

$$\frac{p_d}{\rho_d^{\gamma}} = \frac{p_c}{\rho_c^{\gamma}} \tag{3.13}$$

3.1.2 Corrector Algorithm

The quantities so far computed at *d* serve as a preliminary estimate, and are improved by applying a corrector algorithm [13]. Each iteration of the corrector algorithm is nearly identical to the procedure outlined above for computing the preliminary estimate of flow quantities at *d*, only now the preliminary estimates (or, in later iterations, the improved estimates from the previous iteration) are incorporated into the computations. For an improved estimate of x_d and y_d, (3.4a) and (3.5a) are recast as follows:

$$y_d - y_a = (x_d - x_a)\ tan\left(\frac{\theta_a + \theta_d}{2} + sin^{-1}\left[\frac{2}{M_a + M_d}\right]\right) \tag{3.4b}$$

$$y_d - y_b = (x_d - x_b)\ tan\left(\frac{\theta_b + \theta_d}{2} - sin^{-1}\left[\frac{2}{M_b + M_d}\right]\right) \tag{3.5b}$$

An updated value of θ_c is also computed by averaging its previous value with the previously computed value of θ_d:

$$\theta_c = \frac{\theta_c + \theta_d}{2} \tag{3.14}$$

Equations (3.7) and (3.9) are then solved simultaneously to obtain improved estimates of x_c and y_c, and the interpolation equation (3.8) is again applied with $\phi = \theta$ in order to update θ_c. The sequence of first simultaneously solving (3.7) and (3.9) for x_c and y_c, and then solving (3.8) for θ_c, is repeated until a desired level of convergence is achieved.

As in the predictor step, M_c, p_c, and ρ_c are then estimated by the linear interpolation equation (3.8). Finally, (3.11a) and (3.12a) are modified to incorporate previously computed values of quantities at *d*:

$$ln(p_d) - ln(p_a) + \frac{\gamma\left(\frac{M_a + M_d}{2}\right)^2}{\sqrt{\left(\frac{M_a + M_d}{2}\right)^2 - 1}}(\theta_d - \theta_a) = 0 \qquad (3.11b)$$

$$ln(p_d) - ln(p_b) - \frac{\gamma\left(\frac{M_b + M_d}{2}\right)^2}{\sqrt{\left(\frac{M_b + M_d}{2}\right)^2 - 1}}(\theta_d - \theta_b) = 0 \qquad (3.12b)$$

Equations (3.11b) and (3.12b) are solved along with (3.10) and (3.13) to obtain an improved estimate of flow quantities at *d*. The corrector algorithm outlined here is repeated until the computed quantities at *d* converge within a desired tolerance.

3.1.3 Direct Wall Point

The procedure outlined in 3.1.1 and 3.1.2 is modified slightly for the case when the C_0 characteristic curve coincides with a streamline along a solid wall. A stencil of grid points for such a case appears in the following figure.

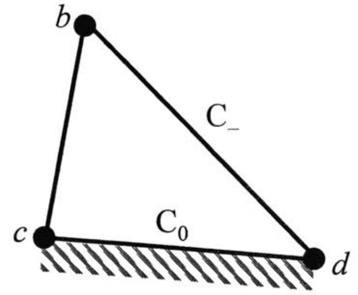

Figure 3-2: The stencil of grid points corresponding to the case when the calculation point *d* is a direct wall point. All quantities at points *b* and *c* are known. Points *b* and *d* lie along the same characteristic curve, C_-. Points *c* and *d*, as well as the C_0 characteristic curve connecting them, all lie along a solid wall.

In the case when the calculation point *d* is a direct wall point, it is assumed that quantities at *c* are known, either from an earlier calculation or from boundary conditions. This is in contrast to the case when *d* is an interior point, and quantities at *c* are estimated from linear interpolation between known quantities at *a* and *b*. Also unique to the direct wall point calculation is the absence of both *a* and the C_+ characteristic curve.

The solution procedure for a direct wall point requires less computation than the interior point case, since there are fewer unknown quantities for which to solve, and it proceeds as follows. First, a preliminary estimate of quantities at *d* is computed. The coordinate positions x_d and y_d are found by simultaneously solving (3.5a) and (3.9). The unknown flow quantities M_d, p_d, and ρ_d are then found by solving (3.10), (3.12a), and (3.13). It is noted that θ_d does not need to be solved for. Rather, it is a known quantity, and equal to the (constant) wall angle in the shock-free physical region corresponding to the computational domain.

With a preliminary estimate of flow quantities at d thus computed, the corrector algorithm is again applied iteratively until convergence within a desired tolerance is achieved. The first step of the corrector algorithm computes improved estimates of x_d and y_d by simultaneously solving (3.9) and (3.5b). In the second and final step of the corrector algorithm, improved estimates of M_d, p_d, and ρ_d are computed by simultaneously solving (3.10), (3.12b), and (3.13).

3.1.4 Special Cases

Attention is drawn to two particular cases of flow regime, in which the method of characteristics solution procedure may be greatly simplified.

The first is the case of uniform flow. Uniform flow exists in regions downstream of a straight shock wave if two additional criteria are satisfied. First, the flow upstream of the straight shock wave must also be uniform, and second, the flow must undergo no further compressions or expansions after the shock wave. In such regions, the solution to the governing equations is a solution in which there is no change in flow quantities along characteristic curves, obviating the need for numerical computation.

The second case, which is slightly more general than the case of uniform flow, is that of irrotational flow. Like uniform flow, irrotational flow appears downstream of straight shock waves when the flow upstream of the straight shock wave is uniform. However, unlike uniform flow, irrotational flow admits the presence of isentropic compression and expansion waves downstream of the shock wave.

To apply the method of characteristics to irrotational flow, the governing equations (2.7)-(2.9) are reduced to the following ordinary differential relations:

$$d\theta \pm \sqrt{M^2 - 1}\,\frac{dV}{V} = 0 \qquad \text{on } \frac{dy}{dx} = tan(\theta \pm \mu) \qquad \text{(3.15), (3.16)}$$

Equations (3.15) and (3.16) for irrotational flow contrast with (3.1)-(3.3) for rotational flow in several important ways. The most obvious difference is the fact that only two equations are required in the description of irrotational flow, whereas three are required for rotational flow. Less immediately obvious is the fact that (3.15)-(3.16) can be written in the compact form:

$$d\theta = \pm\, d\upsilon \qquad \text{on } \frac{dy}{dx} = tan(\theta \pm \mu) \qquad \text{(3.17), (3.18)}$$

The Prandtl-Meyer function, υ, which appears in (3.17) and (3.18), is defined as:

$$\upsilon(M) = \sqrt{\frac{\gamma + 1}{\gamma - 1}}\,tan^{-1}\sqrt{\frac{\gamma - 1}{\gamma + 1}\left(M^2 - 1\right)} - tan^{-1}\sqrt{M^2 - 1} \qquad \text{(3.19)}$$

The numerical solution of (3.17) and (3.18) is quite straightforward. Beginning from two points at which flow quantities are known, (3.17) and (3.18) are integrated exactly along the C_+ characteristic curve passing through one known point, and along the C_- characteristic curve passing through the other known point. The result is two algebraic equations relating M and θ at the point at which the two characteristic curves intersect. The quantities M and θ at the point of intersection are readily solved for, and p and ρ may be subsequently solved for by application of the isentropic relations (3.10) and (3.13).

Note that the isentropic relations may be applied between any two points in the flow field since, by Crocco's theorem, an irrotational flow with uniform stagnation enthalpy also has uniform entropy. In such a manner, the entire flow field may be solved for along characteristic curves, starting from an initial data line.

3.2 Boundary Conditions

The solution methods described above all assume knowledge of an initial line of points at which the values of all flow quantities are known. Far upstream of the displacement body, the initial data line is obtained directly from user-specified freestream quantities. In shock-free regions of the flow located downstream of shock waves, an initial data line is obtained just downstream of each shock wave by application of the oblique shock relations (2.10)-(2.13), which are copied below from Chapter 2 for easier reference:

$$\tan(\theta_2 - \theta_1) = 2\cot(\beta - \theta_1)\,\frac{M_1^2 \sin^2(\beta - \theta_1) - 1}{M_1^2(\gamma + \cos[\,2\{\beta - \theta_1\}\,]) + 2} \tag{2.10}$$

$$M_2^2 \sin^2(\beta - \theta_2) = \frac{M_1^2 \sin^2(\beta - \theta_1) + \dfrac{2}{\gamma - 1}}{\dfrac{2\gamma}{\gamma - 1} M_1^2 \sin^2(\beta - \theta_1) - 1} \tag{2.11}$$

$$\frac{p_2}{p_1} = 1 + \frac{2\gamma}{\gamma + 1}\left(M_1^2 \sin^2(\beta - \theta_1) - 1\right) \tag{2.12}$$

$$\frac{\rho_2}{\rho_1} = \frac{(\gamma + 1)\, M_1^2\, sin^2(\beta - \theta_1)}{(\gamma - 1)\, M_1^2\, sin^2(\beta - \theta_1) + 2} \tag{2.13}$$

The solution procedure described in this chapter is a shock-fitting method of characteristics. Therefore, the existence and location of shock waves in the flow must be arrived at independently of the method of characteristics, which is applied only in regions where shock waves are not present. Shock waves in the flow are generated by sudden deflections, which are induced by the geometry of the displacement body, and so the geometry of the displacement body plays an important role in determining where in the flow an initial data line must be computed. Details of how the geometry of the displacement body is selected are in section 3.2.1.

The geometry of the displacement body is crucial in determining an additional boundary condition for a shock-free region of flow. Each shock-free region is demarcated not only by shock waves, which determine the streamwise extent of the region, but also by an upper and a lower dividing streamline, which limit the extent of the region in the direction normal to the flow. Along these two dividing streamlines, one flow quantity must be arrived at independently of the method of characteristics to serve as a boundary condition on the flow. One type of dividing streamline is a streamline coinciding with a solid wall. Along this type of dividing streamline, the flow quantity that serves as a boundary condition is θ, and thus this boundary condition ties in directly with the geometry of the displacement body.

The other type of dividing streamline is a slip line. A slip line is defined as a line in the flow parallel to the streamlines, across which p and θ are continuous, but other flow quantities, including M, are discontinuous. The boundary conditions associated with a slip line are discussed in section 3.2.2.

3.2.1 Solid Wall Boundary

The results of free-interaction theory are used to set the boundary condition at solid walls by setting the wall angle. Free-interaction theory is discussed extensively in section 2.1.2, as are its implications on the geometry of the displacement body, and so only the key results are summarized here.

The two wall angles that define the displacement body are set such that the pressure at the wall is the same on both sides of the apex of the two-sided ramp, and equal to the plateau value of pressure given by free-interaction theory. For the case of a laminar boundary layer, p_p is computed from (2.3):

$$\frac{p_p - p_0}{q_0} = F(\bar{x}_p)\sqrt{\frac{2C_{f_0}}{\left(M_0^2 - 1\right)^{1/2}}} \qquad (2.3)$$

For the case of a turbulent boundary layer, p_p is computed from (2.5), where the influence of Re through C_f in (2.3) has been eliminated:

$$\frac{p_p}{p_0} = 1 + 0.5M_0 \qquad (2.5)$$

The no-flux boundary condition applied to solid walls requires that the local flow angle be equal to the wall angle. Therefore, the setting of the wall angle by free-interaction theory sets the flow angle, θ, at the solid wall, which provides the boundary condition needed for a solid wall boundary.

3.2.2 Slip Line Boundary

The boundary condition associated with a slip line is less straightforward than that associated with a solid wall. For a slip line, the boundary condition is set by coupling the method of characteristics solution on either side of the slip line with the two no-jump conditions that must be satisfied across the slip line. Specifically, the condition that p and θ are continuous across the slip line is enforced.

In order to illustrate how the boundary condition at a slip line is implemented for an irrotational flow, a discrete change in direction of a slip line is considered. The initial Mach number above the slip line, M_{Ui}, is larger than the initial Mach number below it, M_{Li}. The values of the quantities M, p, and θ are desired at locations U and L, located just above and just below the slip line immediately downstream of the deflection of the slip line. Flow quantities upstream of these two locations are known on both sides of the slip line from the method of characteristics applied in that region.

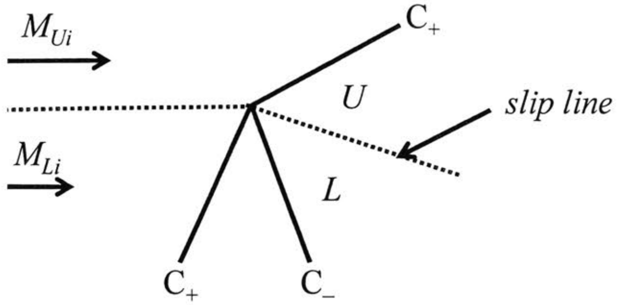

Figure 3-3: A slip line undergoing a discrete deflection. For the deflection depicted, the C_+ characteristic below the slip line represents an incident expansion wave. The C_+ characteristic above the slip line is the transmitted expansion wave, and the C_- characteristic is a reflected compression wave.

In order to compute the six unknown quantities (M, p, and θ, at the two locations U and L), six equations relating the unknown quantities are required. Without a slip line, the method of characteristics for irrotational flow would provide all six of the required relations. These relations would be the equations associated with the two characteristic curves C_+ and C_-, (3.17) and (3.18), along with the isentropic pressure relation (3.10). These three equations, applied both above and below the slip line, would make the six required to solve for the six unknown quantities.

However, two of the six equations cannot be applied in the presence of a slip line. The C_+ characteristic passing through U crosses the slip line when extended in the upstream direction. But a singularity in vorticity exists at the slip line, due to the discontinuity in velocity across it. Thus (3.17), which relies on the assumption of irrotationality along the entire characteristic curve, is no longer valid. A similar situation is encountered if one considers the C_- characteristic passing through L, rendering equation (3.18) invalid.

In place of (3.17) applied at *U* and (3.18) applied at *L*, the following two continuity conditions are enforced across the slip line:

$$p_U = p_L \tag{3.20}$$

$$\theta_U = \theta_L \tag{3.21}$$

In summary, to incorporate the boundary condition at a slip line into the method of characteristics solution procedure for an irrotational flow, equations (3.20) and (3.21) are used to couple the two flow regions separated by the slip line.

3.3 Shock/Wave Interactions

Two types of shock/wave interaction, which are relevant to the displacement-body model calculations, are discussed. The first is the interaction between two shock waves. The second is the interaction between a shock wave and a slip line.

3.3.1 Shock-Wave/Shock-Wave Interaction

In the flow around the displacement body, an intersection between two shock waves occurs where the incident oblique shock wave, which impinges on the boundary layer, meets the separation shock wave, which is formed by the upstream compression corner of the displacement body. The flow downstream of the intersection is determined by application of the oblique shock relations (2.10)-(2.13), as well as additional physical requirements that the flow must satisfy.

When the two shock waves intersect, both are deflected and transmitted downstream. The shock waves deflect such that pressure and flow angle are continuous everywhere downstream of the interaction, except across shock waves. The case of an interaction between two straight shock waves is illustrated below. The general solution admits the presence of a slip line, across which p and θ are continuous but other flow quantities are not.

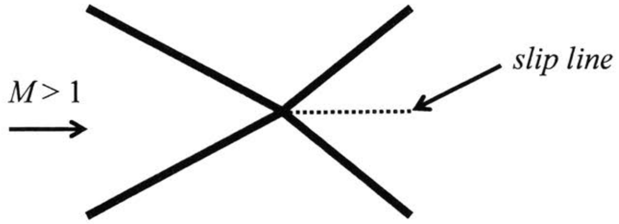

Figure 3-4: A Type I interaction between two intersecting shock waves. The shock waves (solid lines) are straight both upstream and downstream of the intersection, and a slip line extends downstream from the point of intersection.

The interaction between the incident oblique shock wave and the separation shock wave is assumed to be of the Type I category, according to the classification system defined by Edney [14]. Thus, equations (2.10)-(2.13) admit a solution consistent with Figure 3-4, in which the two straight shock waves are transmitted downstream of their intersection as straight shock waves, and a slip line exists downstream of the intersection. The case where the upstream Mach number and the strength of the intersecting shock waves do not admit a solution consistent with Figure 3-4 is a Type II interaction, and is not considered. In general, (2.10)-(2.13) admit two solutions for each of the transmitted shock waves, one known as the strong shock solution and the other the weak shock solution. In cases where both solutions are possible, the weak shock solution is chosen.

3.3.2 Shock-Wave/Slip-Line Interaction

The second interaction that is considered is that between a shock wave and a slip line. In the flow induced by the displacement-body model, such an interaction occurs between the reattachment shock wave, which is formed by the downstream compression corner of the displacement body, and the slip line, which originates from the interaction described in section 3.3.1.

The interaction between an incident shock wave and a slip line is characterized by the following features [1]. Both the incident shock wave and the slip line extend downstream past the point at which the two intersect, but both undergo a finite deflection at the point of intersection. In addition to these first two phenomena, which are analogous to the case of a shock-wave/shock-wave interaction, the incident shock is also reflected off the slip line as a fan of expansion waves.

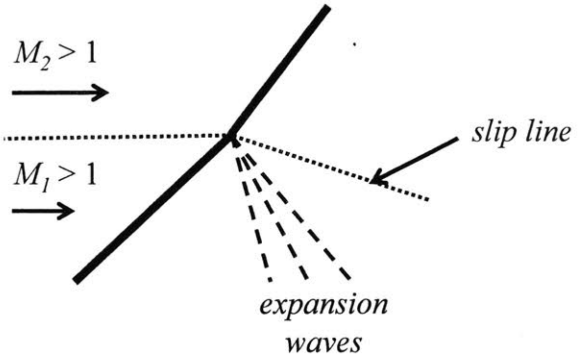

Figure 3-5: An illustration of interaction between a shock wave and a slip line. Upon reaching the slip line, the incident shock wave (which approaches the slip line from below) is transmitted as a deflected shock wave, and also reflected as a series of expansion waves. The slip line also undergoes a deflection at the point of intersection.

The oblique shock relations (2.10)-(2.13) describe the flow downstream of the transmitted shock wave, and the flow quantities through the expansion wave are described by equations (3.17) and (3.18), as well as the isentropic relations (3.10) and (3.13). These equations are combined with the slip line conditions of continuous p and θ to determine the strengths of both the transmitted shock and the expansion wave fan.

3.4 Numerical Procedure for Curved Shock Waves

The reattachment shock wave, which is generated by the downstream compression corner of the displacement body, extends into a region of nonuniform flow. As a result, the shape of the shock wave is curved. In order to compute its shape, the procedure introduced by Moeckel [15] is adopted.

For a shock wave with isentropic upstream conditions, Moeckel presents the following approximate expression relating the change in shock wave angle to the change in upstream Mach number:

$$\frac{d\beta}{dM} = -\frac{\frac{1}{g_1}\frac{\partial g_1}{\partial M} + g_3\frac{\partial g_2}{\partial M}}{\frac{1}{g_1}\frac{\partial g_1}{\partial \beta} + g_3\frac{\partial g_2}{\partial \beta}} - \frac{\left(-\gamma M \pm \frac{\sqrt{M^2-1}}{M} g_3\right)\left(1 + \frac{\gamma-1}{2}M^2\right)^{-1}}{\frac{1}{g_1}\frac{\partial g_1}{\partial \beta} + g_3\frac{\partial g_2}{\partial \beta}} \tag{3.22}$$

For the plus-minus sign in (3.22), the positive is taken where $\frac{d\theta}{dM} > 0$ just upstream of the shock wave, and the negative is taken where $\frac{d\theta}{dM} < 0$. A right-running isentropic wave just upstream of the shock wave falls under the first category, while a left-running isentropic wave in a similar location falls under the second.

The three functions denoted g, which appear in (3.22), are defined as:

$$g_1 = \frac{2\gamma}{\gamma+1}M^2 sin^2\beta - \frac{\gamma-1}{\gamma+1} \tag{3.23}$$

$$g_2 = tan^{-1}\left[\frac{M^2 sin\beta\, cos\beta - cot\beta}{1 + M^2\left(\frac{\gamma+1}{2} - sin^2\beta\right)}\right] \tag{3.24}$$

$$g_3 = \frac{\gamma M_s^2}{\sqrt{M_s^2 - 1}} \tag{3.25}$$

Analytical expressions for the partial derivatives of (3.23)-(3.25), which also appear in (3.22), are readily obtained. The additional parameter M_s, introduced in (3.25), is defined as:

$$M_s^2 = \frac{(\gamma+1)^2 M^4 sin^2\beta - 4\left(M^2 sin^2\beta - 1\right)\left(\gamma M^2 sin^2\beta + 1\right)}{\left(2\gamma M^2 sin^2\beta - \gamma + 1\right)\left([\gamma - 1]M^2 sin^2\beta + 2\right)} \tag{3.26}$$

For known upstream conditions and an initial shock wave angle, (3.22) is solved numerically for the shape of the shock wave. In order to do so, (3.22) is discretized with a central difference scheme:

$$\frac{\beta_{i+1} - \beta_i}{M_{i+1} - M_i} = \left(\frac{d\beta}{dM}\right)_{i+\frac{1}{2}} \tag{3.27}$$

In (3.27), subscripts represent indices in the finite difference scheme, with *i* denoting an integer corresponding to a specific grid point. The right hand side of (3.27) is evaluated using the definition of $\frac{d\beta}{dM}$ provided by (3.22). Evaluation of a flow quantity at a midpoint between two adjacent grid points is approximated by linear interpolation between the known values at the adjacent grid points. Thus, for some quantity ϕ:

$$\phi_{i+\frac{1}{2}} = \frac{\phi_{i+1} + \phi_i}{2} \tag{3.28}$$

Chapter 4

Results and Discussion

4.1 Resolution Parameters and Grid Convergence

In the numerical solution procedure that is outlined in Chapter 3, there are four user-specified resolution parameters that influence the accuracy of the numerical solution. Two of these parameters are related to the grid resolution. The first grid resolution parameter is N_A. It represents the number of finite expansion waves that are used to approximate the continuous expansion fan occurring at the apex of the displacement body. The second grid resolution parameter is N_B. It represents the number of finite expansion waves that are used to approximate the continuous expansion fan resulting from the interaction between the reattachment shock wave and the slip line.

The other two resolution parameters specify the number of iterations carried out in the predictor-corrector algorithm, and they therefore describe the numerical resolution. One of these numerical resolution parameters is denoted k, and it sets the number of iterations carried out to improve the estimate of flow quantities at the interpolated grid point. The interpolated grid point is denoted point c in Chapter 3. The other parameter is denoted l, and it sets the number of iterations of the corrector algorithm that are carried out for each calculation point. The calculation point is denoted point *d* in Chapter 3.

In order to perform a convergence study, a default value was chosen for each resolution parameter. The default values were chosen to be large in order to produce a grid-converged solution, and they are summarized in Table 4.1. The flow specifications used in the convergence studies also appear in Table 4.1.

Table 4.1: Default values for the resolution parameters and the flow specifications used in the convergence studies.

Resolution Parameters		Flow Specifications	
N_A	100	M_0	8.6
N_B	100	$\beta_{incident}$	-19.8°
k	20	Flow Regime	Turbulent
l	20		

The grid-converged flow field is depicted in Figure 4-1. The displacement body is shaded black, and various features of the flow field are depicted. These features are: shock waves, which are represented with thick solid lines; the slip line, which is represented with a dotted line; and isentropic waves, which are represented with thin solid lines.

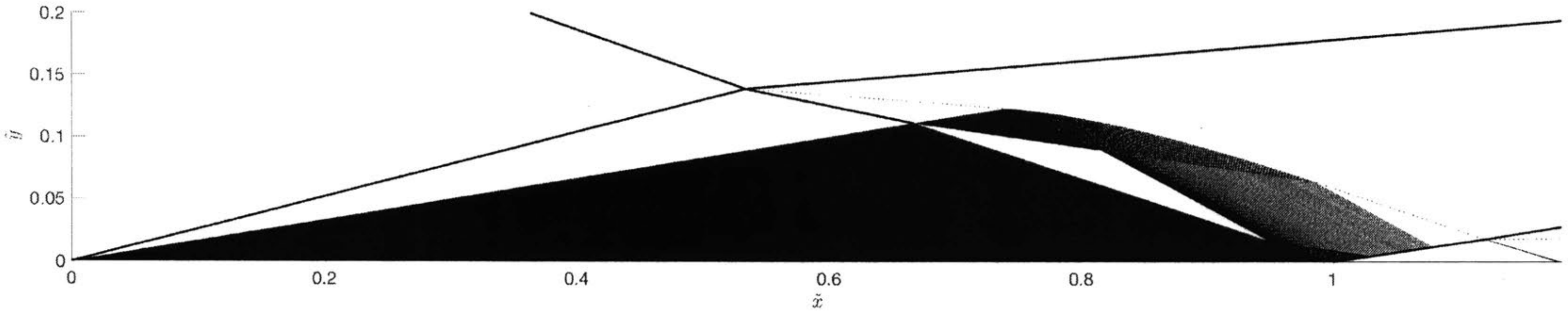

Figure 4-1: The grid-converged flow field over the displacement body. Depicted are: the displacement body (shaded), shock waves (thick solid lines), the slip line (dotted line), and isentropic waves (thin solid lines).

Immediately apparent in Figure 4-1 is the very dense fan of expansion waves that originates at the apex of the displacement body. The number of expansion waves, and thus the density of the fan, is a direct function of N_A; since N_A is large for the grid-converged solution, the number of waves is also large. Further downstream, the expansion waves reflect off the slip line to form compression waves. Some of the compression waves may then reflect off the displacement body, but all eventually reach the reattachment shock wave. Since all the isentropic waves originate at the apex of the displacement body, and are neither created nor destroyed downstream, the number of compression waves that reach the reattachment shock wave is equal to N_A.

Since flow quantities are computed along each isentropic wave by the method of characteristics, the number of isentropic waves chosen to model the expansion fan determines the resolution of the numerical grid. The importance of the parameter N_A in determining the grid resolution is therefore paramount. The influence of N_A on grid resolution is not just limited to the region between the displacement body apex and the reattachment shock wave. Since N_A determines the number of data points that are computed just upstream of the reattachment shock wave, and the same number of data points just downstream of the shock wave are obtained by applying the oblique shock relations to the upstream points, the parameter N_A determines the grid resolution downstream of reattachment as well. The global importance of N_A is demonstrated in Figure 4-2, which shows the grid downstream of the reattachment shock wave for N_A = 10 and N_A = 50. The grid for the N_A = 50 case is clearly the more resolved of the two. It is also interesting to note that, for both cases, the grid becomes coarser as one moves along the reattachment shock wave away from the solid wall ($\tilde{y}_w = 0$). Gradients in the flow quantities on either side of the reattachment shock wave tend to decrease in magnitude as distance from the reattachment point increases, and so the grid at such locations can be coarser without significant adverse effects on the solution accuracy.

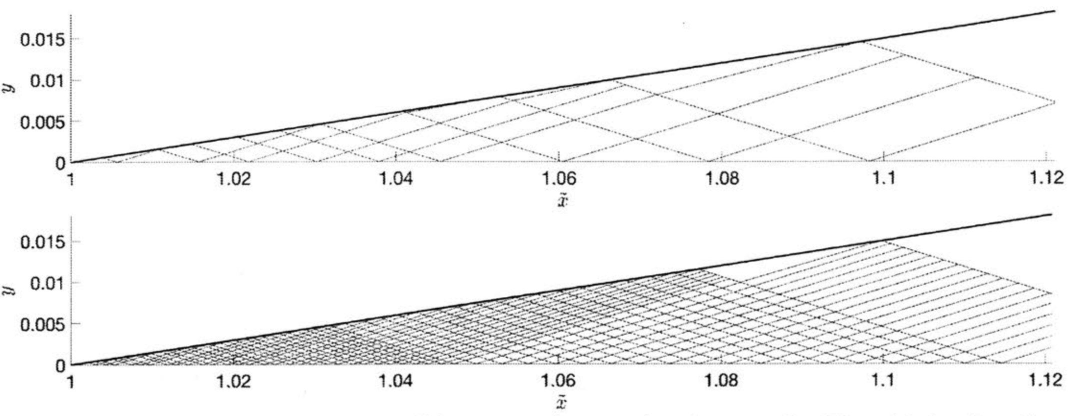

Figure 4-2: The grid downstream of the reattachment shock wave for $N_A = 10$ (top) and $N_A = 50$ (bottom). Depicted for each case are the C_+ and C_- characteristic curves, the intersection points of which correspond to calculation points.

In order to determine the resolution necessary to obtain a grid-converged solution, the effects of N_A, N_B , k, and l on solution accuracy were independently assessed. An identical procedure was followed for each of the four resolution parameters. To assess grid convergence for a resolution parameter, the value of the resolution parameter of interest was gradually increased from its minimum possible value until the sensitivity of a calculated output quantity to further refinement of the resolution parameter was judged to be insignificant. During this process, all other resolution and flow parameters were held fixed at the values given in Table 4.1. The results of the grid convergence study for N_A are displayed in Figure 4-3.

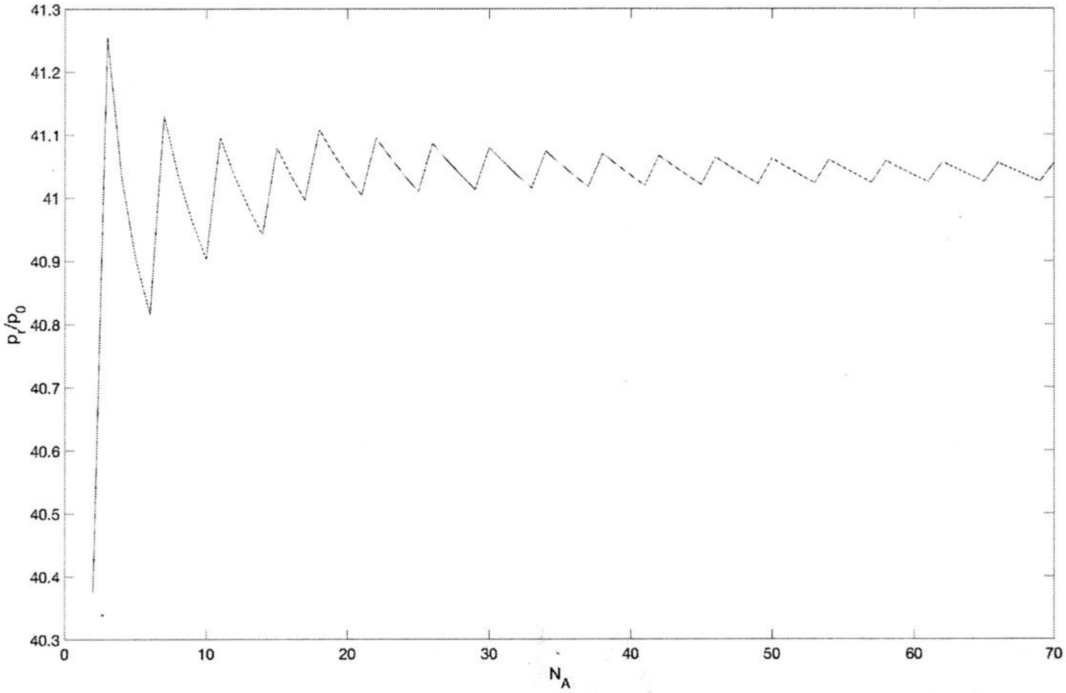

Figure 4-3: Grid convergence study for N_A. The output quantity p_r/p_0 is plotted against N_A.

As N_A is increased, p_r/p_0 oscillates about a mean value of approximately 41.04. Although the oscillatory behavior persists as the grid is refined further and further, it is noted that the amplitude of the oscillation decays. For $N_A \approx 70$, the amplitude has decayed to 0.028, or 7E-4 as a fraction of the mean value. Since the displacement body model itself represents a significant approximation of the viscous separation bubble, a discretization error of this magnitude is deemed to be insignificant, and the grid is considered converged at $N_A = 70$.

For N_B, k, and l, the output quantity p_f/p_0 is monitored instead of p_r/p_0, and it is used as a proxy for convergence of the numerical solution. The quantity p_f is defined as the pressure at the wall just downstream of the expansion fan that results from the interaction of the reattachment shock wave and the slip line (see Figure 4-1, where the location of p_f corresponds to $\tilde{x} = 1.18$, the upper limit of the horizontal axis). The reason for this choice is simple. Unlike N_A, the resolution parameters N_B, k, and l only affect computations in the rotational flow region downstream of the reattachment shock wave. Therefore, due to the hyperbolic nature of supersonic flow, manipulation of these three resolution parameters only affects the numerical solution within and downstream of the rotational region. The quantity p_f is chosen because it is measured downstream of the rotational region, whereas the quantity p_r is not.

The effect of N_B on the output quantity p_f/p_0 is displayed in Figure 4-4. Unlike the plot Figure 4-5, which exhibited an oscillatory behavior, this plot exhibits a purely decaying behavior, with p_f/p_0 approaching a value of 39.93959 for the grid-converged solution. Two important observations are noted. First, some perspective of the vertical axis scale is warranted. Even with $N_B = 2$, the value of p_f/p_0 is within 0.001% of the grid-converged value. Therefore, any improvement in solution accuracy that results in refining the resolution parameter N_B is so small that its effect on the numerical solution is nearly negligible. Second, with $N_B = 10$, the fractional error in p_f/p_0 compared to the grid-converged solution is 2E-6. At this point, the solution is deemed to be converged.

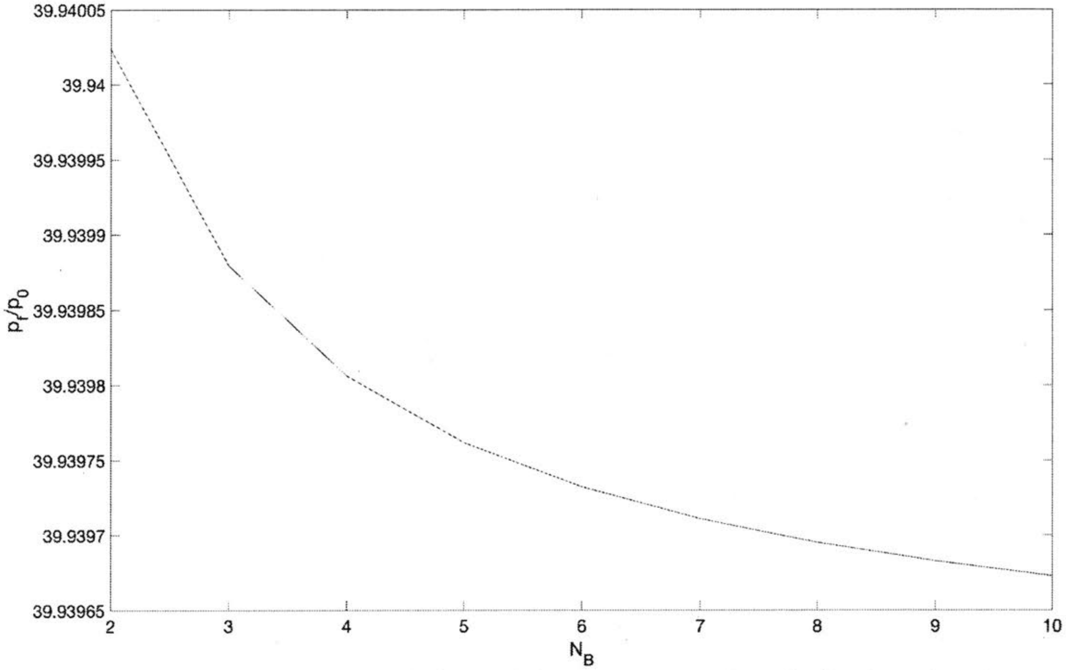

Figure 4-4: Grid convergence study for N_B. The output quantity p_f/p_0 is plotted against N_B.

Convergence studies for the final two resolution parameters, k and l, are shown in Figure 4-5 and Figure 4-6. The two plots are nearly identical, and both show a very weak dependence of the numerical solution on the resolution parameter. For both $k = 5$ and $l = 5$, the fractional error in the output quantity p_f/p_0 is less than 2.5E-15 compared to the grid-converged value. This error is small enough to state that the solution is converged at $k = l = 5$.

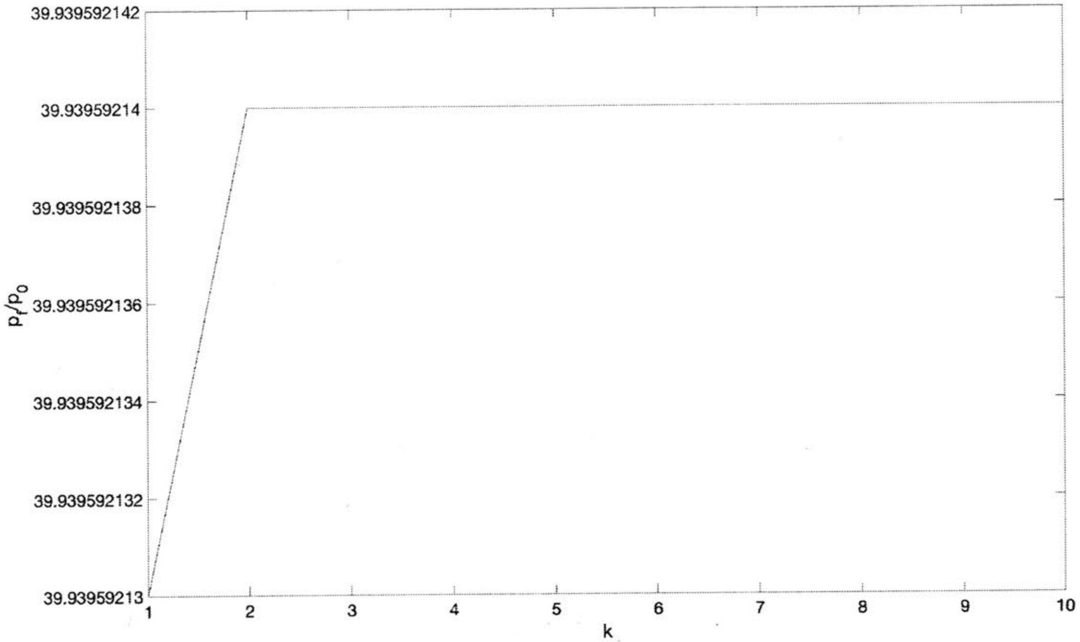

Figure 4-5: Convergence study for *k*. The output quantity p_f/p_0 is plotted against *k*.

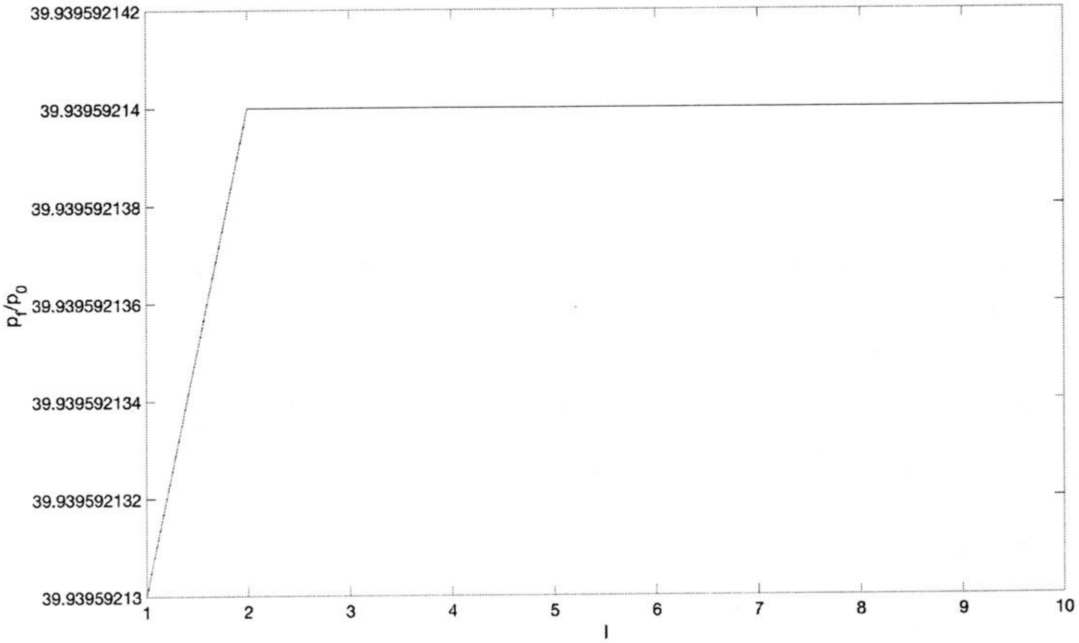

Figure 4-6: Convergence study for *l*. The output quantity p_f/p_0 is plotted against *l*.

Based on the results of the four convergence studies, a value for each resolution parameter, corresponding to a grid-converged solution, was selected. These values are summarized in Table 4.2, and are used to produce all the results that appear hereafter in Chapter 4.

Table 4.2: Grid-converged values for the four resolution parameters.

Resolution Parameter	Grid-Converged Value
N_A	100
N_B	20
k	5
l	5

4.2 Comparisons to Experimental Data

The predictive capability of the displacement-body model was assessed by comparing its prediction of quantities at the reattachment location to experimentally measured values. The comparison is made for both laminar and turbulent boundary layers. Experimental data on laminar boundary layers are provided by Barry et al. [4], and experimental data on turbulent boundary layers are provided by Holden [16]. The laminar experiments were conducted at $M_0 = 2.05$, and the strength of the incident shock wave is reported as $\Delta\theta_{incident}$, the change in flow angle through the incident shock wave. The turbulent experiments were conducted at two different Mach numbers, $M_0 = 8.6$ and $M_0 = 11.3$, and the strength of the incident shock wave is reported as $\beta_{incident}$, the angle between the incident shock wave and the x-axis. Results from the two experiments, along with results from the displacement-body model, are presented in Table 4.3 and Table 4.4.

Table 4.3: Comparison of experimental data [4] and displacement-body model data for laminar boundary layers. For all cases, $M_0 = 2.05$. Error is reported as a percentage of the experiment value. In the displacement-body calculation, C_{f0} is estimated from Re_0 using the incompressible Blasius solution and the temperature correction given by Hakkinen et al. [5], which uses the Chapman-Rubesin relationship between viscosity and temperature.

$\Delta\theta_{incident}$	Re_0	$(\rho_r/\rho_0)_{experiment}$	$(\rho_r/\rho_0)_{model}$	Error
-3°	600,000	1.22	1.27	4 %
-6°	260,000	1.53	1.58	3 %
	630,000	1.54	1.58	3 %

Table 4.4: Comparison of experimental data [16] and displacement-body model data for turbulent boundary layers. Error is again reported as a percentage of the experiment value.

M_0	$\beta_{incident}$	$(p_r/p_0)_{experiment}$	$(p_r/p_0)_{model}$	Error
8.6	-19.8°	36.8	41.0	11 %
11.3	-17.6°	58.9	63.0	7 %

For both laminar and turbulent boundary layers, the displacement-body model overpredicts the non-dimensional density (for the laminar boundary layer) or pressure (for the turbulent boundary layer) at reattachment. The accuracy appears to be better for the laminar case (3-4 %) than for the turbulent case (7-11 %). Considering the underlying assumptions and simplicity of the displacement-body model, it appears to do a remarkable job predicting flow quantities at reattachment. However, there are a couple of caveats attached to this statement. First, neither set of experimental data provides an estimate of uncertainty for the reported measurements, and therefore the model may be more or less accurate than the calculated error values suggest. Second, even if the experimental data are taken at face value, the sample size of five scenarios is not large enough to generalize and quantify the error that should be expected in the model prediction across a larger range of M_0 and incident shock wave strengths. Nonetheless, the comparison with experimental data does lend credibility to the displacement-body model and its predictive capability.

One puzzling trend for the turbulent data is also remarked on. In general, as M increases, μ decreases, and therefore the slopes of the C_+, C_0, and C_- characteristic curves tend to approach the same value. For a given set of initial data points, this trend presents a problem for the method of characteristics. As M is increased, a calculation point defined by the intersection of the characteristic curves originating from three points along the initial data line moves further and further downstream. Therefore, the grid resolution becomes coarser at larger values of M, and the solution accuracy is expected to suffer as a result.

The trend of reduced accuracy for larger M is not what is observed in Table 4.4. In fact, the turbulent data in Table 4.4 suggest that the model is more accurate at $M_0 = 11.3$ than at $M_0 = 8.6$. The most likely reason for the higher accuracy at $M_0 = 11.3$ has to do not with the method of characteristics, but rather with the displacement-body model itself. At $M_0 = 11.3$, the conditions of the experiment are more consistent with the assumptions of the displacement-body model than are the conditions of the experiment at $M_0 = 8.6$. In the $M_0 = 8.6$ case, the incident shock wave strength is only slightly larger than the minimum strength required to induce separation of the boundary layer. The result is that the actual surface pressure distribution inside the separation bubble does not resemble the approximate surface pressure distribution assumed in the displacement-body model, which is a constant distribution at the plateau value of pressure (see section 2.1.2 for more details). Therefore, it is likely that the increased modeling error for the $M_0 = 8.6$ case is more influential than the increased numerical error for the $M_0 = 11.3$ case, and this would explain the counterintuitive relationship between accuracy and M_0 in Table 4.4.

4.3 Effect of Varying Input Parameters

The two user-defined input parameters for the displacement-body model calculations are $\beta_{incident}$, which sets the incident shock wave strength, and M_0, which sets the upstream Mach number. In this section, the influence of these two input parameters on the solution for a turbulent boundary layer is explored.

4.3.1 Influence of Incident Shock Wave Strength

The first input parameter that is investigated is the incident shock wave strength. Its influence on the solution is studied by varying the input parameter $\beta_{incident}$ while holding the upstream Mach number constant at $M_0 = 5$. As previously stated, the results that follow assume a turbulent boundary layer.

In Figure 4-7, the non-dimensional value of p_p is plotted against the pressure ratio across the incident shock wave, $(p_2/p_1)_{incident}$. Clearly, the plateau pressure does not depend at all on the incident shock wave strength. This is by design. The plateau pressure value is not a quantity that is computed with the outer inviscid flow solver. Rather, the plateau pressure is an input to the displacement-body model, and its value is calculated independently of the inviscid flow solver, using the results of free-interaction theory. For a turbulent boundary layer, free-interaction theory suggests that the plateau pressure depends only upon M_0, and since M_0 is held constant as the incident shock wave strength is changed, p_p remains unchanged.

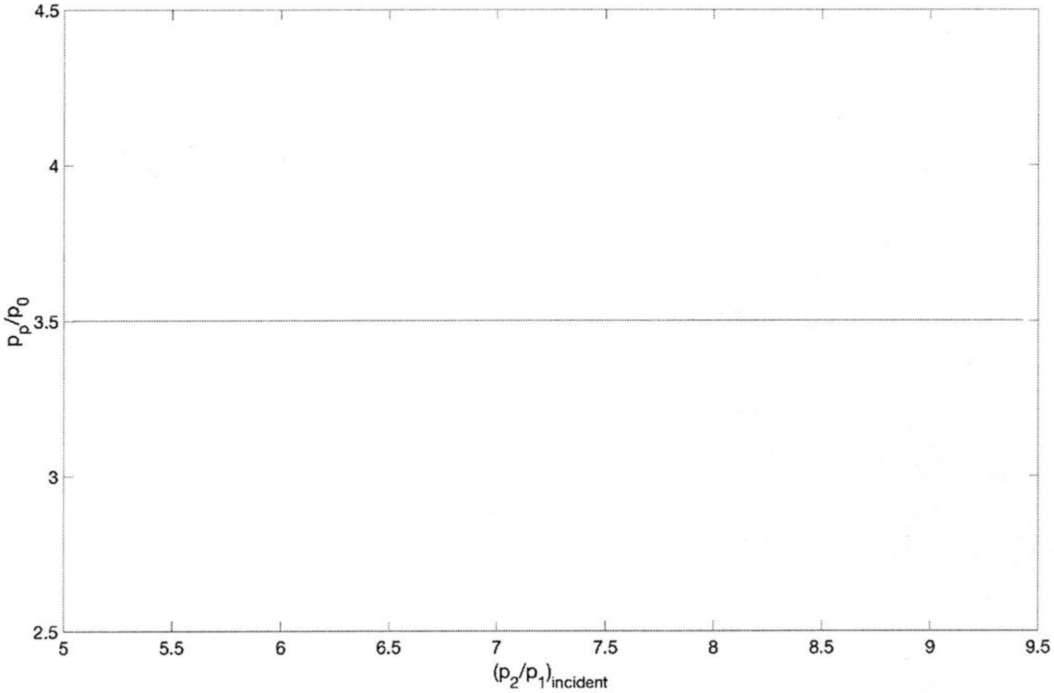

Figure 4-7: The effect of incident shock wave strength on plateau pressure. The non-dimensional plateau pressure, p_p/p_0, is plotted against the pressure ratio across the incident shock wave, $(p_2/p_1)_{incident}$.

The trend of the overall pressure rise through the separation bubble as incident shock wave strength is increased is more insightful. Figure 4-8, where the non-dimensional value of p_r is plotted against $(p_2/p_1)_{incident}$, illustrates this trend. There is a strong positive relationship between the strength of the incident shock wave and the overall pressure rise. A least squares linear regression over the range of data in Figure 4-8 has a slope of 4.5, suggesting that a unit increase in $(p_2/p_1)_{incident}$ in this range of incident shock wave strengths results in approximately a 4.5 factor increase in p_r/p_0. However, it is important to note that the association in Figure 4-8 is not linear; the influence of the incident shock wave strength on pressure rise across the separation bubble becomes stronger as the incident shock wave strength is increased. In fact, the data plotted in Figure 4-8 follow the approximate power law $\frac{p_r}{p_0} \sim \left(\frac{p_2}{p_1}\right)_{incident}^{1.3}$. The non-linearity is most clear visually for $(p_2/p_1)_{incident} > 9$, but it is present over the entire range of incident shock wave strengths that were investigated.

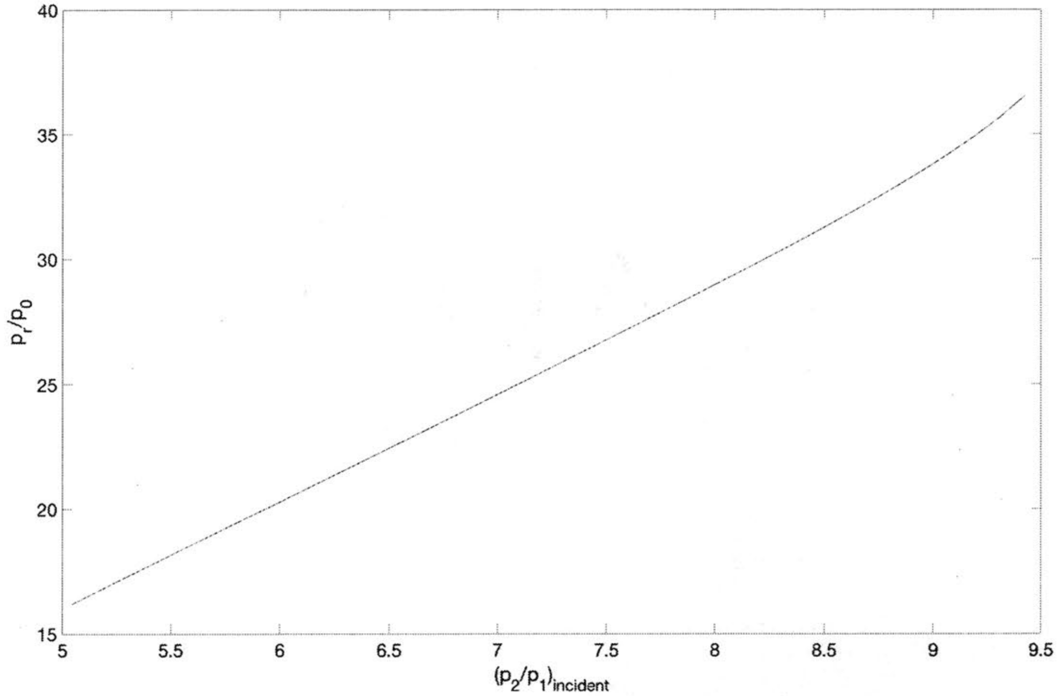

Figure 4-8: The effect of incident shock wave strength on the overall pressure rise through the separation bubble. The non-dimensional pressure at reattachment, p_r/p_0, is plotted against the pressure ratio across the incident shock wave, $(p_2/p_1)_{incident}$.

In addition to the quantitative influence on surface pressure distribution, the incident shock wave strength exerts a subtle qualitative influence on the structure of the outer inviscid flow. This qualitative influence of incident shock wave strength is illustrated in Figure 4-9, which shows the flow fields for $\beta_{incident} = -25°$ and $\beta_{incident} = -30°$. The former corresponds to $(p_2/p_1)_{incident} = 5.04$, and the latter corresponds to $(p_2/p_1)_{incident} = 7.13$. The upstream Mach number is still $M_0 = 5$.

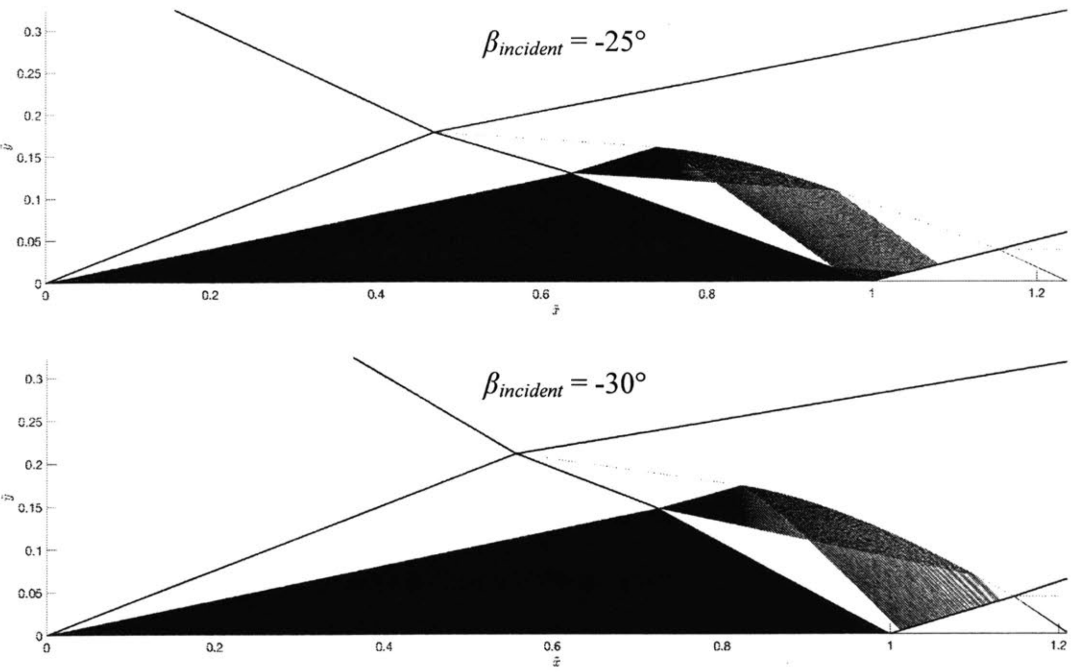

Figure 4-9: The flow fields for $\beta_{incident}$ = -25° (top) and $\beta_{incident}$ = -30° (bottom). For information on the various features depicted, refer to the caption for Figure 4-1.

The overall structures of the two flow fields are quite similar. For both flow fields, M_0 = 5 and p_p/p_0 = 3.5. Therefore, the separation shock wave and the compression corner that produces the separation shock wave are identical for both cases. The obvious difference between the two flow fields is the angle of the incident shock wave, which is more oblique for $\beta_{incident}$ = -25° than for $\beta_{incident}$ = -30°.

A more subtle difference between the two flow fields regards the expansion fan that originates at the apex of the displacement body. In non-dimensional coordinates, the apex is located further downstream for the case of the stronger incident shock wave ($\beta_{incident}$ = -30°). The result is that the flow goes through a stronger expansion for the case of the stronger incident shock wave. This behavior is expected. Just before the expansion, flow along the displacement body surface passes through the deflected incident shock wave, across which the pressure rise is greater for the case of the stronger incident shock wave. However, the pressure must return to the same plateau value for both flows after the expansion. A stronger expansion is therefore required for the case of the stronger incident shock wave.

At the reattachment point, both flows pass through a reattachment shock wave in order to once again be tangent to the wall. The flow must be turned through a larger angle for the $\beta_{incident}$ = -30° case, since the flow in this case has undergone a stronger expansion. This line of reasoning suggests that the reattachment shock wave should be stronger for the $\beta_{incident}$ = -30° case than for the $\beta_{incident}$ = -25° case. That the reattachment shock wave is in fact stronger for the $\beta_{incident}$ = -30° case is confirmed by the trend of Figure 4-8, which, combined with the knowledge that p_p/p_0 is the same for both cases, shows that p_r/p_p is larger for the stronger incident shock wave.

4.3.2 Influence of Mach Number

To investigate the influence of Mach number on the solution, the input parameter M_0 is varied between $M_0 = 3$ and $M_0 = 15$. While M_0 is varied, the value of $\Delta\theta_{incident}$ is held constant at -20°. A physical interpretation of these conditions is that an inverted, 20° wedge located above the boundary layer is held in place while upstream Mach number is varied. Again, the results reported here assume that the boundary layer is turbulent.

The first quantity of interest, the non-dimensional value of p_p, is plotted against M_0 in Figure 4-10. As previously discussed in section 4.3.1, p_p is not a quantity that is computed with the inviscid flow solver, but a quantity that is computed from free-interaction theory to serve as an input to the inviscid flow solver. The linear relationship in Figure 4-10 is simply a graphical representation of equation (2.5), which states that for a turbulent boundary layer, the non-dimensional value of p_p is a linear function of M_0. Nonetheless, the contrast between Figure 4-7, which shows no dependence of p_p on the incident shock wave strength, and Figure 4-10, which shows the linear dependence of p_p on Mach number, underlines that important result from free-interaction theory. That result states that plateau pressure depends only on upstream conditions, and not on the strength of the incident shock wave.

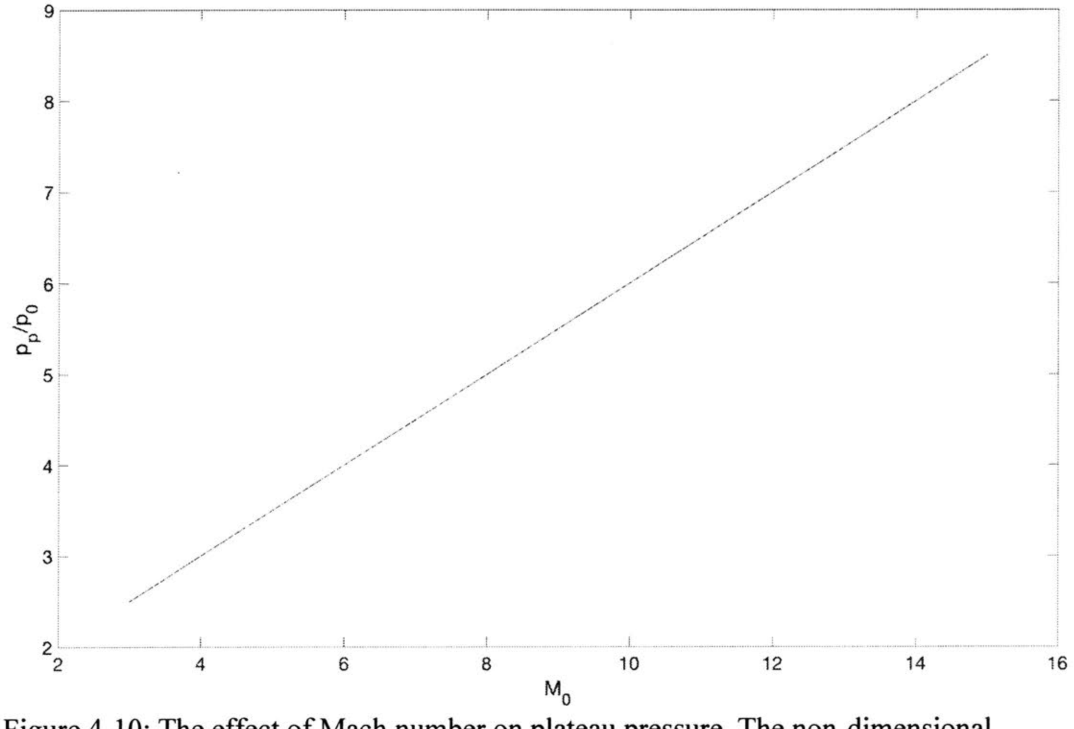

Figure 4-10: The effect of Mach number on plateau pressure. The non-dimensional plateau pressure, p_p/p_0, is plotted against the upstream Mach number, M_0.

In Figure 4-11, the influence of Mach number on the overall pressure rise through the separation bubble is presented. The non-dimensional value of p_r is plotted against M_0.

The relationship between the overall pressure rise and Mach number is qualitatively similar to that between the overall pressure rise and incident shock wave strength. Like Figure 4-8, in which overall pressure rise is plotted against incident shock wave strength, Figure 4-11 shows a strong positive relationship between overall pressure rise and the independent variable, which in this case is upstream Mach number. A least squares linear regression of the data over this range of Mach numbers suggests that a unit increase in M_0 results in an increase in p_r/p_0 by a factor of approximately 16. However, like in Figure 4-8, the association between the two quantities in Figure 4-11 is non-linear. In this case, the association is more accurately described by the approximate power law $\frac{p_r}{p_0} \sim M_0^{1.9}$, which is nearly quadratic.

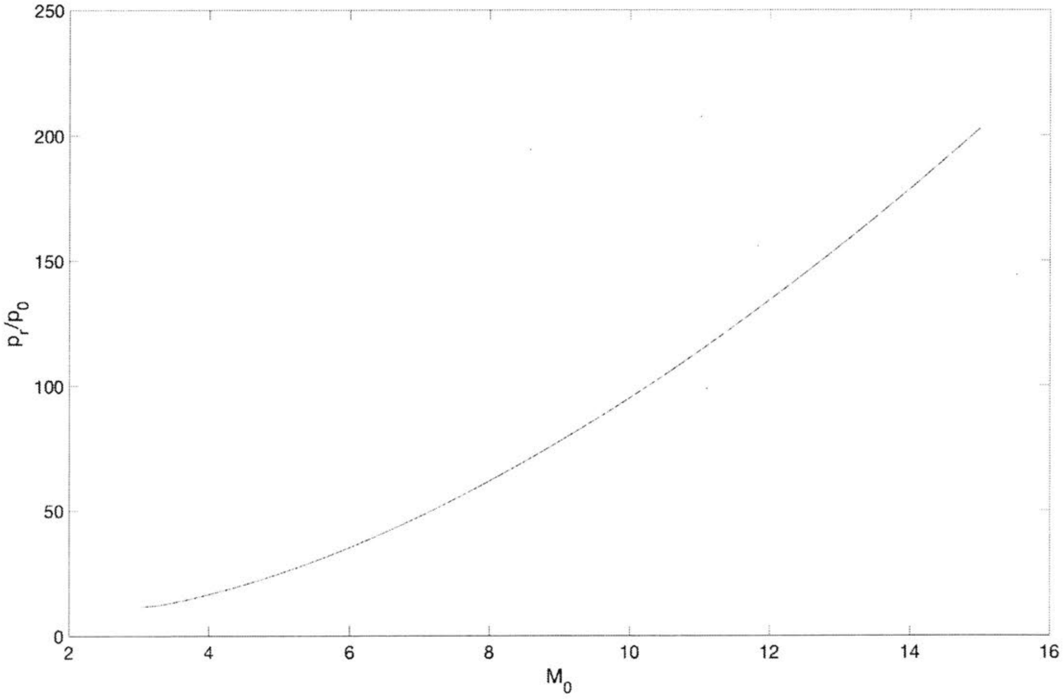

Figure 4-11: The effect of Mach number on the overall pressure rise through the separation bubble. The non-dimensional pressure at reattachment, p_r/p_0, is plotted against the upstream Mach number, M_0.

The qualitative dependence of the outer inviscid flow field on Mach number is illustrated in Figure 4-12. In this figure, the structure of the flow field over the displacement body is depicted for $M_0 = 3$, $M_0 = 8$, and $M_0 = 13$. The change in flow angle through the incident shock wave for the three flow fields pictured is $\Delta\theta_{incident} = -15°$.

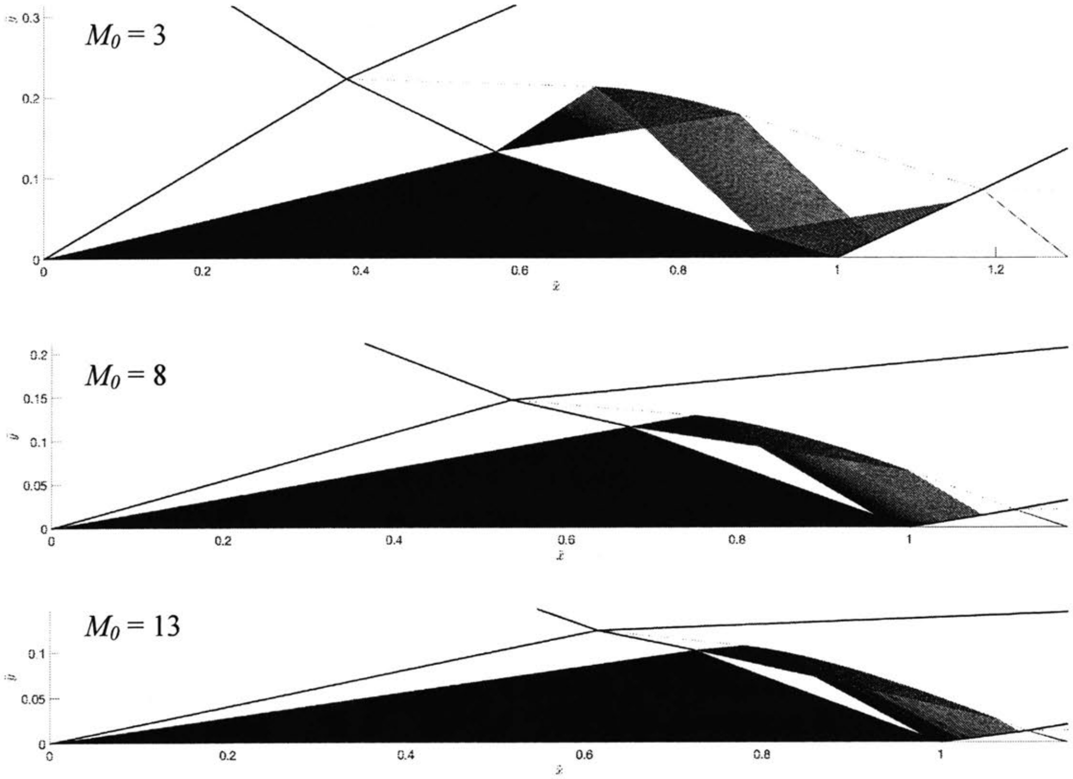

Figure 4-12: The flow fields for $M_0 = 3$ (top), $M_0 = 8$ (middle), and $M_0 = 13$ (bottom). For information on the various features depicted, refer to the caption for Figure 4-1.

Due to the differences in Mach number and plateau pressure between the three flows, it is difficult to make quantitative comparisons based on the visual information provided by Figure 4-12 alone. However, it is quite clear that as M_0 is increased, the flow field compresses in the vertical direction. This observation is consistent with expectation. As Mach number is increased, Mach angles in the flow field decrease, and characteristic curves in the flow align more closely with the streamlines. This phenomenon is clear from the expansion fan that originates at the apex of the displacement body. At larger values of M_0, the expansion waves align more closely with the slip line, which is parallel to the streamlines.

Another important observation, which is a consequence of the vertical compression of the flow field, regards the expansion fan located downstream of the reattachment point. This post-reattachment expansion fan originates at the intersection between the reattachment shock wave and the slip line, and it extends down to the wall where it is reflected (the reflection is not pictured in Figure 4-12). For larger values of M_0, the vertical compression of the flow field means that the slip line is located closer to the wall, and as a result the point where the post-reattachment expansion fan reflects off the wall is located closer to the reattachment point. The post-reattachment expansion fan has a significant effect on the surface pressure distribution at hypersonic Mach number, and this effect is discussed in more detail in the next section.

4.4 Hypersonic Flow Regime

As discussed in the previous section, the flow field over the displacement body exhibits several distinctive qualitative features when the upstream Mach number is within the regime of hypersonic flow, which is defined roughly as $M_0 > 5$. These features include the vertical compression of the flow field, and the movement upstream of the post-reattachment expansion fan as the value of M_0 is increased. Quantitatively, it was found that the non-dimensional overall pressure rise across the separation bubble exhibits a nearly quadratic dependence on upstream Mach number through the hypersonic flow regime. In order to further explore the effect of Mach number on the surface pressure distribution in the hypersonic flow regime, the surface pressure distributions at the three upstream Mach numbers previously investigated in Figure 4-12 ($M_0 = 3$, $M_0 = 8$, and $M_0 = 13$) are plotted in Figure 4-13. The incident shock wave strength is set such that $\Delta\theta_{incident} = -15°$, and the boundary layer is assumed to be turbulent.

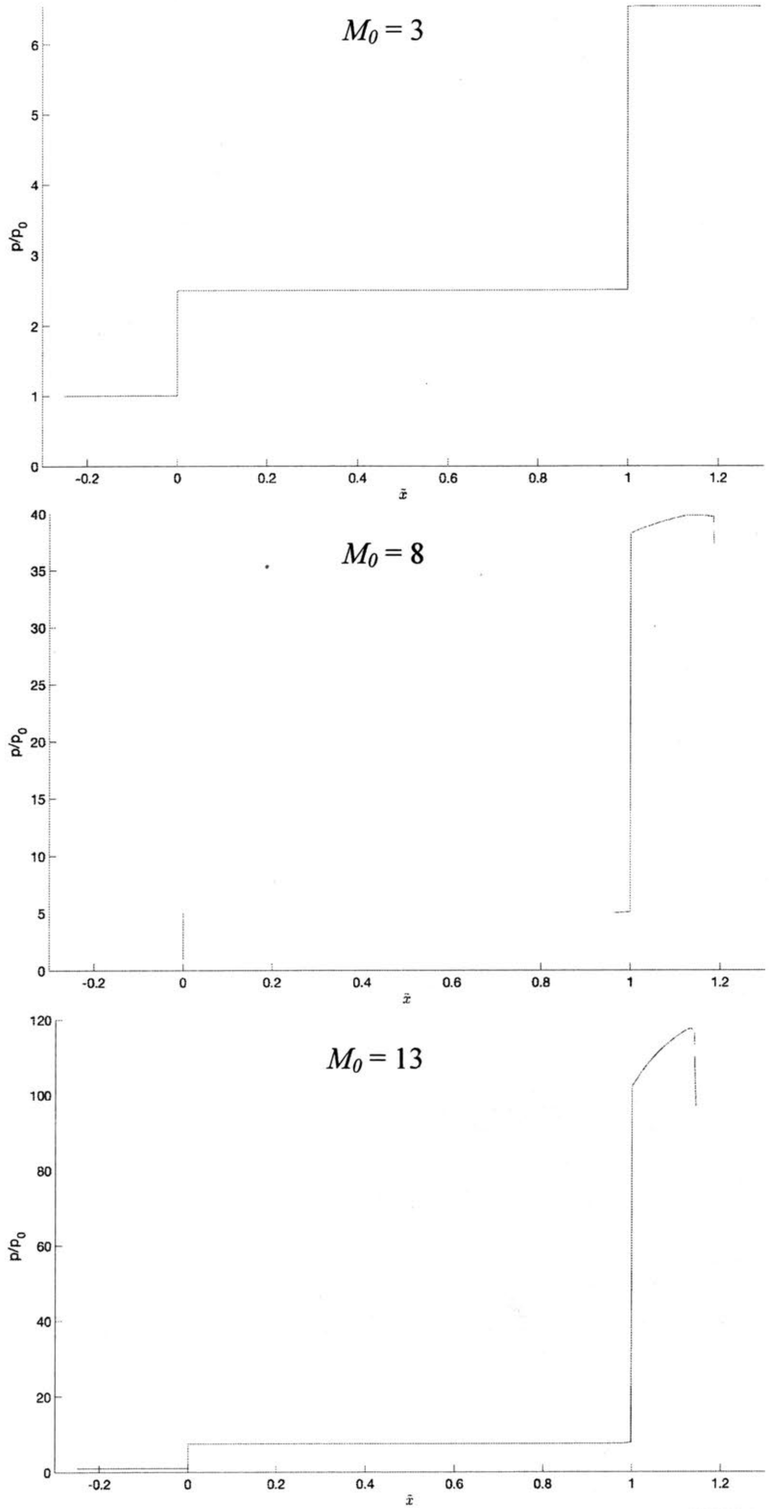

Figure 4-13: Surface pressure distributions for $M_0 = 3$ (top), $M_0 = 8$ (middle), and $M_0 = 13$ (bottom). The non-dimensional surface pressure, p/p_0, is plotted against the non-dimensional distance along the wall, $\tilde{x}$.

Before analyzing the plots of Figure 4-13 in detail, it is noted that the range of values displayed on the vertical axis differs significantly between the three plots. The non-dimensional surface pressure along and downstream of the displacement body are clearly larger for larger values of M_0. This observation is consistent with the Mach number dependence of plateau and reattachment pressures discussed in the previous section.

Attention is now drawn to the effect of the post-reattachment expansion fan on the surface pressure distribution. The post-reattachment expansion fan manifests itself in the surface pressure distribution plots of Figure 4-13 as a pressure drop downstream of the reattachment point. The nature of the pressure drop is largely dependent on Mach number. For $M_0 = 3$, the pressure drop is nearly non-existent; as a fraction of p_0, the magnitude of the pressure drop is 0.0118, or as a percentage of p_r, 0.18 %. The pressure drop is more evident for $M_0 = 8$ (2.46, or 6.4 %), and even more clearly so for $M_0 = 13$ (20.6, or 20.2%). The magnitude of the pressure drop increases in both absolute and relative terms as M_0 is increased. And, consistent with the upstream movement of the post-reattachment expansion fan in Figure 4-12, the location of the pressure drop moves upstream with increasing Mach number. The minimum pressure occurs at $\tilde{x} = 1.29$ for $M_0 = 3$, $\tilde{x} = 1.19$ for $M_0 = 8$, and $\tilde{x} = 1.14$ for $M_0 = 13$.

The existence of a pressure drop downstream of reattachment in hypersonic strong-interaction SBLI is well known. However, several authors [9, 17] erroneously attribute the hypersonic pressure drop to the interaction between the separation and reattachment shock waves. From Figure 4-12, it is quite clear that this explanation is incorrect. The separation and reattachment shock waves do not intersect until much farther downstream than the wall location where the pressure drop is observed. The origin of the post-reattachment expansion fan, and thus the origin of the pressure drop, is in fact the intersection of the reattachment shock wave and the slip line.

Chapter 5

Conclusion

In this thesis, an efficient computational tool for strong-interaction SBLI has been developed and evaluated. The computational tool is based on the displacement-body model suggested by Delery & Marvin [9], which replaces the viscous flow through the interaction region with the inviscid flow over a two-sided ramp at constant surface pressure. The geometry of the displacement body is related to user-specified upstream conditions via free-interaction theory. Finally, a shock-fitting method of characteristics is implemented to numerically compute the flow over the displacement body.

It has been shown that the displacement-body model accurately predicts flow quantities at the reattachment location, both for laminar boundary layers at supersonic Mach number, and for turbulent boundary layers at hypersonic Mach number. In the former case, where conditions are consistent with the assumption of constant pressure through the separation bubble, predicted quantities fall within 3-4 % of experimentally measured values. In the latter case, which strains the model's assumption of constant pressure inside the separation bubble, accuracy is still shown to be 7-11 %.

Several important physical insights are also gleaned from the model. Approximate power-law relationships between upstream conditions and the reattachment pressure are developed for turbulent boundary layers. One such relationship, valid for $M_0 = 5$ and the approximate range $5 < \left(\frac{p_2}{p_1}\right)_{incident} < 9.5$, is found to be $\frac{p_r}{p_0} \sim \left(\frac{p_2}{p_1}\right)_{incident}^{1.3}$. Another, valid for $\Delta\theta_{incident} = -20°$ and the range $3 < M_0 < 15$, is found to be $\frac{p_r}{p_0} \sim M_0^{1.9}$. Furthermore, the well-known phenomenon of a hypersonic pressure drop downstream of reattachment is quantified, and it is shown that the pressure drop results from the intersection of the reattachment shock wave and the slip line.

The efficient computational tool for strong-interaction SBLI that is presented in this thesis has a number of potential applications. It may be used as a preliminary design tool to predict inviscid quantities that result from strong-interaction SBLI. These inviscid quantities include aerodynamic loads exerted through surface pressure forces on the physical body, as well as temperatures in the outer flow field. Additionally, the tool can be used as a preliminary "sanity check" for new CFD approaches, prior to vigorous verification of results with more expensive DNS or experiments.

Future work could address some of the limitations of the displacement-body model. One concept that might be explored is an alternative shape and pressure distribution for the displacement body. As shown in Chapter 4, the modeling error associated with the assumption of constant pressure over the displacement body can become significant for weaker incident shock waves. An even better approach would be to incorporate a viscous boundary layer solver, one that solves the boundary layer equations either in partial differential or integral form. Such a solver, coupled with an inviscid outer flow solver such as that presented in this thesis, could predict viscous quantities as well, including skin friction, heat transfer, and length scales within the interaction region.

Appendix

MATLAB Code

This appendix contains the MATLAB code that was written in order to compute the flow field around the displacement body. The main program appears below. It is followed by supporting .m files.

```
% Before running the code, the user should enter desired values in the
% "user-defined parameters" section. The code calls a number functions that
% are contained in separate .m files. These files should be contained in
% the current folder before running the code. For information on the
% storage of calculated quantities, see notes within the main code below
% and within the supporting .m files.

% Subscripts:

% 1 - upstream of interaction
% 2 - just downstream of separation shock
% 3 - just downstream of incident shock
% 4 - downstream of deflected incident shock, upstream of first expansion
%       fan
% 4' - downstream of first expansion fan, upstream of reflected compression
%       waves
% 4'' - at wall, just upstream of reattachment shock
% 5 - downstream of deflected separation shock, upstream of first expansion
%       fan
% 5' - downstream of expansion fan, above slipline, upstream of
%       reattachment shock
% 6 - downstream of reattachment shock, below slipline

%% User-defined parameters

Re0 = 6e5;
Lsep = 1;
LoT = 2; % set LoT = 1 for laminar BL, = 2 for turbulent BL
N = 100; % number of finite expansion waves to approximate expansion fan
NN = 20; % number of " " fan downstream of reattachment
kk = 5; % number of iterations to compute point 3 in rotational MOC
ll = 5; % number of iterations for corrector algorithm in rotational MOC
R = 287;
gamma = 1.4;

M_1 = 8.6;
beta_c1 = pi - 19.8 * pi / 180;
theta_1 = 0;
T_1 = 273;
```

```
p_1 = 101300;

%% Derived parameters

c_p = gamma / (gamma-1) * R;

%% Region 1 & C1

r_1 = p_1 / R / T_1;

beta1_theta1_M1_gamma = [beta_c1, theta_1, M_1, gamma];

%% Region 3

theta_3 = theta_1 + atan(tbm_rhs(beta1_theta1_M1_gamma));
beta_theta1_thetas_M1_p1_r1_gamma = [beta_c1, theta_1, theta_3, M_1,...
    p_1, r_1, gamma];

M_3 = M_s(beta_theta1_thetas_M1_p1_r1_gamma);
p_3 = p_s(beta_theta1_thetas_M1_p1_r1_gamma);

r_3 = r_s(beta_theta1_thetas_M1_p1_r1_gamma);
T_3 = T_1 * p_3 * r_1 / p_1 / r_3;
sgen_c1 = c_p * log(T_3/T_1) - R * log(p_3/p_1);

%% Region 2 & C2
if LoT == 1
    Cf0 = sqrt(0.85) * 0.664 / Re0^(1/2);
    p_2 = p_1 * (1 + 1.47 / 2 * gamma * M_1^2 * sqrt(2 * Cf0 /...
        sqrt(M_1^2 - 1)));
elseif LoT == 2
    p_2 = p_1 * (1 + 0.5 * M_1);
else
    disp('Error: check value of LoT')
    return
end

if p_3 < p_2
    disp('Error: shock c1 not strong enough to cause separation')
    return
end

beta_c2 = fsolve(@(beta) p_s([beta, theta_1, 0, M_1, p_1, r_1, gamma])/p_2 -...
    1, asin(1/M_1));

if ~(asin(1/M_1)< beta_c2 && beta_c2 < pi/2)
    disp('Error computing shock angle c2: check that M1*sin(betac2) > 1')
    return
end

beta2_theta1_M1_gamma = [beta_c2, theta_1, M_1, gamma];
theta_2 = theta_1 + atan(tbm_rhs(beta2_theta1_M1_gamma));
beta_theta1_thetas_M1_p1_r1_gamma = [beta_c2, theta_1, theta_2, M_1,...
    p_1, r_1, gamma];

M_2 = M_s(beta_theta1_thetas_M1_p1_r1_gamma);
r_2 = r_s(beta_theta1_thetas_M1_p1_r1_gamma);
T_2 = T_1 * p_2 * r_1 / p_1 / r_2;
sgen_c2 = c_p * log(T_2/T_1) - R * log(p_2/p_1);

%% Regions 4,5 & C3,C4
```

```
[beta_c4c3, res, flag] = fsolve(@(betac4c3) type1(betac4c3, [theta_2, theta_3],
...
    [M_2, M_3], [p_2, p_3], gamma), [- asin(1/M_2) + theta_2,...
    asin(1/M_3) + theta_3]);

if flag <= 0
    disp('Error: Type II interference')
    return
end

beta_c4 = beta_c4c3(1);
beta_c3 = beta_c4c3(2);

beta4_theta2_M2_gamma = [beta_c4, theta_2, M_2, gamma];
theta_4 = theta_2 + atan(tbm_rhs(beta4_theta2_M2_gamma));
beta3_theta3_M3_gamma = [beta_c3, theta_3, M_3, gamma];
theta_5 = theta_3 + atan(tbm_rhs(beta3_theta3_M3_gamma));

beta_theta1_thetas_M1_p1_r1_gamma = [beta_c4, theta_2, theta_4, M_2,...
    p_2, r_2, gamma];
M_4 = M_s(beta_theta1_thetas_M1_p1_r1_gamma);

if M_4 < 1
    disp('Error: Region 4 is subsonic')
    return
end

p_4 = p_s(beta_theta1_thetas_M1_p1_r1_gamma);
r_4 = r_s(beta_theta1_thetas_M1_p1_r1_gamma);
T_4 = T_2 * p_4 * r_2 / p_2 / r_4;
sgen_c4 = c_p * log(T_4/T_2) - R * log(p_4/p_2);

beta_theta1_thetas_M1_p1_r1_gamma = [beta_c3, theta_3, theta_5, M_3,...
    p_3, r_3, gamma];
M_5 = M_s(beta_theta1_thetas_M1_p1_r1_gamma);
p_5 = p_s(beta_theta1_thetas_M1_p1_r1_gamma);
r_5 = r_s(beta_theta1_thetas_M1_p1_r1_gamma);
T_5 = T_3 * p_5 * r_3 / p_3 / r_5;
sgen_c3 = c_p * log(T_5/T_3) - R * log(p_5/p_3);

%% Region 4'

M_4p = fsolve(@(M) p_exp([M_4, M, p_4, gamma])/p_2 - 1, M_2);
p_4p = p_exp([M_4, M_4p, p_4, gamma]);

if (p_4p - p_2) / p_2 > 10^-4
    disp('Error computing conditions in 4p')
    return
end

T_4p = T_exp([M_4, M_4p, T_4, gamma]);
r_4p = r_4 * T_4 * p_4p / p_4 / T_4p;
theta_4p = theta_4 + prmey([M_4, gamma]) - prmey([M_4p, gamma]);
beta_ef1 = theta_4 + asin(1/M_4);
beta_ef2 = theta_4p + asin(1/M_4p);

%% Expansion wave slipline interaction (4' and 5')

L1 = Lsep * tan(theta_4p) / (tan(theta_4p) - tan(theta_2));
h = L1 * tan(theta_2);
Hx = (h - tan(beta_c4) * L1) / (tan(beta_c2) - tan(beta_c4));
```

```
Hy = Hx * tan(beta_c2);

th_M_p_T = expw_slipl(theta_4, p_4, M_4, T_4, M_5, T_5,...
    theta_4p, p_4p, M_4p, gamma, N);
xy_char = characteristics(theta_4, M_4, M_5, theta_4p, M_4p,...
    th_M_p_T, N, h, Hy, Hx, L1, Lsep);

%% Reflection of compression waves off ramp (4'')

if xy_char(end,1) ~= L1
    th_M_p_T_2 = compw_ramp(th_M_p_T, theta_4, theta_4p, gamma, N);
    xy_char_2 = characteristics2(theta_4p, M_4p, th_M_p_T, ...
        xy_char, th_M_p_T_2, N, Lsep);
    xy_char_2 = characteristics2simple2(th_M_p_T, ...
    th_M_p_T_2, xy_char, xy_char_2, N, Lsep);

    i = 0;

    for j = 1:N/2
        if xy_char(end,2*j-1) ~= L1
            i = i+1;
        else
            break;
        end
    end

    k = i;
    b = zeros(N,1);
    b(1) = 1;
    for ii = 2:N
        b(ii) = b(ii-1) + ii;
    end

    theta_4pp = th_M_p_T_2(b(i), 1);
    M_4pp = th_M_p_T_2(b(i), 2);
    p_4pp = th_M_p_T_2(b(i), 3);
    T_4pp = th_M_p_T_2(b(i), 4);
    r_4pp = p_4pp / R / T_4pp;
else
    xy_char_2 = characteristics2simple(theta_4p, M_4p, ...
    th_M_p_T, xy_char, N);
    theta_4pp = theta_4p;
    M_4pp = M_4p;
    p_4pp = p_4p;
    T_4pp = T_4p;
    r_4pp = r_4p;
end

%% Moeckel computation of shock c5

theta_6 = zeros(N+1,1);
M_6 = zeros(N+1,1);
Mu = [M_4pp; zeros(N,1)];
p_6 = zeros(N+1,1);
r_6 = zeros(N+1,1);
T_6 = zeros(N+1,1);

beta_c5 = zeros(N+1,1);
xc5 = [Lsep; zeros(N+1,1)];
yc5 = zeros(N+2,1);
sgen_c5 = zeros(N+1,1);
```

```
theta_6(1) = theta_1;

[beta_c5(1), res, flag] = fsolve(@(beta) tbm_rhs([beta, theta_4pp, M_4pp, ...
    gamma]) / tan(theta_6(1) - theta_4pp) - 1, asin(1/M_4pp));

if flag <= 0
    disp('Error: Reattachment shock is detached')
    return
end

beta_theta1_thetas_M1_p1_r1_gamma = [beta_c5(1), theta_4pp, theta_6(1),
M_4pp,...
    p_4pp, r_4pp, gamma];

M_6(1) = M_s(beta_theta1_thetas_M1_p1_r1_gamma);
p_6(1) = p_s(beta_theta1_thetas_M1_p1_r1_gamma);
r_6(1) = r_s(beta_theta1_thetas_M1_p1_r1_gamma);
T_6(1) = p_6(1) / r_6(1) / R;
sgen_c5(1) = c_p * log(T_6(1)/T_4pp) - R * log(p_6(1)/p_4pp);

if xy_char(end,1) ~= L1
    r = zeros(N,1);
    c = zeros(N,1);
end

for l = 2:N+1
    if xy_char(end,1) ~= L1
        thu_Mu_pu_Tu_xip1_yip1_r_c = upstream_c5(beta_c5(l-1), xc5(l-1), ...
            yc5(l-1), xy_char_2, th_M_p_T, th_M_p_T_2, N);
    else
        thu_Mu_pu_Tu_xip1_yip1_r_c = upstream_c5_noreflec(beta_c5(l-1), ...
            xc5(l-1), yc5(l-1), xy_char_2, th_M_p_T, N);
    end

    thu = thu_Mu_pu_Tu_xip1_yip1_r_c(1);
    Mu(l) = thu_Mu_pu_Tu_xip1_yip1_r_c(2);
    pu = thu_Mu_pu_Tu_xip1_yip1_r_c(3);
    Tu = thu_Mu_pu_Tu_xip1_yip1_r_c(4);
    ru = pu / R / Tu;
    xc5(l) = thu_Mu_pu_Tu_xip1_yip1_r_c(5);
    yc5(l) = thu_Mu_pu_Tu_xip1_yip1_r_c(6);

    if xy_char(end,1) ~= L1
        r(l-1) = thu_Mu_pu_Tu_xip1_yip1_r_c(7);
        c(l-1) = thu_Mu_pu_Tu_xip1_yip1_r_c(8);
        beta_c5(l) = fsolve(@(beta) moeckel(beta, gamma, Mu(l-1), Mu(l), ...
            beta_c5(l-1), r(l-1), c(l-1)), beta_c5(l-1));
    else
        beta_c5(l) = fsolve(@(beta) moeckel(beta, gamma, Mu(l-1), Mu(l), ...
            beta_c5(l-1), 1, 2), beta_c5(l-1));
    end

    beta1_theta1_M1_gamma = [beta_c5(l), thu, Mu(l), gamma];
    theta_6(l) = thu + atan(tbm_rhs(beta1_theta1_M1_gamma));

    beta_theta1_thetas_M1_p1_r1_gamma = [beta_c5(l), thu, theta_6(l), Mu(l),...
    pu, ru, gamma];

    M_6(l) = M_s(beta_theta1_thetas_M1_p1_r1_gamma);
    p_6(l) = p_s(beta_theta1_thetas_M1_p1_r1_gamma);
    r_6(l) = r_s(beta_theta1_thetas_M1_p1_r1_gamma);
    T_6(l) = p_6(l) / r_6(l) / R;
    sgen_c5(1) = c_p * log(T_6(1)/Tu) - R * log(p_6(1)/pu);
end

% Compute last xc5 yc5, which is intersection with slip line

xyc5_end = intersection(tan(th_M_p_T(end,1)), tan(beta_c5(end)), ...
    xy_char(end-1,end-1), xy_char(end-1,end), ...
    xc5(end-1), yc5(end-1));
xc5(end) = xyc5_end(1);
yc5(end) = xyc5_end(2);

% Truncate results in case c5 extends beyond slip line

xc5 = xc5(xc5 < xc5(end));
q = length(xc5);
yc5 = yc5(1:q);

theta_6 = theta_6(1:q);
M_6 = M_6(1:q);
Mu = Mu(1:q);
p_6 = p_6(1:q);
r_6 = r_6(1:q);
T_6 = T_6(1:q);
beta_c5 = beta_c5(1:q);
sgen_c5 = sgen_c5(1:q);

% Re-calculate intersection with truncated results

xyc5_end = intersection(tan(th_M_p_T(end,1)), tan(beta_c5(end)), ...
    xy_char(end-1,end-1), xy_char(end-1,end), ...
    xc5(end), yc5(end));
xc5 = [xc5; xyc5_end(1)];
yc5 = [yc5; xyc5_end(2)];

%% Interaction of c5 and slip line

theta_t = th_M_p_T(N*(N+1)/2 + 2*N - 1, 1);
M_t = th_M_p_T(N*(N+1)/2 + 2*N - 1, 2);
p_t = th_M_p_T(N*(N+1)/2 + 2*N - 1, 3);

theta_l = th_M_p_T(N*(N+1)/2 + N - 1, 1);
M_l = th_M_p_T(N*(N+1)/2 + N - 1, 2);
p_l = th_M_p_T(N*(N+1)/2 + N - 1, 3);

theta_lp = theta_6(end);
M_lp = M_6(end);
p_lp = p_6(end);
T_lp = T_6(end);

[betac6_Mlpp, res, flag] = fsolve(@(betac6_Mlpp) c5_sl(theta_lp, M_lp, ...
    p_lp, theta_t, M_t, p_t, gamma, betac6_Mlpp), [beta_c5(end), M_l]);

if flag <= 0
    disp('Error in fsolve computing interaction of c5 with slipline')
    return
end

beta_c6 = betac6_Mlpp(1);
M_lpp = betac6_Mlpp(2);

if M_lpp < M_lp
```

```
    disp('Error in computing expansion fan reflection of shock c5 from
slipline')
    return
end

p_lpp = p_exp([M_lp, M_lpp, p_lp, gamma]);
theta_lpp = theta_lp + prmey([M_lpp, gamma]) - prmey([M_lp, gamma]);
T_lpp = T_exp([M_lp, M_lpp, T_lp, gamma]);

%% MOC for rotational flow behind curved shock c5

Np1 = q;

% Values along each C- characteristic are stored in a column vector. The
% column vectors are stored in a cell array. The first entry of each column
% vector corresponds to a point just behind shock c5, and the last entry
% corresponds to a wall point. The first column vector in the array is the
% reattachment point, and each subsequent column vector is located further
% downstream than the previous.

% Calculation prior to expansion fan

xrmoc = cell(1,Np1);
yrmoc = cell(1,Np1);
Mrmoc = cell(1,Np1);
thrmoc = cell(1,Np1);
prmoc = cell(1,Np1);
rrmoc = cell(1,Np1);
Trmoc = cell(1,Np1);

for jj = 1:Np1
    xtemp = zeros(jj,1);
    ytemp = zeros(jj,1);
    Mtemp = zeros(jj,1);
    thtemp = zeros(jj,1);
    ptemp = zeros(jj,1);
    rtemp = zeros(jj,1);
    Ttemp = zeros(jj,1);

    xtemp(1) = (xc5(jj) + xc5(jj+1)) / 2;
    ytemp(1) = (yc5(jj) + yc5(jj+1)) / 2;
    Mtemp(1) = M_6(jj);
    thtemp(1) = theta_6(jj);
    ptemp(1) = p_6(jj);
    rtemp(1) = r_6(jj);
    Ttemp(1) = T_6(jj);

    if jj ~= 1

        n = 2; % Index for current column
        nn = 2; % Index for previous column

        while nn <= length(xrmoc{jj-1})
            x0 = [xrmoc{jj-1}(nn); xtemp(n-1)];
            y0 = [yrmoc{jj-1}(nn); ytemp(n-1)];
            M0 = [Mrmoc{jj-1}(nn); Mtemp(n-1)];
            th0 = [thrmoc{jj-1}(nn); thtemp(n-1)];
            p0 = [prmoc{jj-1}(nn); ptemp(n-1)];
            r0 = [rrmoc{jj-1}(nn); rtemp(n-1)];
            x4_y4_M4_th4_p4_r4 = interior(x0, y0, M0, th0, p0, r0, ...
                gamma, kk, ll);
            x4 = x4_y4_M4_th4_p4_r4(1);
        y4 = x4_y4_M4_th4_p4_r4(2);
        M4 = x4_y4_M4_th4_p4_r4(3);
        th4 = x4_y4_M4_th4_p4_r4(4);
        p4 = x4_y4_M4_th4_p4_r4(5);
        r4 = x4_y4_M4_th4_p4_r4(6);

        if check_yc5(x4,y4, xc5, yc5, theta_lpp)
            xtemp = xtemp(1:length(xtemp) - 1);
            ytemp = ytemp(1:length(ytemp) - 1);
            Mtemp = Mtemp(1:length(Mtemp) - 1);
            thtemp = thtemp(1:length(thtemp) - 1);
            ptemp = ptemp(1:length(ptemp) - 1);
            rtemp = rtemp(1:length(rtemp) - 1);
            Ttemp = Ttemp(1:length(rtemp) - 1);
            n = n - 1;
        else
            xtemp(n) = x4;
            ytemp(n) = y4;
            Mtemp(n) = M4;
            thtemp(n) = th4;
            ptemp(n) = p4;
            rtemp(n) = r4;
            Ttemp (n) = p4 ./ r4 ./ R;
        end

        n = n + 1;
        nn = nn + 1;
    end

    x0 = [xrmoc{jj-1}(end); xtemp(n-1)];
    y0 = [yrmoc{jj-1}(end); ytemp(n-1)];
    M0 = [Mrmoc{jj-1}(end); Mtemp(n-1)];
    th0 = [thrmoc{jj-1}(end); thtemp(n-1)];
    p0 = [prmoc{jj-1}(end); ptemp(n-1)];
    r0 = [rrmoc{jj-1}(end); rtemp(n-1)];
    x4_y4_M4_th4_p4_r4 = directwall(x0, y0, M0, th0, p0, r0, ...
        gamma, ll);
    xtemp(n) = x4_y4_M4_th4_p4_r4(1);
    ytemp(n) = x4_y4_M4_th4_p4_r4(2);
    Mtemp(n) = x4_y4_M4_th4_p4_r4(3);
    thtemp(n) = x4_y4_M4_th4_p4_r4(4);
    ptemp(n) = x4_y4_M4_th4_p4_r4(5);
    rtemp(n) = x4_y4_M4_th4_p4_r4(6);
    Ttemp(n) = ptemp(n) ./ rtemp(n) ./ R;

    % Truncate further in case jj-1 was truncated
    xtemp = xtemp(1:n);
    ytemp = ytemp(1:n);
    Mtemp = Mtemp(1:n);
    thtemp = thtemp(1:n);
    ptemp = ptemp(1:n);
    rtemp = rtemp(1:n);
    Ttemp = Ttemp(1:n);
end

xrmoc{jj} = xtemp;
yrmoc{jj} = ytemp;
Mrmoc{jj} = Mtemp;
thrmoc{jj} = thtemp;
prmoc{jj} = ptemp;
rrmoc{jj} = rtemp;
Trmoc{jj} = Ttemp;
```

```
end

% Calculation through expansion fan

dth = (theta_lpp - theta_lp) / NN;

xrmoc = [xrmoc, cell(1,NN)];
yrmoc = [yrmoc, cell(1,NN)];
Mrmoc = [Mrmoc, cell(1,NN)];
thrmoc = [thrmoc, cell(1,NN)];
prmoc = [prmoc, cell(1,NN)];
rrmoc = [rrmoc, cell(1,NN)];

for jj = Np1+1:Np1+NN

    xtemp = zeros(jj,1);
    ytemp = zeros(jj,1);
    Mtemp = zeros(jj,1);
    thtemp = zeros(jj,1);
    ptemp = zeros(jj,1);
    rtemp = zeros(jj,1);
    Ttemp = zeros(jj,1);

    [MpdM, res, flag] = fsolve(@(M) (dth * (jj - Np1) + ...
        prmey([M_lp,gamma]) - prmey([M,gamma])) * 180 / pi, M_lpp);
    if flag <= 0
        disp('Error in fsolve computing discretized expansion fan at c5')
        return
    end
    ppdp = p_exp([M_lp, MpdM, p_lp, gamma]);
    TpdT = T_exp([M_lp, MpdM, T_lp, gamma]);
    rpdr = ppdp / R / TpdT;

    xtemp(1) = xc5(end);
    ytemp(1) = yc5(end);
    Mtemp(1) = MpdM;
    thtemp(1) = theta_lp + dth * (jj - Np1);
    ptemp(1) = ppdp;
    rtemp(1) = rpdr;
    Ttemp(1) = TpdT;

    n = 2; % Index for current column
    nn = 2; % Index for previous column

    while nn <= length(xrmoc{jj-1})
        x0 = [xrmoc{jj-1}(nn); xtemp(n-1)];
        y0 = [yrmoc{jj-1}(nn); ytemp(n-1)];
        M0 = [Mrmoc{jj-1}(nn); Mtemp(n-1)];
        th0 = [thrmoc{jj-1}(nn); thtemp(n-1)];
        p0 = [prmoc{jj-1}(nn); ptemp(n-1)];
        r0 = [rrmoc{jj-1}(nn); rtemp(n-1)];
        x4_y4_M4_th4_p4_r4 = interior(x0, y0, M0, th0, p0, r0, ...
            gamma, kk, ll);
        x4 = x4_y4_M4_th4_p4_r4(1);
        y4 = x4_y4_M4_th4_p4_r4(2);
        M4 = x4_y4_M4_th4_p4_r4(3);
        th4 = x4_y4_M4_th4_p4_r4(4);
        p4 = x4_y4_M4_th4_p4_r4(5);
        r4 = x4_y4_M4_th4_p4_r4(6);

        if check_yc5(x4,y4, xc5, yc5, theta_lpp)
            xtemp = xtemp(1:length(xtemp) - 1);
            ytemp = ytemp(1:length(ytemp) - 1);
            Mtemp = Mtemp(1:length(Mtemp) - 1);
            thtemp = thtemp(1:length(thtemp) - 1);
            ptemp = ptemp(1:length(ptemp) - 1);
            rtemp = rtemp(1:length(rtemp) - 1);
            Ttemp = Ttemp(1:length(rtemp) - 1);
            n = n - 1;
        else
            xtemp(n) = x4;
            ytemp(n) = y4;
            Mtemp(n) = M4;
            thtemp(n) = th4;
            ptemp(n) = p4;
            rtemp(n) = r4;
            Ttemp (n) = p4 ./ r4 ./ R;
        end

        n = n + 1;
        nn = nn + 1;
    end

    x0 = [xrmoc{jj-1}(end); xtemp(n-1)];
    y0 = [yrmoc{jj-1}(end); ytemp(n-1)];
    M0 = [Mrmoc{jj-1}(end); Mtemp(n-1)];
    th0 = [thrmoc{jj-1}(end); thtemp(n-1)];
    p0 = [prmoc{jj-1}(end); ptemp(n-1)];
    r0 = [rrmoc{jj-1}(end); rtemp(n-1)];
    x4_y4_M4_th4_p4_r4 = directwall(x0, y0, M0, th0, p0, r0, ...
        gamma, ll);
    xtemp(n) = x4_y4_M4_th4_p4_r4(1);
    ytemp(n) = x4_y4_M4_th4_p4_r4(2);
    Mtemp(n) = x4_y4_M4_th4_p4_r4(3);
    thtemp(n) = x4_y4_M4_th4_p4_r4(4);
    ptemp(n) = x4_y4_M4_th4_p4_r4(5);
    rtemp(n) = x4_y4_M4_th4_p4_r4(6);
    Ttemp(n) = ptemp(n) ./ rtemp(n) ./ R;

    % Truncate further in case jj-1 was truncated
    xtemp = xtemp(1:n);
    ytemp = ytemp(1:n);
    Mtemp = Mtemp(1:n);
    thtemp = thtemp(1:n);
    ptemp = ptemp(1:n);
    rtemp = rtemp(1:n);
    Ttemp = Ttemp(1:n);

    xrmoc{jj} = xtemp;
    yrmoc{jj} = ytemp;
    Mrmoc{jj} = Mtemp;
    thrmoc{jj} = thtemp;
    prmoc{jj} = ptemp;
    rrmoc{jj} = rtemp;
    Trmoc{jj} = Ttemp;

end
```

c5_sl.m

```
function res = c5_sl(th_lp, M_lp, p_lp, th_t, M_t, p_t, gamma, ...
    betac6_Mlpp)

    beta_c6 = betac6_Mlpp(1);
    M_lpp = betac6_Mlpp(2);

    th_lpp = th_lp + prmey([M_lpp, gamma]) - prmey([M_lp, gamma]);
    p_lpp = p_exp([M_lp, M_lpp, p_lp, gamma]);

    th_tp = th_t + atan(tbm_rhs([beta_c6, th_t, M_t, gamma]));
    p_tp = p_s([beta_c6, th_t, 0, M_t, p_t, 0, gamma]);

    res1 = (th_tp - th_lpp) * 180 / pi;
    res2 = (p_tp - p_lpp)/(p_tp + p_lpp);
    res = [res1; res2];
end
```

characteristics.m

```
function xy_char = characteristics(theta_i, M4, M5, theta4p, M4p,...
    th_M_p_T, N, h, Hy, Hx, L1, Lsep)

    % Output is a matrix of the x- and y- coordinates of the C+ and C-
    % characteristics below the slipline, as well as the slipline itself.
    % The first pair of columns corresponds to the x- and y-coordinates
    % (in that order) of the first finite expansion wave, and of the first
    % reflected compression wave. Each subsequent pair of columns
    % contains the coordinates of the next downstream finite
    % expansion wave and its reflected compression wave. The final pair of
    % columns contain the coordinates of the slip line.

    [n,~] = size(th_M_p_T);
    n = n + 3;
    th = [theta_i; th_M_p_T(1:N-1,1); theta4p; ...
        th_M_p_T(N:n-3-N,1); theta_i; th_M_p_T(n-2-N:end,1)];
    M = [M4; th_M_p_T(1:N-1,2); M4p; ...
        th_M_p_T(N:n-3-N,2); M5; th_M_p_T(n-2-N:end,2)];

    m = 2*N + 2 + N*(N-1)/2;
    dydx_sl = tan(th(m:end));

    th_ave_exp = zeros(N * (N+1) / 2, 1);
    M_ave_exp = zeros(N * (N+1) / 2, 1);
    th_ave_comp = zeros(N * (N+1) / 2, 1);
    M_ave_comp = zeros(N * (N+1) / 2, 1);
    mu_ave_exp = zeros(N * (N+1) / 2, 1);
    mu_ave_comp = zeros(N * (N+1) / 2, 1);
    dydx_exp = zeros(N * (N+1) / 2, 1);
    dydx_comp = zeros(N * (N+1) / 2, 1);
    xy_char = zeros(N+2, 2*N+2);
    xy_char(1, end-1:end) = [Hx - L1, Hy - h];

    i = 1;

    for j = 1:N % Compute first simple region & nonsimple region

        k = 1;

        th_ave_exp(i) = (th(j) + th(j+1)) / 2;
        M_ave_exp(i) = (M(j) + M(j+1)) / 2;
        mu_ave_exp(i) = asin(1/M_ave_exp(i));
        dydx_exp(i) = tan(th_ave_exp(i) + mu_ave_exp(i));
        th_ave_comp(i) = (th(j+1) + th(j+1+N)) / 2;
        M_ave_comp(i) = (M(j+1) + M(j+1+N)) / 2;
        mu_ave_comp(i) = asin(1/M_ave_comp(i));
        dydx_comp(i) = tan(th_ave_comp(i) - mu_ave_comp(i));

        if j == 1 % Initial slip line point given by input geometry
            xy_int = intersection(dydx_exp(1), dydx_sl(1), 0,...
                0, Hx - L1, Hy - h);
            xy_char(k+1, 2*j-1:2*j) = xy_int;
        else
            xy_int = intersection(dydx_exp(i), dydx_comp(i-j+1),...
                0, 0, xy_char(j, 2*k-1), xy_char(j, 2*k));
            xy_char(k+1, 2*j-1:2*j) = xy_int;
            xy_char(j+1, 2*k-1:2*k) = xy_int;
        end
        i = i + 1;
        k = k + 1;
        ii = j + N;

        while k <= j

            th_ave_exp(i) = (th(ii) + th(ii+1)) / 2;
            M_ave_exp(i) = (M(ii) + M(ii+1)) / 2;
            mu_ave_exp(i) = asin(1/M_ave_exp(i));
            dydx_exp(i) = tan(th_ave_exp(i) + mu_ave_exp(i));
            th_ave_comp(i) = (th(ii+1) + th(ii+1+N-k+1)) / 2;
            M_ave_comp(i) = (M(ii+1) + M(ii+1+N-k+1)) / 2;
            mu_ave_comp(i) = asin(1/M_ave_comp(i));
            dydx_comp(i) = tan(th_ave_comp(i) - mu_ave_comp(i));

            if j == k % Use slip line slope
                xy_int = intersection(dydx_exp(i), dydx_sl(j),...
                    xy_char(k, 2*j-1), xy_char(k, 2*j),...
                    xy_char(j, 2*k-3), xy_char(j, 2*k-2));
                xy_char(k+1, 2*j-1:2*j) = xy_int;
            else
                xy_int = intersection(dydx_exp(i), dydx_comp(i-j+1),...
                    xy_char(k, 2*j-1), xy_char(k, 2*j),...
                    xy_char(j, 2*k-1), xy_char(j, 2*k));
                xy_char(k+1, 2*j-1:2*j) = xy_int;
                xy_char(j+1, 2*k-1:2*k) = xy_int;
            end

            i = i + 1;
            k = k + 1;
            ii = ii + N - k + 2;

        end

        xy_char(j+1, end-1:end) = xy_char(j+1, 2*j-1:2*j);

    end

    % Compute second simple region (if compression waves reflect on ramp)

    xy_int = intersection(dydx_comp((N-1)*(N-2)/2 + N:end), tan(theta4p),...
        xy_char(end-1, 1:2:end-3)', xy_char(end-1, 2:2:end-2)', 0, 0);

    for i = 1:N
        if xy_int(i, 1)<= Lsep - L1
            xy_char(end, 2*i-1:2*i) = xy_int(i, :);
        else
            break;
        end
    end

    % Translation to match coordinate systems

    T = L1 * ones(N+2, 2*N+2);
    T(:,2:2:end) = h;
    xy_char = xy_char + T;

end
```

characteristics2.m

```
function xy_char_2 = characteristics2(theta4p, M4p,...
    th_M_p_T, xy_char, th_M_p_T_2, N, Lsep)

    a = zeros(N,1);
    b = zeros(N,1);
    a(1) = 1;
    b(1) = 2*N-1;
    for i = 2:N
        a(i) = a(i-1) + i - 1;
        b(i) = b(i-1) + N + 1 - i;
    end

    [n,~] = size(th_M_p_T_2);
    n = n + N + 1;
    th = zeros(n,1);
    M = zeros(n,1);
    th(1:3) = [theta4p; th_M_p_T(b(1), 1); th_M_p_T_2(1,1)];
    M(1:3) = [M4p; th_M_p_T(b(1), 2); th_M_p_T_2(1,2)];

    i = 4;
    j = 2;
    k = 2;

    while i <= n
        if ismember(j,a)
            th(i) = th_M_p_T(b(k), 1);
            M(i) = th_M_p_T(b(k), 2);
            i = i + 1;
            k = k + 1;
            th(i) = th_M_p_T_2(j, 1);
            M(i) = th_M_p_T_2(j, 2);
            i = i + 1;
            j = j + 1;
        else
            th(i) = th_M_p_T_2(j, 1);
            M(i) = th_M_p_T_2(j, 2);
            i = i + 1;
            j = j + 1;
        end
    end

    dydx_wall = tan(theta4p);

    th = flipud(th);
    M = flipud(M);

    th_ave_exp = zeros(N * (N+1) / 2, 1);
    M_ave_exp = zeros(N * (N+1) / 2, 1);
    th_ave_comp = zeros(N * (N+1) / 2, 1);
    M_ave_comp = zeros(N * (N+1) / 2, 1);
    mu_ave_exp = zeros(N * (N+1) / 2, 1);
    mu_ave_comp = zeros(N * (N+1) / 2, 1);
    dydx_exp = zeros(N * (N+1) / 2, 1);
    dydx_comp = zeros(N * (N+1) / 2, 1);
    xy_char_2 = zeros(N+1, 2*N);
    xy_char_2(1,:) = xy_char(end-1,1:end-2);

    i = 1;
```

```
for j = 1:N % Copied from characteristics.m. Note that "exp" is a
            % reflected comp wave, and "comp" is an incident comp wave.
            % Also note that order/indexing is flipped (code progresses
            % from downstream to upstream) inside the loop.

    k = 1;

    th_ave_exp(i) = (th(j) + th(j+1)) / 2;
    M_ave_exp(i) = (M(j) + M(j+1)) / 2;
    mu_ave_exp(i) = asin(1/M_ave_exp(i));
    dydx_exp(i) = tan(th_ave_exp(i) + mu_ave_exp(i));
    th_ave_comp(i) = (th(j+1) + th(j+1+N)) / 2;
    M_ave_comp(i) = (M(j+1) + M(j+1+N)) / 2;
    mu_ave_comp(i) = asin(1/M_ave_comp(i));
    dydx_comp(i) = tan(th_ave_comp(i) - mu_ave_comp(i));

    i = i + 1;
    k = k + 1;
    ii = j + N;

    while k <= j

        th_ave_exp(i) = (th(ii) + th(ii+1)) / 2;
        M_ave_exp(i) = (M(ii) + M(ii+1)) / 2;
        mu_ave_exp(i) = asin(1/M_ave_exp(i));
        dydx_exp(i) = tan(th_ave_exp(i) + mu_ave_exp(i));
        th_ave_comp(i) = (th(ii+1) + th(ii+1+N-k+1)) / 2;
        M_ave_comp(i) = (M(ii+1) + M(ii+1+N-k+1)) / 2;
        mu_ave_comp(i) = asin(1/M_ave_comp(i));
        dydx_comp(i) = tan(th_ave_comp(i) - mu_ave_comp(i));

        i = i + 1;
        k = k + 1;
        ii = ii + N - k + 2;

    end

end

dydx_exp = flipud(dydx_exp);
dydx_comp = flipud(dydx_comp);

i = 1;

for j = 1:N % Also copied, only used here to compute coordinates. Also
            % slightly modified, to account for flipping of ave value
            % vectors & to match indexing.
    k = 1;

    if j == 1 % Use first row of xy_char_2 AND ramp geometry
        xy_int = intersection(dydx_comp(1), dydx_wall, ...
            xy_char_2(1,1), xy_char_2(1,2), Lsep, 0);
        xy_char_2(j+1, 2*j-1:2*j) = xy_int;
    else
        xy_int = intersection(dydx_comp(i), dydx_wall, ...
            xy_char_2(j, 2*j-1), xy_char_2(j, 2*j), ...
            Lsep, 0);
        xy_char_2(j+1, 2*j-1:2*j) = xy_int;
    end

    i = i + 1;
    k = k + 1;
```

```
        while k <= N + 1 - j
                xy_int = intersection(dydx_comp(i), dydx_exp(i-1),...
                    xy_char_2(j, 2*(j+k-2)+1), xy_char_2(j, 2*(j+k-2)+2),...
                    xy_char_2(j+k-1, 2*(j-1)+1), xy_char_2(j+k-1, 2*(j-1)+2));
                xy_char_2(j+1, 2*(j+k-2)+1:2*(j+k-2)+2) = xy_int;
                xy_char_2(j+k, 2*(j-1)+1:2*(j-1)+2) = xy_int;

            i = i + 1;
            k = k + 1;
        end

    end

end
```

characteristics2simple.m

```
function xy_char_2 = characteristics2simple(theta_4p, M_4p, ...
    th_M_p_T, xy_char, N)
% Outputs x- and y- coordinates of the noncurved characteristics that
% extend past the first nonsimple region, in the case when there is not a
% second nonsimple region (i.e., the waves do not reflect off the ramp)

    xy_char_2 = zeros(N+2, 2*N);
    xy_char_2(1,:) = xy_char(end-1,1:end-2);

    M = zeros(N+1, 1);
    th = zeros(N+1, 1);
    M(1) = M_4p;
    th(1) = theta_4p;
    j = N-1+N;
    for i = 2:N+1
        M(i) = th_M_p_T(j,2);
        th(i) = th_M_p_T(j,1);
        j = j + N - (i-1);
    end

    th_ave = (th(1:end-1) + th(2:end)) ./ 2;
    M_ave = (M(1:end-1) + M(2:end)) ./ 2;
    mu_ave = asin(1./M_ave);
    dydx = tan(th_ave - mu_ave);

    for i = 2:2:2*N
        x0 = xy_char_2(1,i-1);
        y0 = xy_char_2(1,i);
        xy_char_2(:,i) = linspace(y0, 0, N+2)';
        xy_char_2(:,i-1) = (xy_char_2(:,i) - y0) ./ dydx(i/2) + x0;
    end

end
```

characteristics2simple2.m

```
function xy_char_2 = characteristics2simple2(th_M_p_T, ...
    th_M_p_T_2, xy_char, xy_char_2, N, Lsep)
% Appends one additional row to xy_char_2. The additional row extends each
% characteristic curve through the third simple region, up to an x value of
% 2 * Lsep.

    m = N * (N+1) / 2;
    th = th_M_p_T_2(m-N+1:m,1);
    M = th_M_p_T_2(m-N+1:m,2);
    th = [th_M_p_T(m+N-1, 1); th];
    M = [th_M_p_T(m+N-1, 2); M];
    th_ave = (th(1:end-1) + th(2:end)) ./ 2;
    M_ave = (M(1:end-1) + M(2:end)) ./ 2;
    mu_ave = asin(1./M_ave);

    dydx_char = tan(th_ave + mu_ave);
    x0_char = xy_char_2(end,1:2:end-1)';
    y0_char = xy_char_2(end,2:2:end)';

    lastrow_x = 2 * ones(N, 1);
    lastrow_y = y0_char + dydx_char .* (2*Lsep - x0_char);
    lastrow = [lastrow_x, lastrow_y];
    lastrow = reshape(lastrow', [2*N, 1]);
    xy_char_2 = [xy_char_2; lastrow'];

end
```

check_yc5.m

```
function yn = check_yc5(x, y, xc5, yc5, theta_lpp)

    % Returns true if (x,y) is located above shock c5, false otherwise.
    % Note that false will be returned if x is not in or greater than the
    % range of xc5.

    yn = false;

    if x >= xc5(end)

        yc5p = yc5(end) + (x - xc5(end)) * tan(theta_lpp);

        if y > yc5p
            yn = true;
        end

    else

        for i = 1:length(xc5)-1
            if xc5(i) <= x < xc5(i+1)
                yc5p = yc5(i) + (yc5(i+1) - yc5(i)) * (x - xc5(i)) ...
                    / (xc5(i+1) - xc5(i));
                if y > yc5p
                    yn = true;
                end
                break;
            end
        end

    end
end
```

compw_ramp.m

```
function th_M_p_T_2 = compw_ramp(th_M_p_T, theta4, theta4p, gamma, N)

% Computes theta, M, p, and T in the second nonsimple region assuming that
% the compression waves originating at the slip line reflect off the ramp.
% Columns of the output matrix correspond to, in order, theta, M, p, and T.
% Each row corresponds to a region enclosed by finite compression waves.
% The first row is the first region downstream of the first reflected
% compression wave, and adjacent to the wall. The second row is also just
% downstream of the first reflected wave, and adjacent to the first region.
% Each subsequent row is the next region along the C- characteristic, until
% the wall is reached. The process then repeats until the entire region is
% solved.

    dtheta = (theta4p - theta4) / N;

    k = N*(N-1)/2 + N;
    A = zeros(k, 4);

    a = zeros(N,1);
    b = zeros(N,1);
    a(1) = 1;
    b(1) = 1;
    for i = 2:N
        a(i) = a(i-1) + i - 1;
        b(i) = b(i-1) + i;
    end

    j = 1;
    m = N-1;

    % For i = 1

    m = m + N + 1 - j;

    M = fsolve(@(M) wall_region(th_M_p_T(m, 1),...
        th_M_p_T(m, 2), theta4p, M, gamma, dtheta), ...
        th_M_p_T(m, 2));
    A(1,1) = theta4p;
    A(1,2) = M;
    A(1,3) = p_exp([th_M_p_T(m, 2), A(1,2), ...
        th_M_p_T(m, 3), gamma]);
    A(1,4) = T_exp([th_M_p_T(m, 2), A(1,2), ...
        th_M_p_T(m, 4), gamma]);

    for i = 2:k
        if ismember(i,a) % Downstream of first reflected comp wave
            j = j + 1;
            m = m + N + 1 - j;

            th_M = fsolve(@(th_M) int_region(th_M_p_T(m, 1), ...
                th_M_p_T(m, 2), A(a(j-1), 1), A(a(j-1), 2), ...
                gamma, dtheta, th_M), [A(a(j-1), 1), A(a(j-1), 2)]);
            A(i,1) = th_M(1);
            A(i,2) = th_M(2);
            A(i,3) = p_exp([A(a(j-1), 2), A(i,2), ...
                A(a(j-1), 3), gamma]);
            A(i,4) = T_exp([A(a(j-1), 2), A(i,2), ...
                A(a(j-1), 4), gamma]);
        elseif ismember(i,b) % Adjacent to wall
            M = fsolve(@(M) wall_region(A(i-1, 1), A(i-1, 2), ...
                theta4p, M, gamma, dtheta), A(i-1, 2));
            A(i,1) = theta4p;
            A(i,2) = M;
            A(i,3) = p_exp([A(i-1,2), A(i,2), A(i-1,3), gamma]);
            A(i,4) = T_exp([A(i-1,2), A(i,2), A(i-1,4), gamma]);

        else % General interior region
            th_M = fsolve(@(th_M) int_region(A(i-1, 1), A(i-1, 2), ...
                A(i-j+1, 1), A(i-j+1, 2), gamma, dtheta, th_M),...
                [A(i-1,1), A(i-1,2)]);
            A(i,1) = th_M(1);
            A(i,2) = th_M(2);
            A(i,3) = p_exp([A(i-1,2), A(i,2), A(i-1,3), gamma]);
            A(i,4) = T_exp([A(i-1,2), A(i,2), A(i-1,4), gamma]);

        end
    end

    th_M_p_T_2 = A;

end

function res = wall_region(thm, Mm, th4p, M, gamma, dtheta)

    res = (thm + prmey([Mm,gamma]) - th4p - prmey([M,gamma])) /...
        dtheta;

end

function res = int_region(thm, Mm, thp, Mp, gamma, dtheta, th_M)

    theta = th_M(1);
    M = th_M(2);

    res1 = (theta - prmey([M,gamma]) + prmey([Mp,gamma]) - thp)...
        / dtheta;
    res2 = (theta + prmey([M,gamma]) - prmey([Mm,gamma]) - thm)...
        / dtheta;
    res = [res1; res2];

end
```

directwall.m

```
function x4_y4_M4_th4_p4_r4 = directwall(x0, y0, M0, th0, p0, r0, ...
    gamma, l)
% Note: for constant theta_wall. Function is nearly identical to
% interior.m, except that the initial data "line" is only two points: point
% 3 (known point on the wall) and point 2 (known point above the wall).
% Since theta_wall is assumed to be constant, the first entry of input th0
% (which corresponds to point 3) is taken to be the wall angle.

    n = length(x0);

    x3 = x0(1:end-1);
    x2 = x0(2:end);
    y3 = y0(1:end-1);
    y2 = y0(2:end);
    M3 = M0(1:end-1);
    M2 = M0(2:end);
    mu3 = asin(1./M3);
    mu2 = asin(1./M2);
    th3 = th0(1:end-1);
    th4 = th3;
    th2 = th0(2:end);
    p3 = p0(1:end-1);
    p2 = p0(2:end);
    r3 = r0(1:end-1);
    r2 = r0(2:end);
    dx0 = x0(2:end) - x0(1:end-1);
    dy0 = y0(2:end) - y0(1:end-1);

    % Initial estimate for (x4, y4)
    A1 = [diag(-tan(th3)), eye(n-1); diag(-tan(th2-mu2)), eye(n-1)];
    B1 = [y3 - tan(th3) .* x3; y2 - tan(th2-mu2) .* x2];
    x4y4 = A1\B1;
    x4 = x4y4(1:n-1);
    y4 = x4y4(n:end);

    % Initial calculation of properties at 4
    R2 = gamma .* M2.^2 ./ sqrt(M2.^2 - 1);
    lnp4 = log(p2) + R2 .* (th4-th2);
    p4 = exp(lnp4);
    M4 = sqrt(2./(gamma-1) .* ((1 + (gamma-1)/2 .* M3.^2) ./ ...
        (p4./p3).^((gamma-1)/gamma) - 1));
    r4 = (p4./p3).^(1/gamma) .* r3;

    % Apply corrector algorithm (l iterations)
    for k2 = 1:l
        % (x4, y4)
        A1 = [diag(-tan(th3)), eye(n-1); ...
            diag(-tan((th2+th4)./2 - asin(2./(M2+M4)))), eye(n-1)];
        B1 = [y3 - tan(th3) .* x3; y2 - ...
            tan((th2+th4)./2 - asin(2./(M2+M4))) .* x2];
        x4y4 = A1\B1;
        x4 = x4y4(1:n-1);
        y4 = x4y4(n:end);

        % Improved calculation of properties at 4
        R2 = gamma .* ((M2+M4)./2).^2 ./ sqrt(((M2+M4)./2).^2 - 1);
        lnp4 = log(p2) + R2 .* (th4-th2);
        p4 = exp(lnp4);
        M4 = sqrt(2./(gamma-1) .* ((1 + (gamma-1)/2 .* M3.^2) ./ ...
            (p4./p3).^((gamma-1)/gamma) - 1));
        r4 = (p4./p3).^(1/gamma) .* r3;

    end

    x4_y4_M4_th4_p4_r4 = [x4, y4, M4, th4, p4, r4];

end
```

expsol.m

```
function res = expsol(M4p_M5p, M4, M5, th4, th5, p4, p5, gamma)

    M4p = M4p_M5p(1);
    M5p = M4p_M5p(2);

    th4p = th4 + prmey([M4, gamma]) - prmey([M4p, gamma]);
    th5p = th5 + prmey([M5, gamma]) - prmey([M5p, gamma]);
    res1 = (th5p - th4p) * 180 / pi;

    p4p = p_exp([M4, M4p, p4, gamma]);
    p5p = p_exp([M5, M5p, p5, gamma]);
    res2 = (p5p - p4p) / (p5p + p4p);

    res = [res1; res2];

end
```

expw_slipl.m

```
function th_M_p_T = expw_slipl(theta_i, p_i, M4, T4, M5, T5,...
    theta4p, p4p, M4p, gamma, N)

    % Outputs matrix of four column vectors, each corresponding to (in
    % order) theta, M, p, and T. The first row corresponds
    % to the region downstream of the first finite expansion wave
    % originating at the corner, and below all reflected compression waves.
    % Each subsequent row corresponds to the region downstream of the
    % next finite expansion wave, but still below all reflected waves. Once
    % the region just upstream of 4' is reached, and its state solved for,
    % the next row of the matrix skips to the region just downstream of the
    % first reflected wave, and adjacent to the slip line. The process is
    % again marched downstream, without crossing reflected waves, until the
    % region adjacent to 4' is reached. The process repeats until the
    % final downstream region adjacent to the slipline is reached. The
    % subsequent rows of the ouput matrix correspond to the regions above
    % the slipline, again ordered from upstream to downstream. The final row of
    % the matrix corresponds to 5'. Note that the known regions 4, 4', and 5
    % do not appear in the output matrix.

    dtheta = (theta4p - theta_i) / N;
    dp = (p4p - p_i) / N;
    k = N * (N+1) / 2 + 2 * (N-1);
    m = N * (N+1) / 2 + N;

    % Calculate Region 1
    A = zeros(k + 1, 4);
    A(1,1) = theta_i + dtheta;
    A(1,2) = fsolve(@(M) (prmey([M, gamma]) - prmey([M4, gamma]) +...
        dtheta) / dtheta, M4);
    A(1,3) = p_exp([M4, A(1,2), p_i, gamma]);
    A(1,4) = T_exp([M4, A(1,2), T4, gamma]);

    % Simple Region: Calculate regions 2 through N-1
    for i = 2:N-1
        A(i,1) = A(i-1,1) + dtheta;
        A(i,2) = fsolve(@(M) (prmey([M, gamma]) - prmey([A(i-1,2), gamma]) +...
            dtheta) / dtheta, A(i-1,2));
        A(i,3) = p_exp([A(i-1,2), A(i,2), A(i-1,3), gamma]);
        A(i,4) = T_exp([A(i-1,2), A(i,2), A(i-1,4), gamma]);
    end

    a = zeros(N,1);
    a(1) = N;
    for i = 2:N
        a(i) = a(i-1) + N - (i-2);
    end

    j = 1;

    % Nonsimple Region: Calculate remaining regions
    for i = N:m-1
        if ismember(i,a) % Check for adjacency to slip line
            j = j + 1;
            if i == N % First unknown adjacent region--depends on 5
                lu = fsolve(@(lu) sl_region(A(1,1), A(1,2),...
                    A(1,3), gamma, theta_i, M5, p_i,...
                    dtheta, dp, lu), [A(1,1), A(1,2), ...
                    theta_i, M5]);
                A(i,1) = lu(1);
                A(i,2) = lu(2);
                A(i,3) = p_exp([A(1,2),A(i,2),A(1,3),gamma]);
                A(i,4) = T_exp([A(1,2),A(i,2),A(1,4),gamma]);
                A(m,1) = lu(3);
                A(m,2) = lu(4);
                A(m,3) = p_exp([M5,A(m,2),p_i,gamma]);
                A(m,4) = T_exp([M5,A(m,2),T5,gamma]);
            else % Subsequent regions adjecent to slipline--regular indexing
                lu = fsolve(@(lu) sl_region(A(i-N+j-2,1), A(i-N+j-2,2),...
                    A(i-N+j-2,3), gamma, A(m+j-3,1), A(m+j-3,2), A(m+j-3,3),...
                    dtheta, dp, lu), [A(i-N+j-2,1), A(i-N+j-2,2), ...
                    A(m+j-3,1), A(m+j-3,2)]);
                A(i,1) = lu(1);
                A(i,2) = lu(2);
                A(i,3) = p_exp([A(i-N+j-2,2),A(i,2),A(i-N+j-2,3),gamma]);
                A(i,4) = T_exp([A(i-N+j-2,2),A(i,2),A(i-N+j-2,4),gamma]);
                A(m+j-2,1) = lu(3);
                A(m+j-2,2) = lu(4);
                A(m+j-2,3) = p_exp([A(m+j-3,2),A(m+j-2,2),A(m+j-3,3),gamma]);
                A(m+j-2,4) = T_exp([A(m+j-3,2),A(m+j-2,2),A(m+j-3,4),gamma]);
            end

        elseif i == 2*N-1 % Region above first reflected wave & adjacent to 4'
            th_M = fsolve(@(th_M) int_region(A(i-1,1), A(i-1,2),...
                theta4p, M4p, gamma, dtheta, th_M),...
                [A(i-1,1), A(i-1,2)]);
            A(i,1) = th_M(1);
            A(i,2) = th_M(2);
            A(i,3) = p_exp([A(i-1,2), A(i,2), A(i-1,3), gamma]);
            A(i,4) = T_exp([A(i-1,2), A(i,2), A(i-1,4), gamma]);

        else
            if j == 2 % Special indexing for region above first reflected wave
                th_M = fsolve(@(th_M) int_region(A(i-1,1), A(i-1,2),...
                    A(i-N+1,1), A(i-N+1,2), gamma, dtheta, th_M),...
                    [A(i-1,1), A(i-1,2)]);
            else % Regular indexing
                th_M = fsolve(@(th_M) int_region(A(i-1,1), A(i-1,2),...
                    A(i-N+j-2,1), A(i-N+j-2,2), gamma, dtheta, th_M),...
                    [A(i-1,1), A(i-1,2)]);
            end
            A(i,1) = th_M(1);
            A(i,2) = th_M(2);
            A(i,3) = p_exp([A(i-1,2), A(i,2), A(i-1,3), gamma]);
            A(i,4) = T_exp([A(i-1,2), A(i,2), A(i-1,4), gamma]);
        end
    end

    th_M_p_T = A;

end

function res = sl_region(thcompl, Mcompl, pcompl, gamma, thexpu, Mexpu,
pexpu,...
    dtheta, dp, thl_Ml_thu_Mu)

    thetal = thl_Ml_thu_Mu(1);
    Ml = thl_Ml_thu_Mu(2);
    thetau = thl_Ml_thu_Mu(3);
    Mu = thl_Ml_thu_Mu(4);

    res1 = (thetal - prmey([Ml,gamma]) + prmey([Mcompl,gamma]) - thcompl)...
        / dtheta;
    res2 = (thetau + prmey([Mu,gamma]) - prmey([Mexpu,gamma]) - thexpu)...
        / dtheta;
    res3 = (p_exp([Mcompl,Ml,pcompl,gamma]) - p_exp([Mexpu,Mu,pexpu,gamma]))...
        / dp;
    res4 = (thetau - thetal) / dtheta;
    res = [res1; res2; res3; res4];

end

function res = int_region(thexp, Mexp, thcomp, Mcomp, gamma,...
    dtheta, th_M)

    theta = th_M(1);
    M = th_M(2);

    res1 = (theta - prmey([M,gamma]) + prmey([Mcomp,gamma]) - thcomp)...
        / dtheta;
    res2 = (theta + prmey([M,gamma]) - prmey([Mexp,gamma]) - thexp)...
        / dtheta;
    res = [res1; res2];

end
```

interior.m

```
function x4_y4_M4_th4_p4_r4 = interior(x0, y0, M0, th0, p0, r0, ...
    gamma, k, l)
% Takes initial data line, outputs interior points at next downstream
% station, by applying MOC. Note that in computing each point, the upstream
% points are denoted 1 and 2, the downsteam point is denoted 4, and the
% interpolated point on the initial data line is denoted 3. k is the number
% of iterations desired to compute point 3 within each larger iteration. l
% is the number of iterations for the corrector algorithm.

% CAUTION: To avoid infinity errors, input values with x-axis corresponding
% to primary flow direction, and ensure that initial data line aligns
% roughly with y-axis. Also, may want to non-dimensionalize x, y, p, r.

    n = length(x0);

    x1 = x0(1:end-1);
    x2 = x0(2:end);
    y1 = y0(1:end-1);
    y2 = y0(2:end);
    M1 = M0(1:end-1);
    M2 = M0(2:end);
    mu1 = asin(1./M1);
    mu2 = asin(1./M2);
    th1 = th0(1:end-1);
    th2 = th0(2:end);
    p1 = p0(1:end-1);
    p2 = p0(2:end);
    r1 = r0(1:end-1);
    r2 = r0(2:end);
    dx0 = x0(2:end) - x0(1:end-1);
    dy0 = y0(2:end) - y0(1:end-1);

    % Initial estimates for (x4, y4), (x3, y3), and th3
    A1 = [diag(-tan(th1+mu1)), eye(n-1); diag(-tan(th2-mu2)), eye(n-1)];
    B1 = [y1 - tan(th1+mu1) .* x1; y2 - tan(th2-mu2) .* x2];
    x4y4 = A1\B1;
    x4 = x4y4(1:n-1);
    y4 = x4y4(n:end);

    A2 = [diag(-tan((th1+th2)./2)), eye(n-1); diag((y1-y2)./(x2-x1)), eye(n-
1)];
    B2 = [y4 - tan((th1+th2)./2) .* x4; y1 + (y1-y2)./(x2-x1) .* x1];
    x3y3 = A2\B2;
    x3 = x3y3(1:n-1);
    y3 = x3y3(n:end);

    th3 = th1 + (y3-y1) .* (th2-th1) ./ (y2-y1);

    % Improve estimate for (x3 y3) and th3 (k iterations)
    for k1 = 1:k
        A2 = [diag(-tan(th3)), eye(n-1); diag((y1-y2)./(x2-x1)), eye(n-1)];
        B2 = [y4 - tan(th3) .* x4; y1 + (y1-y2)./(x2-x1) .* x1];
        x3y3 = A2\B2;
        x3 = x3y3(1:n-1);
        y3 = x3y3(n:end);

        th3 = th1 + (y3-y1) .* (th2-th1) ./ (y2-y1);
    end
```

```
% Compute interpolated p3, r3, M3
p3 = p1 + (y3-y1) .* (p2-p1) ./ (y2-y1);
r3 = r1 + (y3-y1) .* (r2-r1) ./ (y2-y1);
M3 = M1 + (y3-y1) .* (M2-M1) ./ (y2-y1);

% Initial calculation of properties at 4
R1 = gamma .* M1.^2 ./ sqrt(M1.^2 - 1);
R2 = gamma .* M2.^2 ./ sqrt(M2.^2 - 1);
A4 = [eye(n-1), diag(R1); eye(n-1), diag(-R2)];
B4 = [log(p1) + R1 .* th1; log(p2) - R2 .* th2];
lnp4th4 = A4\B4;
lnp4 = lnp4th4(1:n-1);
p4 = exp(lnp4);
th4 = lnp4th4(n:end);
M4 = sqrt(2./(gamma-1) .* ((1 + (gamma-1)/2 .* M3.^2) ./ ...
    (p4./p3).^((gamma-1)/gamma) - 1));
r4 = (p4./p3).^(1/gamma) .* r3;

% Apply corrector algorithm (l iterations)
for k2 = 1:l
    % (x4, y4) and th3
    A1 = [diag(-tan((th1+th4)./2 + asin(2./(M1+M4)))), ...
        eye(n-1); diag(-tan((th2+th4)./2 - asin(2./(M2+M4)))), ...
        eye(n-1)];
    B1 = [y1 - tan((th1+th4)./2 + asin(2./(M1+M4))) .* ...
        x1; y2 - tan((th2+th4)./2 - asin(2./(M2+M4))) .* x2];
    x4y4 = A1\B1;
    x4 = x4y4(1:n-1);
    y4 = x4y4(n:end);

    th3 = (th3+th4) ./ 2;

    % Improve th3 and (x3, y3) (k iterations)
    for k1 = 1:k
        A2 = [diag(-tan(th3)), eye(n-1); diag((y1-y2)./(x2-x1)), eye(n-1)];
        B2 = [y4 - tan(th3) .* x4; y1 + (y1-y2)./(x2-x1) .* x1];
        x3y3 = A2\B2;
        x3 = x3y3(1:n-1);
        y3 = x3y3(n:end);

        th3 = th1 + (y3-y1) .* (th2-th1) ./ (y2-y1);
    end

    % Compute interpolated p3, r3, M3
    p3 = p1 + (y3-y1) .* (p2-p1) ./ (y2-y1);
    r3 = r1 + (y3-y1) .* (r2-r1) ./ (y2-y1);
    M3 = M1 + (y3-y1) .* (M2-M1) ./ (y2-y1);

    % Improved calculation of properties at 4
    R1 = gamma .* ((M1+M4)./2).^2 ./ sqrt(((M1+M4)./2).^2 - 1);
    R2 = gamma .* ((M2+M4)./2).^2 ./ sqrt(((M2+M4)./2).^2 - 1);
    A4 = [eye(n-1), diag(R1); eye(n-1), diag(-R2)];
    B4 = [log(p1) + R1 .* th1; log(p2) - R2 .* th2];
    lnp4th4 = A4\B4;
    lnp4 = lnp4th4(1:n-1);
    p4 = exp(lnp4);
    th4 = lnp4th4(n:end);
    M4 = sqrt(2./(gamma-1) .* ((1 + (gamma-1)/2 .* M3.^2) ./ ...
        (p4./p3).^((gamma-1)/gamma) - 1));
    r4 = (p4./p3).^(1/gamma) .* r3;

end
```

```
    x4_y4_M4_th4_p4_r4 = [x4, y4, M4, th4, p4, r4];

end
```

intersection.m

```
function xy_int = intersection(dydx_1, dydx_2, x_01, y_01, x_02, y_02)

    x = (y_02 - y_01 + dydx_1 .* x_01 - dydx_2 .* x_02) ./ (dydx_1 - dydx_2);
    y = y_01 + dydx_1 .* (x - x_01);
    xy_int = [x, y];

end
```

M_s.m

```
function Ms = M_s(beta_theta1_thetas_M1_p1_r1_gamma)

    beta = beta_theta1_thetas_M1_p1_r1_gamma(1);
    theta1 = beta_theta1_thetas_M1_p1_r1_gamma(2);
    thetas = beta_theta1_thetas_M1_p1_r1_gamma(3);
    M1 = beta_theta1_thetas_M1_p1_r1_gamma(4);
    gamma = beta_theta1_thetas_M1_p1_r1_gamma(7);
    Mn1 = M1 * sin(beta - theta1);

    num = Mn1^2 + 2 / (gamma - 1);
    den = 2 * gamma / (gamma - 1) * Mn1^2 - 1;
    Mns = sqrt(num/den);
    Ms = abs(Mns / sin(beta - thetas));

end
```

moeckel.m

```
function res = moeckel(beta_ip1, gamma, M_i, M_ip1, beta_i, ...
    r, c)
% Outputs residual of Mockel eq (30)
% Check magnitude of RHS--is normalizing residual with RHS a problem?

    M_bar = 0.5 * (M_i + M_ip1);
    beta_bar = 0.5 * (beta_i + beta_ip1);
    f_1 = f1(gamma, M_bar, beta_bar);
    g_2 = g2(gamma, M_bar, beta_bar);
    f_3 = f3(gamma, M_bar, beta_bar);
    df1_db = df1db(gamma, M_bar, beta_bar);
    df1_dM = df1dM(gamma, M_bar, beta_bar);
    df2_db = df2db(gamma, M_bar, beta_bar, g_2);
    df2_dM = df2dM(gamma, M_bar, beta_bar, g_2);

    LHS = (beta_ip1 - beta_i) ./ (M_ip1 - M_i);
    den = df1_db ./ f_1 + f_3 .* df2_db;
    T1 = (df1_dM ./ f_1 + f_3 .* df2_dM) ./ den;

    if r <= c
        T2 = ((-1 * gamma * M_bar) + sqrt(M_bar.^2 - 1) ./ M_bar .* f_3) ./ ...
            (1 + (gamma-1)/2 * M_bar.^2) ./ den;
    else
        T2 = ((-1 * gamma * M_bar) - sqrt(M_bar.^2 - 1) ./ M_bar .* f_3) ./ ...
            (1 + (gamma-1)/2 * M_bar.^2) ./ den;
    end

    RHS = -1 * T1 - T2;

    res = (LHS - RHS) ./ RHS;

    function f_1 = f1(gamma, M_bar, beta_bar)
        f_1 = (2*gamma) / (gamma+1) * M_bar.^2 * (sin(beta_bar)).^2 - ...
            (gamma-1) / (gamma+1);
    end

    function g_2 = g2(gamma, M_bar, beta_bar)
        num_g2 = M_bar.^2 .* sin(beta_bar) .* cos(beta_bar) - cot(beta_bar);
        den_g2 = 1 + M_bar.^2 .* ((gamma+1)/2 - (sin(beta_bar)).^2);
        g_2 = num_g2 ./ den_g2;
    end

    function f_3 = f3(gamma, M_bar, beta_bar)
        num_f3 = (gamma+1)^2 * M_bar.^4 .* (sin(beta_bar)).^2 - ...
            4 * (M_bar.^2 .* (sin(beta_bar)).^2 - 1) .* ...
            (gamma * M_bar.^2 .* (sin(beta_bar)).^2 + 1);
        den_f3 = (2 * gamma * M_bar.^2 .* (sin(beta_bar)).^2 - gamma + ...
            1) .* ((gamma-1) * M_bar.^2 .* (sin(beta_bar)).^2 + 2);
        Ms_sq = num_f3 ./ den_f3;
        f_3 = gamma * Ms_sq ./ sqrt(Ms_sq - 1);
    end

    function df1_db = df1db(gamma, M_bar, beta_bar)
        df1_db = 2*gamma / (gamma+1) * M_bar.^2 .* sin(2*beta_bar);
    end

    function df1_dM = df1dM(gamma, M_bar, beta_bar)
        df1_dM = 4*gamma / (gamma+1) * M_bar .* (sin(beta_bar)).^2;
    end
    function df2_db = df2db(gamma, M_bar, beta_bar, g_2)
        num_df2db = (1 + M_bar.^2 .* ((gamma+1)/2 - (sin(beta_bar)).^2)) .* ...
            (M_bar.^2 .* cos(2*beta_bar) + (csc(beta_bar)).^2) ...
            + M_bar.^2 .* sin(2*beta_bar) .* ...
            (M_bar.^2 .* sin(beta_bar) .* cos(beta_bar) - cot(beta_bar));
        den_df2db = (1 + M_bar.^2 .* ((gamma+1)/2 - (sin(beta_bar)).^2)).^2;
        df2_db = (1./(g_2.^2 + 1)) .* num_df2db ./ den_df2db;
    end

    function df2_dM = df2dM(gamma, M_bar, beta_bar, g_2)
        num_df2dM = (1 + M_bar.^2 .* ((gamma+1)/2 - (sin(beta_bar)).^2)) .* ...
            M_bar .* sin(2*beta_bar) - 2*M_bar .* ...
            ((gamma+1)/2 - (sin(beta_bar)).^2) .* ...
            (M_bar.^2 .* sin(beta_bar) .* cos(beta_bar) - cot(beta_bar));
        den_df2dM = (1 + M_bar.^2 .* ((gamma+1)/2 - (sin(beta_bar)).^2)).^2;
        df2_dM = (1./(g_2.^2 + 1)) .* num_df2dM ./ den_df2dM;
    end

end
```

p_exp.m

```
function pexp = p_exp(M1_M2_p1_gamma)

    M1 = M1_M2_p1_gamma(1);
    M2 = M1_M2_p1_gamma(2);
    p1 = M1_M2_p1_gamma(3);
    gamma = M1_M2_p1_gamma(4);

    num = 1 + (gamma-1) / 2 * M2^2;
    den = 1 + (gamma-1) / 2 * M1^2;

    pexp = p1 * (den/num)^(gamma/(gamma-1));

end
```

p_s.m

```
function ps = p_s(beta_theta1_thetas_M1_p1_r1_gamma)

    beta = beta_theta1_thetas_M1_p1_r1_gamma(1);
    theta1 = beta_theta1_thetas_M1_p1_r1_gamma(2);
    M1 = beta_theta1_thetas_M1_p1_r1_gamma(4);
    p1 = beta_theta1_thetas_M1_p1_r1_gamma(5);
    gamma = beta_theta1_thetas_M1_p1_r1_gamma(7);

    Mn1 = M1 * sin(beta-theta1);
    ps = p1 * (1 + 2*gamma/(gamma+1) * (Mn1^2 - 1));

end
```

prmey.m

```
function v = prmey(M_gamma)

    M = M_gamma(1);
    gamma = M_gamma(2);

    r = (gamma + 1) / (gamma - 1);
    t1 = sqrt(r) * atan(sqrt(1/r * (M^2 - 1)));
    t2 = atan(sqrt(M^2 - 1));

    v = t1 - t2;

end
```

r_s.m

```
function rs = r_s(beta_theta1_thetas_M1_p1_r1_gamma)

    beta = beta_theta1_thetas_M1_p1_r1_gamma(1);
    theta1 = beta_theta1_thetas_M1_p1_r1_gamma(2);
    M1 = beta_theta1_thetas_M1_p1_r1_gamma(4);
    r1 = beta_theta1_thetas_M1_p1_r1_gamma(6);
    gamma = beta_theta1_thetas_M1_p1_r1_gamma(7);

    Mn1 = M1 * sin(beta-theta1);
    num = (gamma + 1) * Mn1^2;
    den = (gamma - 1) * Mn1^2 + 2;
    rs = r1 * num / den;

end
```

T_exp.m

```
function Texp = T_exp(M1_M2_T1_gamma)

    M1 = M1_M2_T1_gamma(1);
    M2 = M1_M2_T1_gamma(2);
    T1 = M1_M2_T1_gamma(3);
    gamma = M1_M2_T1_gamma(4);

    num = 1 + (gamma-1) / 2 * M2^2;
    den = 1 + (gamma-1) / 2 * M1^2;

    Texp = T1 * den / num;

end
```

tbm_rhs.m

```
function rhs = tbm_rhs( beta_theta1_M1_gamma )

    beta = beta_theta1_M1_gamma(1);
    theta1 = beta_theta1_M1_gamma(2);
    M1 = beta_theta1_M1_gamma(3);
    gamma = beta_theta1_M1_gamma(4);
    beta_eff = beta - theta1;

    num = M1^2 * sin(beta_eff)^2 - 1;
    den = M1^2 * (gamma + cos(2*beta_eff)) + 2;

    rhs = 2 * cot(beta_eff) * num / den;

end
```

type1.m

```
function res = type1(betac4c3, theta23, M23, p23, gamma)

    betac4 = betac4c3(1);
    betac3 = betac4c3(2);
    theta2 = theta23(1);
    theta3 = theta23(2);
    M2 = M23(1);
    M3 = M23(2);
    p2 = p23(1);
    p3 = p23(2);

    betac4_theta2_M2_gamma = [betac4, theta2, M2, gamma];
    theta4 = theta2 + atan(tbm_rhs(betac4_theta2_M2_gamma));
    betac3_theta3_M3_gamma = [betac3, theta3, M3, gamma];
    theta5 = theta3 + atan(tbm_rhs(betac3_theta3_M3_gamma));
    res1 = (theta4 - theta5) * 180 / pi;

    betac4_theta2_thetas_M2_p2_r_gamma = [betac4, theta2, 0, M2, p2, 0, gamma];
    p4 = p_s(betac4_theta2_thetas_M2_p2_r_gamma);
    betac3_theta3_thetas_M3_p3_r_gamma = [betac3, theta3, 0, M3, p3, 0, gamma];
    p5 = p_s(betac3_theta3_thetas_M3_p3_r_gamma);
    res2 = (p5 - p4) / (p5 + p4);

    res = [res1; res2];

end
```

upstream_c5.m

```
function thu_Mu_pu_Tu_xip1_yip1_r_c = upstream_c5(beta_i, x_i, y_i, ...
    xy_char_2, th_M_p_T, th_M_p_T_2, N)

% Outputs next upstream th,M,P,T when computing reattachment shock
% shape, x_{i+1} and y_{i+1}, and index in xy_char_2

    % Determine row-location of intersection with each characteristic line
    y_hat = y_i + tan(beta_i) .* (xy_char_2(:,1:2:end-1) - x_i);
    delta_y = xy_char_2(:,2:2:end) - y_hat;
    npos = zeros(1,N);
    for ii = 1:N
        a = find(delta_y(:,ii) < 0, 1);
        a = a - 1;
        npos(ii) = a;
    end

    % Calculate point of intersection with each characterstic segment,
    % based on row location
    dydx_2 = (diag(xy_char_2(npos+1,2:2:end)) - diag(xy_char_2(npos,2:2:end)))
./...
        (diag(xy_char_2(npos+1,1:2:end-1)) - diag(xy_char_2(npos,1:2:end-1)));
    xy_int = intersection(tan(beta_i), dydx_2, x_i, y_i, ...
        diag(xy_char_2(npos,1:2:end-1)), diag(xy_char_2(npos,2:2:end)));
    x_int = xy_int(:,1);

    % Find next interesction between shock and a characterstic; find column
    % of xychar_2 corresponding to intersection. Note that actual columns
    % in xy_char_2 are 2*r-1 and 2*r
    delta_x = x_int - x_i;
    min_dx = min(delta_x(delta_x>1e-8));
    rr = find(delta_x == min_dx);
    r = rr(1);

    i = npos(r); % row of xychar_2 entry before intersection
    j = r;

    if i > N+1
        error('Error in upstream_c5.m: shock extends beyond x limit of
xy_char_2. Try increasing limit.')
    end

    if i == 1
        R = (j+1)*N - 1 - j*(j-1)/2;
        thu = th_M_p_T(R,1);
        Mu = th_M_p_T(R,2);
        pu = th_M_p_T(R,3);
        Tu = th_M_p_T(R,4);
    elseif j == 1 && i ~= 1
        j = i - 1;
        R = (j+1)*N - 1 - j*(j-1)/2;
        thu = th_M_p_T(R,1);
        Mu = th_M_p_T(R,2);
        pu = th_M_p_T(R,3);
        Tu = th_M_p_T(R,4);
    elseif i <= j && i ~= 1 && j ~= 1
        R = (j-1)*j/2 + i - 1;
        thu = th_M_p_T_2(R,1);
        Mu = th_M_p_T_2(R,2);
        pu = th_M_p_T_2(R,3);
        Tu = th_M_p_T_2(R,4);
    elseif i > j && i ~= 1 && j ~= 1
        R = (i-1)*(i-2)/2 + j - 1;
        thu = th_M_p_T_2(R,1);
        Mu = th_M_p_T_2(R,2);
        pu = th_M_p_T_2(R,3);
        Tu = th_M_p_T_2(R,4);
    else
        error('Error in upstream_c5.m')
    end

    thu_Mu_pu_Tu_xip1_yip1_r_c = [thu; Mu; pu; Tu; x_int(r);...
        xy_int(r,2); npos(r); r];

end
```

upstream_c5_noreflec.m

```
function thu_Mu_pu_Tu_xip1_yip1 = upstream_c5_noreflec(beta_i, x_i, y_i, ...
    xy_char_2, th_M_p_T, N)

    i = 1;
    j = N - 1;
    dx = -1;

    while dx <= 1e-6
        xy_int = intersection(tan(beta_i), (xy_char_2(end,2*i) - ...
            xy_char_2(1,2*i)) / ...
            (xy_char_2(end,2*i-1) - xy_char_2(1,2*i-1)), ...
            x_i, y_i, xy_char_2(1,2*i-1), xy_char_2(1,2*i));
        dx = xy_int(1) - x_i;
        j = j + N + 1 - i;
        i = i + 1;
    end

    thu_Mu_pu_Tu_xip1_yip1 = [th_M_p_T(j,:)'; xy_int(1); xy_int(2)];

end
```

Bibliography

[1] Henderson, L.F., "The Reflection of a Shock Wave at a Rigid Wall in the Presence of a Boundary Layer," *Journal of Fluid Mechanics*, Vol. 30, Part 4, 1967, pp. 699-722.

[2] Lees, L., "Interaction Between the Laminar Boundary Layer over a Plane Surface and an Incident Oblique Shock Wave," Princeton University Aeronautical Engineering Laboratory, Report No. 143, Princeton, NJ, January 1949.

[3] Chapman, D.R., Kuehn, D.M., and Larson, H.K., "Investigation of Separated Flows in Supersonic and Subsonic Streams with Emphasis on the Effect of Transition," NACA TR-1356, January 1958.

[4] Barry, F.W., Shapiro, A.H., and Neumann, E.P., "The Interaction of Shock Waves with Boundary Layers on a Flat Surface," *Journal of the Aeronautical Sciences*, Vol. 18, No. 4, 1951, pp. 229-238.

[5] Hakkinen, R.J., Greber, I., Trilling, L., and Abarbanel, S.S., "The Interaction of an Oblique Shock Wave with a Laminar Boundary Layer," Massachusetts Institute of Technology, Fluid Dynamic Research Group, TR-57-1, Cambridge, MA, May 1957.

[6] Lees, L., Reeves, B.L., "Supersonic Separated and Reattaching Laminar Flows: I. General Theory and Application to Adiabatic Boundary-Layer/Shock-Wave Interactions," *AIAA Journal*, Vol. 2, No. 11, 1964, pp. 1907-20.

[7] Dolling, D.S., "Fifty Years of Shock-Wave/Boundary-Layer Interaction Research: What Next?" *AIAA Journal*, Vol. 39, No. 8, 2001, pp. 1517-31.

[8] Gaitonde, D.V., "Progress in Shock Wave/Boundary Layer Interactions," *Progress in Aerospace Sciences*, Vol. 72, 2015, pp. 80-99.

[9] Delery, J., and Marvin, J.G., "Shock-Wave Boundary Layer Interactions," AGARD AG-280, Neuilly-sur-Seine, February 1986.

[10] Erdos, J., and Pallone, A., "Shock/Boundary-Layer Interaction and Flow Separation," *Proceedings of the 1962 Heat Transfer and Fluid Mechanics Institute*, Stanford University Press, Stanford, 1962.

[11] Zukoski, E.E., "Turbulent Boundary-Layer Separation in Front of a Forward-Facing Step," *AIAA Journal*, Vol. 5, No. 10, 1967, pp. 1746-53.

[12] Babinsky, H., and Harvey, J.K., *Shock Wave—Boundary-Layer Interactions*, Cambridge University Press, Cambridge, 2011.

[13] Zucrow, M.J., and Hoffman, J.D., *Gas Dynamics*, Vol. 2, Wiley, New York, 1977, pp. 185-266.

[14] Edney, B., "Anomalous heat transfer and pressure distributions on blunt bodies at hypersonic speeds in the presence of an impinging shock," Aeronautical Research Institute of Sweden, FFA Report 115, Stockholm, February 1968.

[15] Moeckel, W.E., "Interaction of Oblique Shock Waves with Regions of Variable Pressure, Entropy, and Energy," NACA TN-2725, Washington, 1952.

[16] Holden, M.S., "Shock Wave-Turbulent Boundary Layer Interaction in Hypersonic Flow," 10th AIAA Aerospace Sciences Meeting, AIAA Paper No. 72-74, January 1972.

[17] Needham, D.A., and Stollery, J.L., "Boundary Layer Separation in Hypersonic Flow," 4th AIAA Aerospace Sciences Meeting, AIAA Paper No. 66-455, June 1966.