A Cartesian Grid Embedded Boundary Method for the Compressible Navier Stokes Equations

Daniel T. Graves [†] Phillip Colella [†] David Modiano [‡]
Jeffrey Johnson [†] Bjorn Sjogreen [§] Xinfeng Gao[¶]

October 9, 2013

Abstract

In this paper, we present an unsplit method for the time-dependent compressible Navier-Stokes equations in two and three dimensions. We use a a conservative, second-order Godunov algorithm. We use a Cartesian grid, embedded boundary method to resolve complex boundaries. We solve for viscous and conductive terms with a second-order semi-implicit algorithm. We demonstrate second-order accuracy in solutions of smooth problems in smooth geometries and demonstrate robust behavior for strongly discontinuous initial conditions in complex geometries.

1 Introduction

In this paper, we present an unsplit method for the time-dependent compressible Navier-Stokes equations in two and three dimensions. This algorithm is an extension of the algorithm in [9] to flows with viscous and thermal diffusion. The Navier-Stokes equations contain parabolic terms that arise from conductivity and viscosity. There are several methods to advance these terms. In [10], for example, a kinetic energy equation is evolved to get a stable approximation to the viscous term in the energy equation. This solution is elegant but also difficult to extend to multiple dimensions. We use a conservative, semi-implicit method in which the hyperbolic terms are advanced explicitly and the parabolic terms advanced implicitly. This approach to the compressible Navier-Stokes equations has been used without embedded boundaries [3, 30, 16, 14, 11]. Our algorithm

[†]Lawrence Berkeley National Laboratory, Berkeley, CA. Research at LBNL was supported financially by the Office of Advanced Scientific Computing Research of the US Department of Energy under contract number DE-AC02-05CH11231.

[‡]Sanzaru Games, Foster City CA

[§]Lawrence Livermore National Laboratory, Livermore CA. This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

[¶]Colorado State University, Fort Collins, CO

follows the basic outline in the mapped grid algorithm presented in [30], in which the velocity and temperature evolution are split. They use a Crank-Nicolson time evolution with the energy-momentum coupling term treated explicitly. We use a hybrid approach to energy-momentum coupling. Also, since Crank Nicolson has been shown to be marginally stable in certain cases [23], we use the L_0-stable algorithm presented in [31] for elliptic coupling. We present our changes to the [31] algorithm that were necessary to make the linear equations tractable in the presence of small cells. This algorithm has been implemented with adaptive mesh refinement (AMR) as described in [4, 1]. All our cut cells are refined to the finest level, reducing all coarse-fine interactions (such as refluxing and coarse-fine interpolation) to exactly those described in [30].

Dragovic, et al. [12] present a two-dimensional algorithm for viscous, conducting compressible flow with embedded boundaries. They use a split hyperbolic scheme, explicit updates of the viscous state and the (formally inconsistent) extended state algorithm developed in [24]. Our algorithm uses an unsplit scheme (as seen in [8, 25, 2]) and works in two and three dimensions. Ghias, et al. [13] present an immersed boundary method to solve the same set of equations for subsonic applications. Hartmann, et al. [15] present a cut-cell method that uses a form of cell merging to achieve small-cell stability. Berger, et al. [5] survey a wide variety of these algorithmic permutations. We use redistribution (first presented by Chern, et al. [7]) for small-cell stability. We use the (formally consistent) approach in [9] to construct extended states. To evaluate viscous fluxes at the embedded boundary we use the ray-casting algorithm developed in [18] for Poisson's equation. Also, for increased stability, we treat the viscous stress and conductivity terms implicitly.

This algorithm is suitable for use in applications where compressibility is important and the geometries are complex. Our target application is flow inside of capillary tubes in laser wakefield particle accelerators. In these accelerators, the pressure and temperature is driven very high along the axis of a capillary tube. The resulting flow produces a low density core through which lasers are shot. The capillary is connected to fill tubes which are used to fill the capillary with gas [27, 19, 20, 29]. We present a simplified version of this problem as our example to demonstrate robustness while acknowledging that other physics in these problems (such as ionization and magnetization) are very important. We drive a capillary tube with a large pressure pulse to demonstrate the stability of the algorithm under extreme conditions. The geometric configuration is derived from the experimental setup described in [29].

There are of course many regimes for which the compressible Navier Stokes equations are relevant. The regime of interest for this algorithm has substantial compressibility effects (including shocks) as well as substantial viscous effects. We are also interested in time-accurate (as opposed to steady state calculations).For algorithm validation,we run several examples which demonstrate the efficacy of the algorithm in this regime.

First we present convergence tests demonstrating second-order solution error accuracy in two and three dimensions. For these tests, we use a smooth, subsonic ($M = 0.5$) flow inside a sphere. This demonstrates that, even with

compressibility effects, we get the expected convergence rate for smooth problems.

Next, for more a quantitative validations, we present a boundary layer calculation and a viscous shock reflection calculation. Charest, et al. [6], present a low-Mach number algorithm for steady state calculations. They present a boundary layer calculation that reproduces the behavior of the similarity solution which emerges from analysis (a Blasius boundary layer profile). We present a similar run which also reproduces Blasius behavior. This demonstrates that the algorithm has correct boundary layer behavior.

Glaz, et al. [14] present a comparison between inviscid calculations of shock reflections and experimental results. They show a case where viscous effects cause substantial changes in the reflection pattern. We present both viscous and inviscid calculations of the same problem and show good agreement with their results. This demonstrates, that even in this very complex, time-dependent flow, we compare well with experiment.

2 Notation

Cartesian grids with embedded boundaries are useful to describe finite-volume representations of solutions to partial differential equations in the presence of irregular boundaries. In figure 1, the grey area represents the region excluded from the solution domain. The underlying description of space is given by rectangular control volumes on a Cartesian mesh $\Upsilon_{\mathbf{i}} = [(\mathbf{i} - \frac{1}{2}\mathbf{v})h, (\mathbf{i} + \frac{1}{2}\mathbf{v})h], \mathbf{i} \in \mathbb{Z}^D$, where D is the dimensionality of the problem, h is the mesh spacing, and $\mathbf{v}$ is the vector whose entries are all one. Given an irregular domain Ω, we obtain control volumes $V_{\mathbf{i}} = \Upsilon_{\mathbf{i}} \bigcap \Omega$ and faces $A_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d}$ which are the intersection of the boundary of $\partial V_{\mathbf{i}}$ with the coordinate planes $\{\mathbf{x} : x_d = (i_d \pm \frac{1}{2})h\}$. We also define $A^B_{\mathbf{i}}$ to be the intersection of the boundary of the irregular domain with the Cartesian control volume: $A^B_{\mathbf{i}} = \partial\Omega \bigcap \Upsilon_{\mathbf{i}}$. For ease of exposition, we will assume here that there is only one control volume per Cartesian cell. The algorithm described here has been generalized to allow for boundaries whose width is less that the mesh spacing.

To construct finite-volume methods using this description, we will need several quantities derived from these geometric objects.

- Volume fractions κ and area fraction α:

$$\kappa_{\mathbf{i}} = \frac{|V_{\mathbf{i}}|}{h^D} \quad , \quad \alpha_{\mathbf{i}+\frac{1}{2}\mathbf{e}_s} = \frac{|A_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d_s}|}{h^{D-1}}, \quad \alpha^B_{\mathbf{i}} = \frac{|A^B_{\mathbf{i}}|}{h^{D-1}}$$

- The centroids of the faces and of $A^B_{\mathbf{i}}$; and $\mathbf{n}$, the average of outward normal

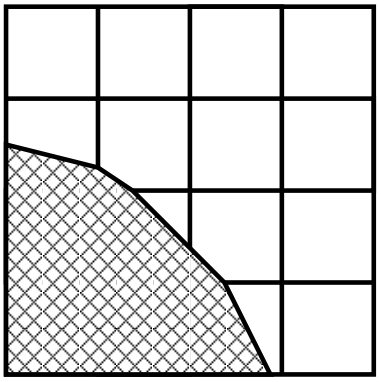

Figure 1: Illustration of cut cells. The shaded area is outside the solution domain.

of $\partial\Omega$ over $A_{\mathbf{i}}^B$.

$$\mathbf{x}_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d} = \frac{1}{|A_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d}|} \int\limits_{A_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d}} \mathbf{x} dA - (\mathbf{i} + \frac{1}{2}\mathbf{e}^d)h$$

$$\mathbf{x}_{\mathbf{i}}^B = \frac{1}{|A_{\mathbf{i}}^B|} \int\limits_{A_{\mathbf{i}}^B} \mathbf{x} dA - \mathbf{i}h$$

$$\mathbf{n}_{\mathbf{i}} = \frac{1}{|A_{\mathbf{i}}^B|} \int\limits_{A_{\mathbf{i}}^B} \mathbf{n} dA$$

where D is the dimension of space and $1 <= d <= D$. We assume we can compute all derived quantities to $O(h^2)$. With just these geometric descriptors, we can define a conservative discretization of the divergence operator. Let $\vec{F} = (F^1 ... F^D)$ be a function of $\mathbf{x}$, then

$$\nabla \cdot \vec{F} \approx \frac{1}{|V_{\mathbf{i}}|} \int\limits_{V_{\mathbf{i}}} \vec{F} dV = \frac{1}{|V_{\mathbf{i}}|} \int\limits_{\partial V_{\mathbf{i}}} \vec{F} \cdot \mathbf{n} dA.$$

We discretize the divergence of the flux as

$$\kappa D(F)_{\mathbf{i}} = \frac{1}{h} \left(\sum_{d=1}^{D} \sum_{\pm=+,-} \pm \alpha_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d} F^d(\mathbf{x}_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d}) + \alpha_{\mathbf{i}}^B \mathbf{n}_{\mathbf{i}} \cdot \vec{F}(\mathbf{x}_{\mathbf{i}}^B) \right) \tag{1}$$

where (1) is obtained by replacing the normal components of the vector field $\vec{F}$ with the values at the centroids. This converges to the exact divergence by the relation $D(F)_{\mathbf{i}} = \nabla \cdot F + O\left(\frac{h}{\kappa_{\mathbf{i}}}\right)$ in cells which intersect the embedded boundary and converges to $O(h^2)$ away from the boundary. The elliptic operators in this calculation all take the form

$$L(\phi) = a(\mathbf{x})\phi + D(F(\phi)).$$

We refer to a in this context as the identity coefficient.

3 System of Equations

We are solving the compressible Navier-Stokes equations, given here in conservation form with hyperbolic terms to the left and elliptic terms to the right.

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0 \\ \frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u} + pI) &= \nabla \cdot \sigma \\ \frac{\partial (\rho E)}{\partial t} + \nabla \cdot (\rho \mathbf{u} E + \mathbf{u} p) &= \nabla \cdot (\sigma \mathbf{u}) + \nabla \cdot (\xi (\nabla T)) \end{aligned} \tag{2}$$

In these equations, ρ is the mass density, $\mathbf{u}$ is the velocity, ξ is the thermal conductivity, p is the pressure, and T is the temperature. The shear stress tensor σ is given by

$$\sigma = \mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T) + \lambda(\nabla \cdot \mathbf{u})I.$$

where μ and λ are the viscosity coefficients (typically $\lambda = \frac{-2}{3}\mu$). The total energy is given by $E = e + \frac{1}{2}|\mathbf{u}|^2$; the internal energy is given by $e = C_v T$ (where C_v is the specific heat at constant volume). The fluid is assumed to be an ideal gas ($p = C_v(\gamma - 1)\rho T$).

4 Algorithm Description

We define $U = (\rho, \rho\mathbf{u}, \rho E)$ and we define $L^H(U) = \nabla \cdot F$, the divergence of the hyperbolic flux. The flux is given by

$$F = \begin{pmatrix} \rho\mathbf{u} \\ \rho\mathbf{u}\mathbf{u} + pI \\ \rho\mathbf{u}E + \mathbf{u}p \end{pmatrix}.$$

The divergence and the fluxes are computed in the same way as in [9]. To summarize, a Taylor series extrapolation is done to produce second-order (in both space and time) approximations to the fluxes at the centroids of the faces. A conservative approximation to the divergence ($D^c(F)$) is computed using (1). Ideally, we would use $D^c(F)$ for our hyperbolic divergence. The difficulty with this approach is that the CFL stability constraint on the time step for an algorithm using the conservative divergence for an explicit update is at best $\Delta t = O(\frac{h}{v_{\mathbf{i}}^{max}}(\kappa_{\mathbf{i}})^{\frac{1}{D}})$, where $v_{\mathbf{i}}^{max}$ is the magnitude of the maximum wave speed for the $\mathbf{i}^{th}$ control volume. This is the well-known small-cell problem for embedded boundary methods. Instead, we compute a stable, non-conservative approximation to the divergence ($D^{nc}(F)$) using an extended state where necessary and ignoring the embedded boundary. This extended state is extrapolated from the interior. The effective divergence is

$$L^H(U) = \kappa D^c(F) + (1 - \kappa)D^{nc}(F).$$

The mass difference (δM) between using L^H and using only the conservative divergence $D^c(F)$ is given by

$$\delta M = \kappa(1 - \kappa)(D^c(F) - D^{nc}(F))$$

. This mass difference is redistributed to neighboring cells. The redistribution algorithm is described in [7]. This hybrid formulation preserves conservation and allows this algorithm to be stable using a time step constraint based on full cells. We compute our time step as follows:

$$\Delta t = \frac{C_F h}{W^{\max}}, \tag{3}$$

where $W^{\max}$ is the maximum wave speed in the problem and C_F is the Courant number ($0 < C_F < 1$).

Define L^v to be the elliptic terms in the system of equations

$$L^v(U)_{\mathbf{i}} = \begin{pmatrix} 0 \\ L^m(\mathbf{u})_{\mathbf{i}} \\ L^k(T)_{\mathbf{i}} + L^d(\mathbf{u})_{\mathbf{i}} \end{pmatrix}$$

The term L^m is the discretization of the viscous stress term ($L^m \approx \nabla \cdot \sigma$) and is described in section 5.3. The term $L^k \approx \nabla \cdot \xi \nabla T$ is a discretization of the heat conduction term and is described in section 5.2. The term $L^d \approx \nabla \cdot (\sigma \mathbf{u})$ is the viscous heating term and is described in section 5.1.

4.1 Outline

We begin with the state at time $U^n = U(n\Delta t)$, we advance the solution as follows.

1. Compute U^*, the solution advanced explicitly using only hyperbolic terms.

   $$U^*_{\mathbf{i}} = U^n_{\mathbf{i}} - \Delta t L^H_{\mathbf{i}}(U^n)$$

   This produces the final value of density ($\rho^{n+1} = \rho^*$). From U^*, we compute $\mathbf{u}^*$ and T^*, the intermediate values of velocity and temperature (which exclude the effects of conduction and viscosity).

2. Compute L_0-stable approximations to the momentum diffusion $L^m(U) = \nabla \cdot \sigma$ by advancing the diffusion equation

   $$\rho \frac{\partial \mathbf{u}}{\partial t} = L^m(\mathbf{u})$$

   using the method described in section 5

   $$\mathbf{u}^{n+1} = G_{L^m}(\rho^{n+1})\mathbf{u}^*$$

   where G is defined in equation 6. The stable approximation to $L^m(\mathbf{u})$ is calculated as

   $$(L^m(\mathbf{u}))^{n+\frac{1}{2}} = \rho^{n+1}\left(\frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t}\right),$$

   giving us the final value of momentum:

   $$(\rho\mathbf{u})^{n+1} = (\rho\mathbf{u})^* + \Delta t (L^m(\mathbf{u}))^{n+\frac{1}{2}}.$$

   The operator L^m is described in section 5.3.

3. Using the value of $\mathbf{u}$ calculated above, calculate the viscous dissipation of energy ($L^d \approx \nabla \cdot (\sigma \mathbf{u})$) as described in section 4.2. We then update the energy with this term.

   $$(\rho E)^{**} = (\rho E)^* + \Delta t L^d(\mathbf{u}^{n+1})$$

   From E^{**}, we compute the intermediate value of temperature T^{**}.

4. Compute L_0-stable approximations to the conduction term

$$L^k(T) = \nabla \cdot \xi \nabla T$$

by advancing the diffusion equation

$$\rho C_v \frac{\partial T}{\partial t} = L^k(T)$$

using the method described in section 5,

$$T^{n+1} = G_{L^m}(\rho^{n+1} C_v) T^{**}.$$

where G is described in equation 6. The stable approximation to $L^k(T)$ is computed by

$$(L^k(T))^{n+\frac{1}{2}} = \rho^{n+1} C_v \left(\frac{T^{n+1} - T^{**}}{\Delta t} \right),$$

giving us the final value of energy

$$(\rho E)^{n+1} = (\rho E)^{**} + \Delta t (L^k(T))^{n+\frac{1}{2}}.$$

The operator $L^k(T)$ is described in section 5.2.

4.2 Viscous Dissipation Calculation

To avoid small-cell instabilities, we split up the $L^d(U)$ into conservative and non-conservative approximations much as we did with L^H. The conservative approximation to $L^{d,c} = \nabla \cdot (\sigma \mathbf{u})$ is described in section 5.1. The non-conservative form of the operator is given by the volume-weighted average of the neighbor's conservative operator evaluations. Define $N(\mathbf{i})$ to be the set of cells reachable from $\mathbf{i}$ by a unit monotone path. The non-conservative approximation of L^d is

$$L^{d,nc}(\mathbf{u})_{\mathbf{i}} = \frac{\sum_{j \in N(\mathbf{i})} (\kappa L^{d,c}(\mathbf{u}))_j}{\sum_{j \in N(\mathbf{i})} \kappa_j}.$$

We use a linear combination of conservative and non-conservative versions of the divergence to advance the solution.

$$L^d(U) = \kappa L^{d,c}(\mathbf{u}) + (1 - \kappa)(L^{d,nc}(\mathbf{u}))$$

To preserve conservation, we compute the the energy difference between this version and the conservative version.

$$\delta E = \Delta t \kappa (1 - \kappa)(L^{d,c} - L^{d,nc})$$

We push this energy correction δE into the solution implicitly. First we set a right hand side $R = 0$ and redistribute δE into the cells of R that can be reached

by a unit monotone path (as described in [7]). We then solve for a temperature difference that can account for this energy using the conduction operator

$$(\rho^{n+1}C_v I - \Delta t L^k)\delta^T = \Delta t R$$

This change in temperature is interpreted as an increment to the energy as follows:

$$(\delta E)^{**} = \rho^* C_v \delta^T.$$

We add $(\delta E)^{**}$ into E^{**}.

5 Stable Parabolic Discretizations

Twizell, et al. ([31]) present a second-order L_0-stable algorithm to advance the constant coefficient heat equation. Given the equation

$$\frac{\partial \phi}{\partial t} = \nu L \phi \tag{4}$$

their time advance takes the form

$$\phi^{n+1} = (I - \mu_1 L)^{-1}(I - \mu_2 L)^{-1}(I + \mu_3 L)\phi^n \tag{5}$$

where μ_1, μ_2, μ_3 are constants. In the present algorithm we have two parabolic equations of the form

$$a\frac{\partial \phi}{\partial t} = L(\phi)$$

where $a = a(\mathbf{x}) > 0$ is the identity coefficient. Define the operator $M(\phi) = \frac{L(\phi)}{a}$. In the case of our viscous operator (section 5.3) $M^m = \frac{L^m(\mathbf{u})}{\rho}$ and the case of conduction (section 5.2), $M^k(T) = \frac{L^k(T)}{\rho C_v}$. In both cases, the denominators are positive and restricted away from zero. In each case, a naive interpretation of (5) yields the following:

$$\phi^{n+1} = (I - \mu_1 M)^{-1}(I - \mu_2 M)^{-1}(I + \mu_3 M)\phi^n.$$

This is problematic in the presence of small cells because this would involve dividing by the volume fraction to evaluate M (see (1)) and volume fractions here can be arbitrarily close to zero. Using the matrix identity $(AB)^{-1} = B^{-1}A^{-1}$, we refactor the above equation.

$$\phi^{n+1} = G_L(a)\phi = (\kappa a I - \mu_1 \kappa L)^{-1}(\kappa a)(\kappa a I - \mu_2 \kappa L)^{-1}(\kappa a I + \mu_3 \kappa L)\phi^n \tag{6}$$

This is the implicit advance we use for stable discretizations of $L^m(\mathbf{u})$ and $L^k(T)$.

5.1 Viscous Heating Operator

The viscous heating operator flux is an approximation to the shear stress dotted with the velocity ($F^h = \sigma \cdot \mathbf{u}$).

$$F^h = (\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T) + \lambda I \nabla \cdot \mathbf{u}) \cdot \mathbf{u} \tag{7}$$

We compute the shear stress as described in section 5.3. To get face-centered velocities, we average from neighboring cells

$$\mathbf{u}_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d} = \frac{1}{2}(\mathbf{u}_{\mathbf{i}+\mathbf{e}^d} + \mathbf{u}_{\mathbf{i}})$$

At embedded boundaries and domain boundaries we set this flux to zero because the no slip condition requires that $\mathbf{u}|_{\partial\Omega} = 0$. We then can find the conservative discretization of the operator $L^{d,c}$ as given by (1).

5.2 Conductivity Operator

Our operator for heat conduction

$$L^k(T) = \nabla \cdot (\xi \nabla T)$$

is an extension to variable coefficients of the operator described in by Schwartz, et al. [28]. The flux at face centers for the discretization in (1) is given by

$$F^T_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d} = \xi_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d} \frac{T_{\mathbf{i}+\mathbf{e}^d} - T_{\mathbf{i}}}{\Delta x}$$

Since we are representing thermally insulated embedded boundaries, $F^T_B = 0$. Given these fluxes, discretization of the operator is given by (1).

5.3 Viscous Stress Operator

For viscous diffusion, we first calculate the cell-centered gradient of the solution using centered differences.

$$\frac{\partial u^{d1}}{\partial x_{d2}} = \frac{u^{d1}_{\mathbf{i}+\mathbf{e}^{d2}} - u^{d1}_{\mathbf{i}-\mathbf{e}^{d2}}}{2\Delta x}$$

The face centered gradient uses this gradient for tangential gradients and differences normal gradients directly.

$$(\nabla \mathbf{u})^{d'}_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d} = \begin{cases} \frac{1}{h}(\mathbf{u}_{\mathbf{i}+\mathbf{e}^d} - \mathbf{u}_{\mathbf{i}}) \text{ if } d = d' \\ \frac{1}{2}((\nabla \mathbf{u})^{d'}_{\mathbf{i}+\mathbf{e}^d} + (\nabla \mathbf{u})^{d'}_{\mathbf{i}}) \text{ if } d \neq d' \end{cases}$$

where

$$(\nabla \mathbf{u})^d_{\mathbf{i}} = \frac{1}{2h}(\mathbf{u}_{\mathbf{i}+\mathbf{e}^d} - \mathbf{u}_{\mathbf{i}-\mathbf{e}^d}).$$

We then construct the flux at the face using the appropriate gradients

$$F^v = \mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T) + \lambda I \nabla \cdot \mathbf{u} \tag{8}$$

At the embedded boundary, we have a physical boundary condition that $\mathbf{u} = 0$. Define a local coordinate system rotated to align with the normal to the embedded boundary $\hat{n}$ and the tangent plane $(\hat{t}^1, \hat{t}^2)$. The Jacobian J of this rotational transformation is given by

$$J = \begin{pmatrix} \hat{n} \\ \hat{t}^1 \\ \hat{t}^2 \end{pmatrix}.$$

The transformation between a vector in Cartesian space (v) and a vector in rotated space (v^R) is given by

$$v^R = Jv.$$

We start by treating each component of the velocity as a scalar ϕ. To create our boundary flux, we use the Johansen extrapolation [18] to compute the normal gradient of ϕ, $(\nabla \phi^{R,n})$. We set the tangential components of the gradient of ϕ to zero (a consequence of the no-slip condition). So, in the rotated frame $(\nabla \phi)^R = (\nabla \phi^{R,n}, 0, 0)$. We then compute the Cartesian gradient of ϕ

$$\nabla \phi = J^{-1} (\nabla \phi)^R.$$

We then construct the boundary flux using (8). Given these fluxes, discretization of the operator is given by (1).

5.4 Performance Implications of Implicit Parabolic Discretization

The time step constraint for the present algorithm is given by equation 3. Since we are advancing our elliptic terms implicitly, this adds no additional time step constraint. Suppose we were to advance equation 4 explicitly:

$$\phi^{n+1} = \phi^n + \nu \Delta t_{exp} L(\phi^n) \tag{9}$$

In the absence of cut cells, the stability constraint on this method is

$$\Delta t_{exp}^{noeb} < \frac{\Delta x^2}{2D\nu}.$$

where D is the dimensionality of the problem. For the conductivity operator at constant density with constant coefficients, this relationship is exact with $\nu = \xi/(\rho C_v)$.

To illustrate the performance tradeoff in the design decision to use the implicit discretization, we compare the number of operator evaluations required

to advance the solution. Define N_{exp} to be the number of operator evaluations needed to advance the solution to a fixed time t_f using equation 9:

$$N_{exp} == \frac{t_f}{\Delta t_{exp}}.$$

Define N_{imp} to be the number of operator evaluations needed to advance the solution to a fixed time t_f using equation 6. We solve our elliptic equations using multigrid and we measure how many times the operators are applied. This puts the implicit method in the worst light possible because coarse and fine applications of the operator (through multigrid) are counted the same.

The problem in section 7 is the target application for this algorithm. When we run this problem with 4 levels of refinement for a final time of 0.7 μs (which accounts for 3 steps at the coarsest level and 48 total steps at all levels), the conductivity operator is called $N_{imp} = 600$ times. For these parameters, the time step restriction for the explicit advance is $\Delta t_{exp}^{noeb} = 2.63e - 10$), so an explicit advance would call the operator $N_{exp} = 2665$ times. For problems with less resolution or lower viscosity, this performance tradeoff can easily flip and make the explicit method more efficient. In the shock-boundary layer calculation presented in section 9, for example, $\Delta t_{exp}^{noeb} > \Delta t$, which means that the explicit parabolic advance for this case presents no addition time step constraint in the absence of embedded boundaries.

With embedded boundaries, however, the time step constraint for the explicit advance (equation 9) is far more severe. If κ_{min} is the smallest volume fraction in the domain, the true time step constraint for the explicit advance is given by

$$\Delta t_{exp} < \frac{\Delta x^2 (\kappa_{min})^{\frac{2}{D}}}{2D\nu}.$$

In this context, let us reconsider the shock-boundary layer calculation for a final time of $0.57\mu s$ (and all other parameters described in section 9), which is one time step at the coarsest level and 97 time steps at all levels. The smallest volume fraction at the finest level of this calculation is $\kappa_{min} = 3.83e - 7$, which means that $\Delta t_{exp} = 4.26e - 15$ and the number of operator evaluations required for stability is given by $N_{exp} = 1.34e8$. The number of operator evaluations we count for our implicit algorithm is $N_{imp} = 37536$. Clearly, in the presence of small cells, the implicit advance is the more efficient algorithm to advance our elliptic terms.

6 Convergence Tests

To test the convergence rate of the algorithm we start with an initial condition of flow within a sphere (or a circle in two dimensions). All tests are done using Richardson extrapolation which means that an average of a finer solution is used as an exact solution. Define A^{h-2h} to be a volume-weighted averaging operator.

Given S_f to the the set of fine volumes which cover a coarse volume $\mathbf{i}$,

$$A^{h-2h}(f)_{\mathbf{i}} = \frac{\sum_{\mathbf{i}_f \in S_f} \kappa_{\mathbf{i}_f} f_{\mathbf{i}_f}}{\sum_{\mathbf{i}_f \in S_f} \kappa_{\mathbf{i}_f}}$$

U_h is defined to be our solution on a grid with resolution h. For an exact solution U^e, we use $U^e_{2h} = A^{h-2h}(U_h)$ and the error is given by

$$\epsilon^h = U^h(t) - U^e(t). \tag{10}$$

The order of convergence ϖ is estimated by

$$\varpi = \frac{\log(\frac{||\epsilon^{2h}||}{||\epsilon^h||})}{\log(2)}. \tag{11}$$

We compute the convergence rates using compute using L_∞, L_1, and L_2 norms (all these norms are defined in [9]). The geometry of the test is a sphere with radius in the center of a domain of length L. The initial condition of the tests is given by an axisymmetric Gaussian disturbance $f(r) = exp(-30(r/r_0 - 0.5)^2)$. The maximum Mach number is set to $M = 0.5$. Define (x, y, z) to be Cartesian coordinates in a coordinate system whose origin is the sphere center. Define the distance $r = (x^2 + y^2 + z^2)^{\frac{1}{2}}$. The velocity is given by $\mathbf{u} = (-Mf(r)y/r_0, Mf(r)x/r_0)$ in two dimensions and $\mathbf{u} = (Mf(r)(z-y)/r_0, Mf(r)(x-z)/r_0, Mf(r)(y-x)/r_0)$ in three dimensions. Define v to be the magnitude of the velocity vector. The density and pressure are given by $\rho = \gamma(1 + v^2/r), p = (1 + v^2/r)$. See table 7 for other solution parameters.

Solution error is a measure of the convergence rate of the solution run to a fixed time. All refinements were advanced to a fixed time $t_f = 32\mu s$. The finest solution was advanced 64 time steps with $\Delta t = 0.5\mu s$. Each successively coarser solution was advanced half as many steps with twice as big a time step. This results in a Courant number (C_F, see equation 3) of approximately 0.1 for full cells. The results of the solution error test are given in tables 1 through 6. We demonstrate second order accuracy in all norms.

Variable	$e_{4h\to 2h}$	ϖ	$e_{2h\to h}$
(ρ)	1.103e-02	1.980e+00	2.796e-03
$(\rho\mathbf{u})_x$	2.740e-03	1.822e+00	7.748e-04
$(\rho\mathbf{u})_y$	2.740e-03	1.822e+00	7.748e-04
(ρE)	2.006e-02	1.978e+00	5.092e-03

Table 1: Solution error convergence rates in two dimensions using L_∞-norm for $h = \frac{1}{1024}$cm.

Variable	$e_{4h\to 2h}$	ϖ	$e_{2h\to h}$
(ρ)	2.519e-03	1.982e+00	6.377e-04
$(\rho\mathbf{u})_x$	3.645e-04	1.822e+00	1.031e-04
$(\rho\mathbf{u})_y$	3.645e-04	1.822e+00	1.031e-04
(ρE)	4.560e-03	1.978e+00	1.158e-03

Table 2: Solution error convergence rates in two dimensions using L_1-norm for $h = \frac{1}{1024}$ cm.

Variable	$e_{4h\to 2h}$	ϖ	$e_{2h\to h}$
(ρ)	3.900e-03	1.978e+00	9.903e-04
$(\rho\mathbf{u})_x$	6.795e-04	1.824e+00	1.920e-04
$(\rho\mathbf{u})_y$	6.795e-04	1.824e+00	1.920e-04
(ρE)	7.066e-03	1.973e+00	1.801e-03

Table 3: Solution error convergence rates in two dimensions using L_2-norm for $h = \frac{1}{1024}$cm.

Variable	$e_{4h\to 2h}$	ϖ	$e_{2h\to h}$
(ρ)	3.536e-02	1.979e+00	8.968e-03
$(\rho\mathbf{u})_x$	7.406e-03	1.814e+00	2.107e-03
$(\rho\mathbf{u})_y$	7.406e-03	1.814e+00	2.107e-03
$(\rho\mathbf{u})_z$	7.406e-03	1.814e+00	2.107e-03
(ρE)	6.887e-02	1.978e+00	1.748e-02

Table 4: Solution error convergence rates in three dimensions using L_∞-norm for $h = \frac{1}{1024}$cm.

Variable	$e_{4h\to 2h}$	ϖ	$e_{2h\to h}$
(ρ)	4.167e-03	1.986e+00	1.053e-03
$(\rho\mathbf{u})_x$	4.503e-04	1.805e+00	1.289e-04
$(\rho\mathbf{u})_y$	4.503e-04	1.805e+00	1.289e-04
$(\rho\mathbf{u})_z$	4.503e-04	1.805e+00	1.289e-04
(ρE)	7.767e-03	1.983e+00	1.965e-03

Table 5: Solution error convergence rates in three dimensions using L_1-norm for $h = \frac{1}{1024}$cm.

Variable	$e_{4h\to 2h}$	ϖ	$e_{2h\to h}$
(ρ)	7.931e-03	1.982e+00	2.007e-03
$(\rho\mathbf{u})_x$	1.060e-03	1.810e+00	3.024e-04
$(\rho\mathbf{u})_y$	1.060e-03	1.810e+00	3.024e-04
$(\rho\mathbf{u})_z$	1.060e-03	1.810e+00	3.024e-04
(ρE)	1.495e-02	1.980e+00	3.790e-03

Table 6: Solution error convergence rates in three dimensions using L_2-norm for $h = \frac{1}{1024}$cm.

μ	2.1e-5 kg/(m s)
λ	-1.4e-5 kg/(m s)
C_v	3.00e2 J/(kg K)
ξ	1.7e-2 W/(m K)
L	1.0e-2m
r_0	4.5e-3m
γ	1.4

Table 7: Initial condition setup for the convergence tests. See the text for variable definitions.

7 Capillary Tube Simulation

Our target application is the flow inside of capillary tubes in laser wakefield particle accelerators. We present a simplified version of this problem as our robustness calculation while acknowledging that other physics in these problems (such as ionization and magnetization) are very important. Refer to figure 2. The main tube (C) and the fill tubes (A) are filled with gas. The experimentalists drive the core pressure p_{core} along the axis of the tube to a high value using electrical charge, leaving the density constant. The resulting flow causes the core to expand and create a low density, high energy core. In the experiment, the laser (B) is shot through this low density core. Ideally this core should be cylindrical and have a relatively flat density profile There is some concern in the community, however, that the fill tubes can alter the core shape before the laser is shot.

We present both two and three dimensional runs that are meant to be an approximation to this problem. For a computational geometry we intersect a 200 micron diameter main tube with a perpendicular 50 micron diameter fill tube. Figure 5 shows the geometric configuration. Both are filled with argon at 1Pa, 1 kg/m^3. We initialize the core pressure to be $p_{core} = 20Pa$, leave the density constant and initialize the velocity everywhere to zero. The core diameter is 100 microns. Figure 3 shows a two-dimensional run of the plane normal to the central tube cutting through a filler tube. We plot the logarithm of density after 35 μs. Though the density profile in the core is relatively flat, the core shape is no longer circular. Figure 4 shows a two-dimensional run of the plane along the central tube cutting through a filler tube. We plot the logarithm of density after 35 μs. Though the density profile in the core is relatively flat, the core shape is once again distorted by the presence of the filler tube. Figure 6 shows an axial slice through a three-dimensional run after 50 μs and shows a similar result. To be clear, since we do not include any source terms for the effects of ionization or magnetization, this is greatly simplified approximation. We have, however, managed to show that purely hydrodynamic effects can distort the shape of the low density core.

$p_{outside}$	1.0 Pa
p_{core}	20.0 Pa
ρ_{core}	1.0 kg/m^3
ρ_{ouside}	1.0 kg/m^3
μ	2.1e-5 kg/(m s)
λ	-1.4e-5 kg/(m s)
C_v	3.00e2 J/(kg K)
ξ	1.7e-2 W/(m K)
γ	$\frac{5}{3}$

Table 8: Initial condition setup for capillary tube problem. The initial velocity is zero.

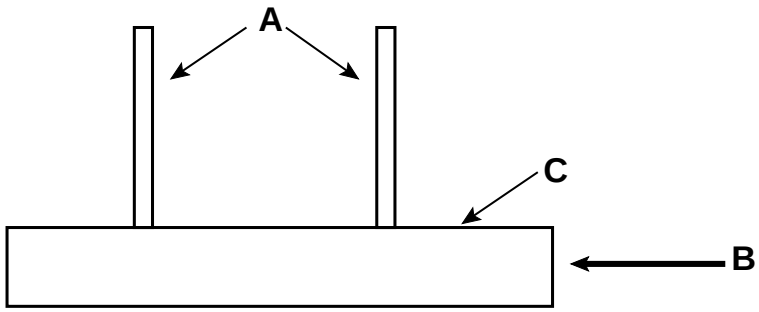

Figure 2: Illustration of wakefield accelerator. The main tube (C) and the fill tubes (A) are filled with gas. The pressure and temperature is initialized to high values up along the axis of the tube. The resulting flow causes this region to expand and create a low density, high energy core. A laser (B) is shot through this core.

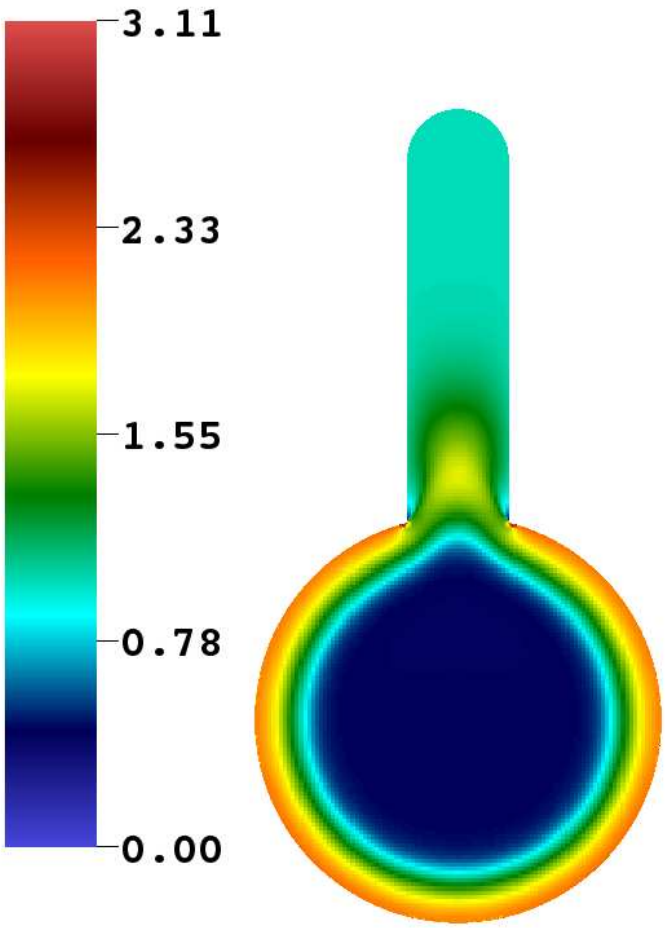

Figure 3: Two dimensional plot of $\log(\rho)$ after 35 μs. The base grid is 256^2 and there are 2 levels of refinement, all by a factor of 2. This means the effective grid resolution is 1024^2. Though the density profile in the core is relatively flat, the core shape is no longer circular.

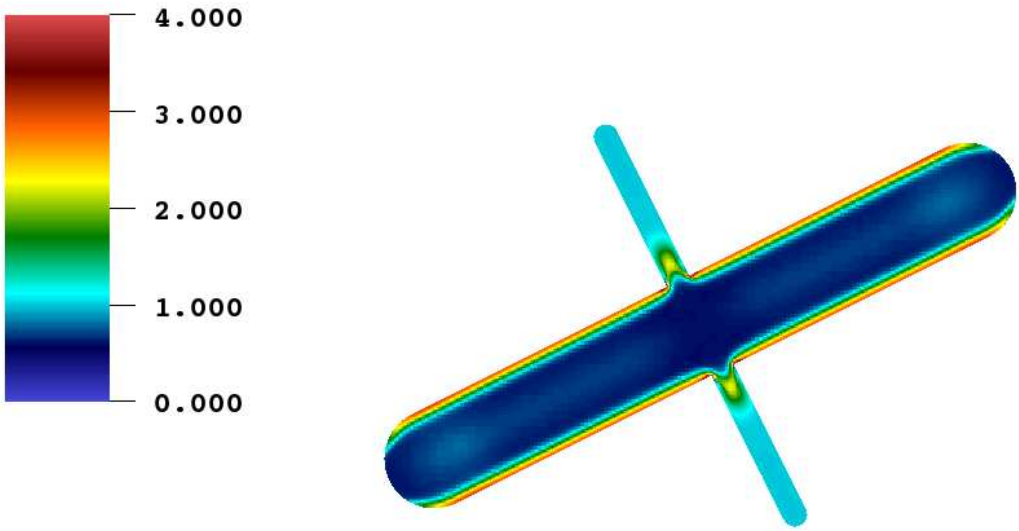

Figure 4: Two dimensional plot of $\log(\rho)$ after 35 μs. The base grid is $128x64$ and there are 3 levels of refinement, all by a factor of 2. This means the effective grid resolution is $1024x512$.Though the density profile in the core is relatively flat, the filler tube has distorted the profile.

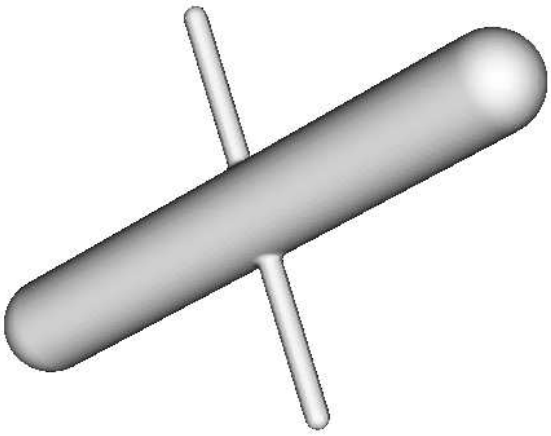

Figure 5: Geometric configuration of the three-dimensional example. The core tube's diameter is 200 microns; the filler tube's diameter is 50 microns. The core tube's length is 1.2mm; the filler tube's length is 0.85mm.

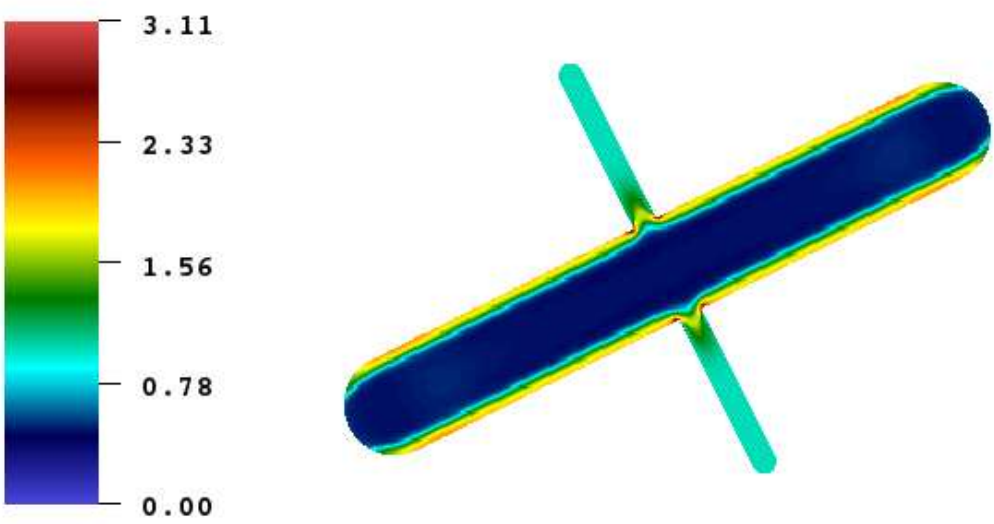

Figure 6: Axial slice through three dimensional run plot of $\log(\rho)$ after 50 μs. This is a one-level calculation with resolution $256x128x128$. Though the density profile in the core is relatively flat, the filler tube has distorted the profile.

8 Boundary Layer Similarity Solution

Given a semi-infinite flat plate in a flow field at zero incidence to the flow, the velocity profile over the plate can be calculated using a similarity solution in the absence of thermal and compressibility effects. Define x to be the distance along the plate and y to be the distance from the surface of the plate and ν to be the kinematic viscosity and $U = Mc$ is the incident velocity (refer to figure 7). The similarity variable η is given by

$$\eta = y\sqrt{\frac{U}{\nu x}} \tag{12}$$

This reduces the equations to a nonlinear ordinary differential equation, the solution of which is the familiar Blasius boundary layer. See Schlichting [26] or White [32] for a full exposition of this derivation. Charest, et al. [6], present a low-Mach number algorithm for steady state calculations. In this calculation, they present a boundary layer calculation that reproduces the behavior Blasius layer. Berger, et al. [5] present a wide variety of these calculations.

We cut a rectangular grid with a wedge of angle θ. Refer to figure 8 and table 9 for the initial and boundary conditions. The density and temperature are set to constants $\rho = \rho_0, T = T_0 = P_0/(RT_0)$. The velocity is set to $(U\cos(\theta), U\sin(\theta))$ everywhere. The velocity boundary conditions are inflow-outflow left to right (the top boundary is an outflow boundary). The boundary conditions at the embedded boundary begin as slip conditions and become no-slip to simulate the start of the semi-infinite plate (the cross-hatched region of the figure 8. Our inflow Mach number is set to $M = 0.2$ and the viscosity is set to make a Reynolds number $Re_L = 30000$. Temperature boundary conditions top and bottom are insulated; at the inflow $T = T_0$. We present two calculations, both with a base grid of $256x256$. We refine near the boundary by a factor of 16 (four levels of refinement, each factor of two) to make an effective resolution near the boundary of $4096x4096$.. The solution is allowed to run to steady state. We cast rays into the fluid at every point along the boundary within the local Reynolds number ranges $5000 < Re_x < 15000$. In figure 9, we present a scatter plot of the normalized velocity vs. the similarity variable η. We compare our results to the Blasius profile. We show good agreement with the similarity solution.

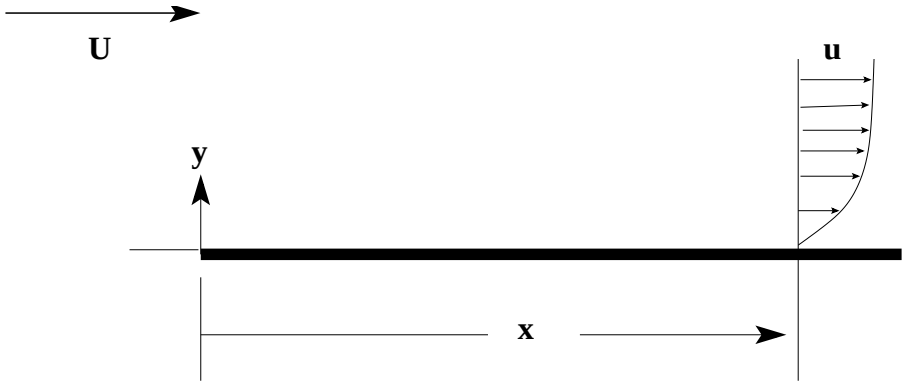

Figure 7: Formulation of semi-infinite flat plate boundary layer problem. U is the (constant) inflow velocity, x is the distance along the plate and y is the distance above the plate.

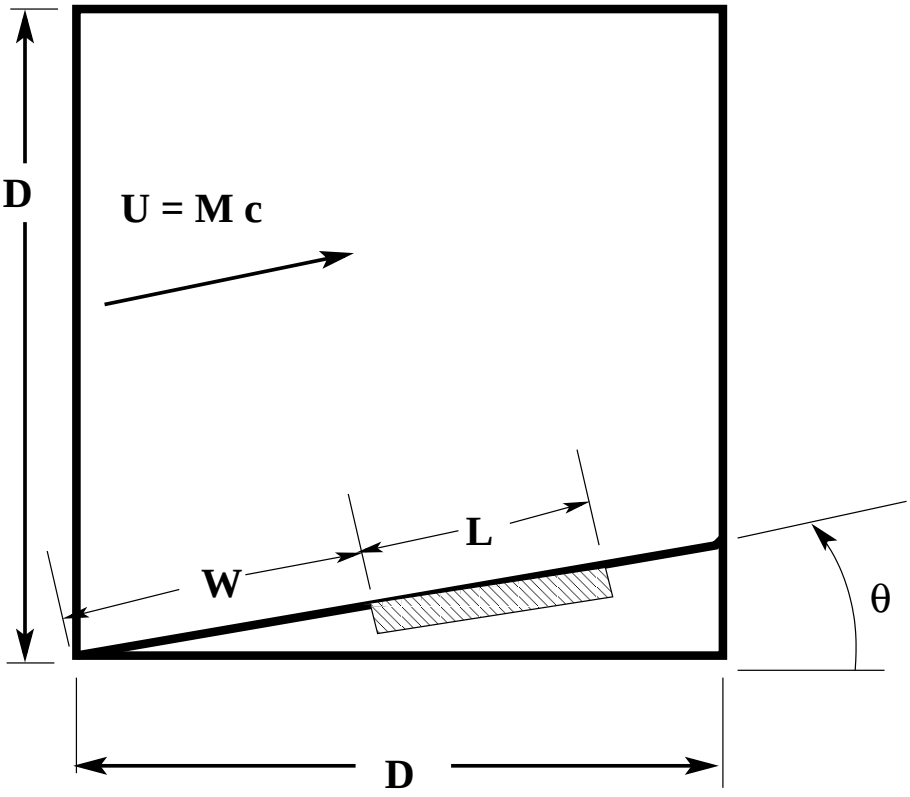

Figure 8: Initial and boundary conditions for boundary layer calculation. The density and temperature are set to constants. The velocity is set to $(U\cos(\theta), U\sin(\theta))$ everywhere. The embedded boundary cuts the grid at an angle θ from the bottom of the domain. The no-slip condition for velocity is only in effect in the crosshatched region. Values for all these quantities are given in 9.

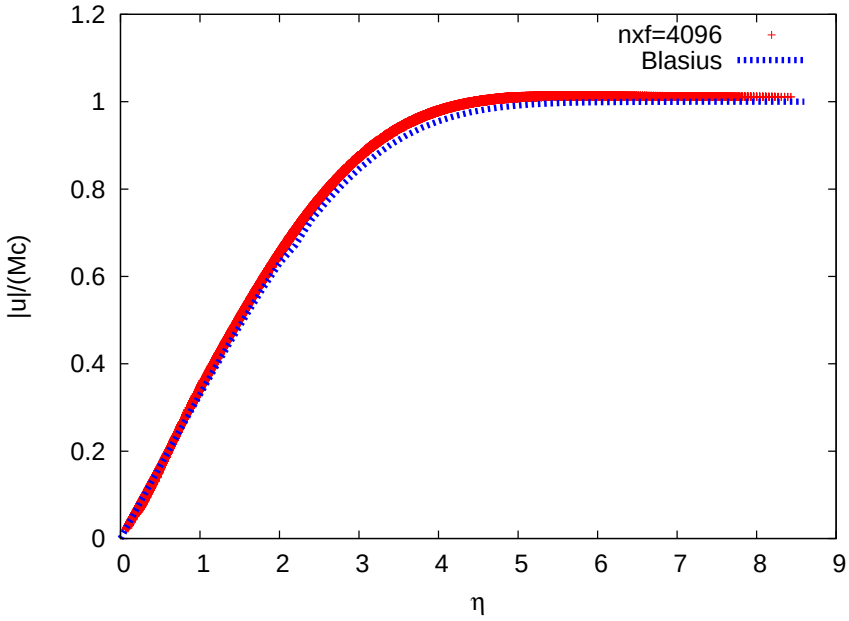

Figure 9: Results of boundary layer calculation as compared with the Blasius solution. This is a scatter plot of normalized velocity vs. the similarity variable η at two different resolutions. The magnitude of solution velocity is $|\mathbf{u}|$, c is the sound speed and the similarity variable η is defined in equation 12. The Blasius solution is in blue. The solution with effective resolution of $4096x4096$ is in red. We cast a ray from every point along the boundary where the local Reynolds number is in the range $5000 < Re_x < 15000$. We plot every point along every ray. The rays are 30 points long.

p_0	2.4e5 Pa
ρ_0	1.0 kg/m^3
L	3.0 m
D	8.0 m
W	4.0 m
M	0.2
γ	$\frac{5}{3}$
μ	1.2649e-2 kg/(m s)
λ	-8.4327e-3 kg/(m s)
C_v	5.00e2 J/(kg K)
ξ	1.7e-2 W/(m K)
Re_L	3.0e4 W/(m K)
θ	5^o

Table 9: Initial conditions for the boundary layer calculation. The velocity everywhere is initialized to $(Mc\cos(\theta), Mc\sin(\theta)$. See figure 8 for variable definitions.

9 Shock Reflection

Define M to be the Mach number of a shock propagating into a gas at rest. Glaz, et al. [14] present a comparison between inviscid calculations of shock reflections and experimental results. They show a case where viscous effects cause substantial changes in the reflection pattern. For $M = 7.1$ shock reflection from a 49 degree wedge, they show that the Mach stem is much shorter in the experiment than in an inviscid calculation. The reason cited for this difference is that the viscosity of argon varies strongly with temperature and the temperature behind the shock is quite high (the initial temperature behind the diaphragm is 10265K). The viscosity is approximated to vary with the Sutherland's power law (dynamic viscosity varies with $T^{3/2}$). We use the viscosity shown in table 10. We compute this viscosity using the highest value given in [22] and extrapolating to the initial high temperature. The specific heat and conductivity of argon are left at the room temperature values. These approximations are sufficient to illustrate the phenomenon.

Refer to figure 10 for an illustration of the initial conditions. Table 10 has the numerical values of the inputs. Both calculations have a 128x64 base grid with seven levels of adaptive mesh refinement, all by a factor of 2. This makes the effective resolution 16384x8192. All embedded boundary cells are refined to the finest level. This gives resolution at the boundary layer $h = 9.1$ microns.

Figure 11 illustrates the Mach reflection problem. Figures 13 and 12 show the viscous and inviscid calculations at the same scale after 9.61 μs. The viscous calculation shows an interesting shock-boundary layer interaction which is magnified in figure 16. The shock reflects off of the boundary layer, creating a separation bubble. This is followed by a compression (from the reflected shock) and boundary layer reattachment. For steady shocks interacting with laminar boundary layers, this is the classical lambda shock phenomenon. Both Schlichting [26] and Liepmann et al. [21] explain this in detail and include a wealth of experimental images. This is also observed (albeit barely) in the experiment presented in [14]. The interferogram they show has only two or three density contours in that region which makes the feature difficult to see.

Figures 13 and 12 clearly show that the viscous boundary layer has reduced the Mach stem substantially and a density stratification on the left. Recall that the problem is configured as a shock tube. The initial conditions are zero velocity with a discontinuity in pressure and density. As the shock moves to the right, a rarefaction fan moves to the left, producing this density variation. We show the two shock reflection patterns more closely in figures 15 and 14. For a quantitative look at this reduction, we refer to the experimental and computational results in [14]. See figure 11 for an illustration of the relevant lengths. The ratio of the Mach stem length L_M to the shock distance L_R is the quantity of interest:

$$R_m = \frac{L_m}{L_R}.$$

Glaz et al. report a value of $R_m = 0.07$ in their inviscid calculation and $R_m = 0.038$ for an experimental result (see figure 10 in [14]). Our inviscid calculation

has $R_m = 0.072$ and our viscous calculation has $R_m = 0.03$. We believe that our agreement is reasonable since not all the experimental setup information is available (the time at which the interferogram is taken, for example, is not available). For more examples of this viscous effect, see Henderson, et al. [17].

p_0	1.95e3 Pa
p_1	7.42e5 Pa
ρ_0	3.29e-2 kg/m^3
ρ_1	3.61e-1 kg/m^3
X_1	9.0e-2 m
X_2	1.0e-1 m
X_3	1.5e-1 m
Y_3	7.50e-2 m
μ	1.21e-3 kg/(m s)
λ	-8.08e-4 kg/(m s)
C_v	3.00e2 J/(kg K)
ξ	1.7e-2 W/(m K)
θ	49^o
γ	$\frac{5}{3}$

Table 10: Initial condition setup for shock reflection problem. The initial velocity is zero. See figure 10 for variable definitions.

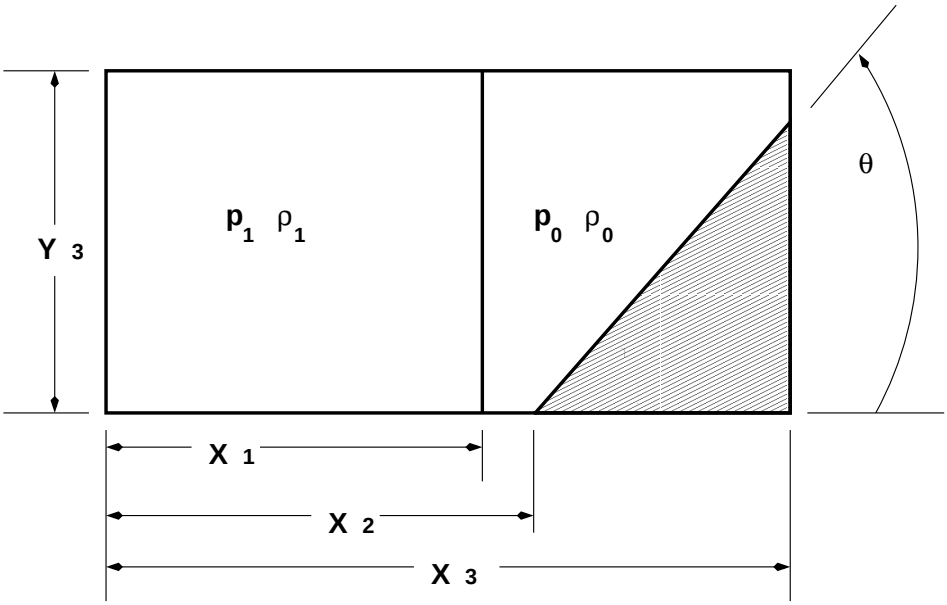

Figure 10: Shock tube setup. The initial velocity is zero. The initial pressures and densities are tailored to make a $M = 7.1$ shock. See table 10 for details.

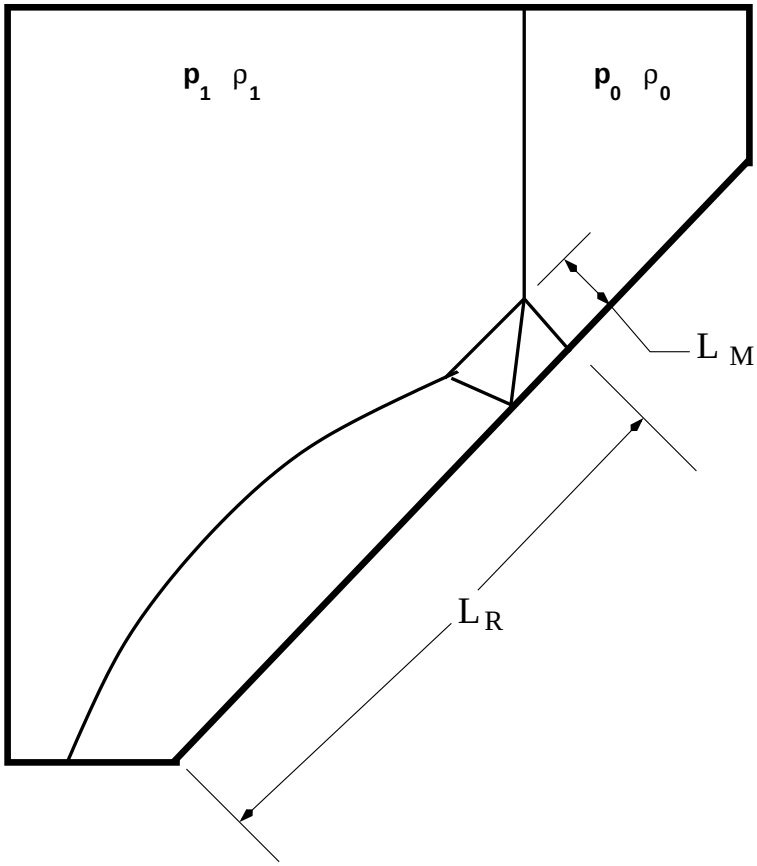

Figure 11: Shock reflection illustration. The ratio of the Mach stem length L_M to the shock distance L_R is the quantity of interest.

Figure 12: Mass density (kg/m^3) in the inviscid calculation of $M = 7.1$, 49^o shock reflection at 9.6 μs.This calculation was run 128x64 base grid with seven levels of adaptive mesh refinement, all by a factor of 2. This makes the effective resolution 16384x8192. All embedded boundary cells are refined to the finest level. This gives resolution at the boundary layer $h = 9.1$ microns.

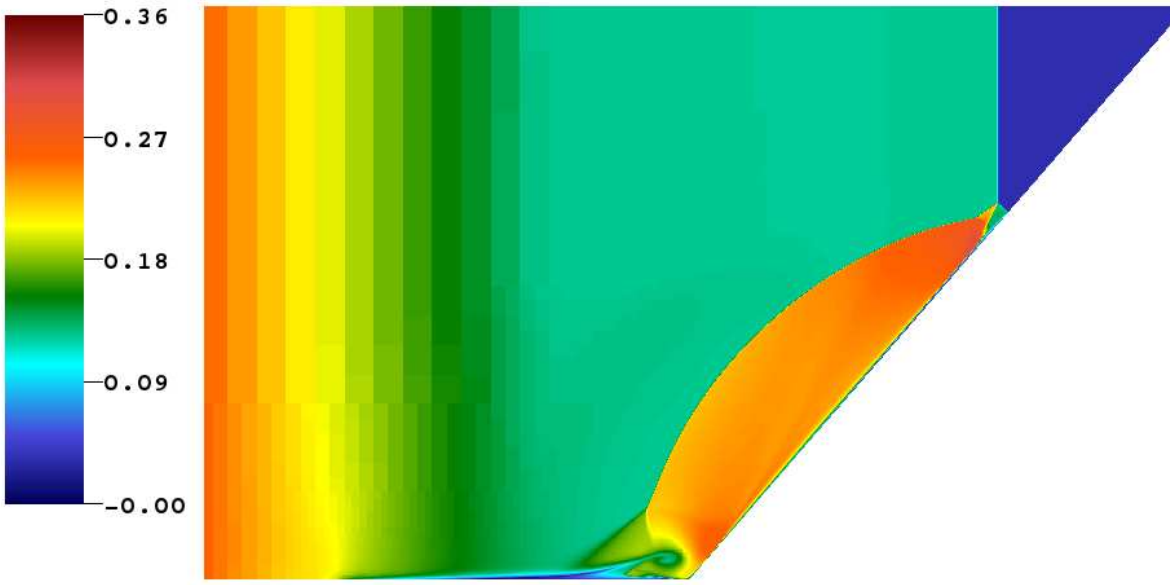

Figure 13: Mass density (kg/m^3) in the viscous calculation of $M = 7.1$, 49^o shock reflection at 9.6 μs. This calculation was run 128x64 base grid with seven levels of adaptive mesh refinement, all by a factor of 2. This makes the effective resolution 16384x8192. All embedded boundary cells are refined to the finest level. This gives resolution at the boundary layer $h = 9.1$ microns.

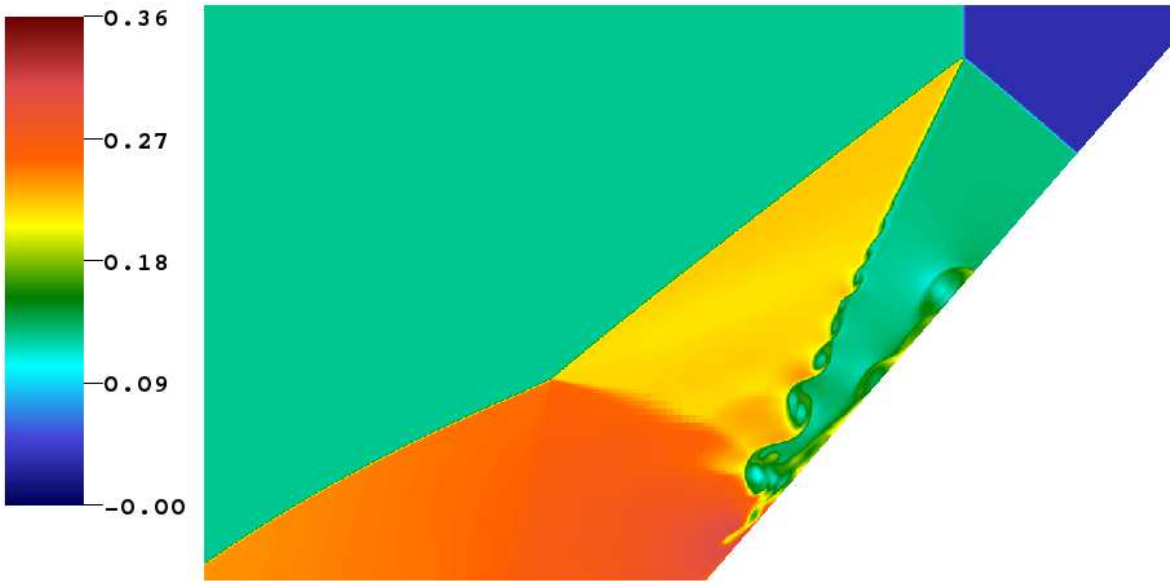

Figure 14: Mass density (kg/m^3) in the inviscid calculation of $M = 7.1$, 49^o shock reflection at 9.6 μs, zoomed in (to the same degree as figure 15) to show the reflection pattern.

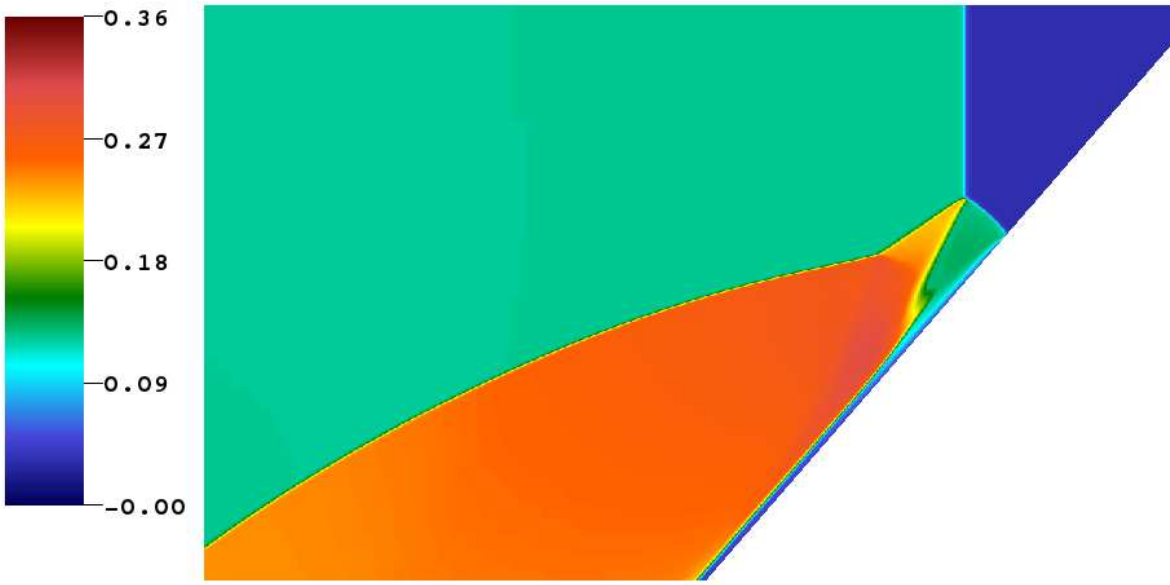

Figure 15: Mass density (kg/m^3) in the viscous calculation of $M = 7.1$, 49^o shock reflection at 9.6 μs, zoomed in (to the same degree as figure 14) to show the reflection pattern.

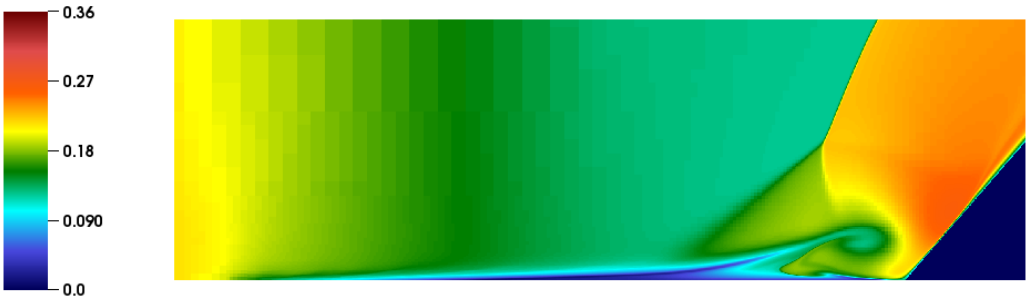

Figure 16: Magnification of the lambda shock-boundary layer pattern in the viscous calculation. Density (in kg/m^3) shown here.

10 Conclusion

We present a stable, second-order method for solving the two and three dimensional compressible Navier-Stokes equations in the presence of complex geometries. This semi-implicit method advances parabolic terms implicitly and hyperbolic terms explicitly. This allows a time step controlled by the CFL constraint associated with the hyperbolic wave speeds. We demonstrate second order accuracy for smooth initial conditions in smooth geometric configurations and robust behavior in the presence of strong discontinuities and geometric complexity that mimic the conditions in a plasma wakefield accelerator in the absence of magnetic or ionization effects. We also show good quantitative agreement with experimental results in a viscous shock reflection problem and a boundary layer problem.

11 Acknowledgements

The authors would like to thank Dr. Ann Almgren, Dr. John Bell, Dr. Marc Day, Dr. Cameron Geddes, Dr. Wim Leemans, and Dr. Daniel Martin for valuable technical discussions.

References

[1] J. B. Bell, M. J. Berger, J. S. Saltzman, and M. Welcome. A three–dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 15:127–138, 1994.

[2] J.B. Bell, P. Colella, and M.L. Welcome. Conservative front-tracking for inviscid compressible flow. In *AIAA 10th Computational Fluid Dynamics Conference. Honolulu*, pages 814–822, 1991.

[3] John B. Bell, Phillip Colella, Jeffrey A. Greenough, and Daniel L. Marcus. A multi-fluid algorithm for compressible, reacting flow. In *AIAA 95-1720, Proceedings of the 12th AIAA Computational Fluid Dynamics Conference, San Diego, CA*, 1995.

[4] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82(1):64–84, May 1989.

[5] Marsha Berger, Michael J. Aftosmis, and Steven Allmaras. Progress toward a Cartesian cut-cell method for compressible viscous flow. In *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Nashville TN*, 2012.

[6] M. R. J. Charest, C. P. T. Groth, and P. Q. Gauthier. High-order CENO finite-volume scheme for low-speed viscous flows on three-dimensional unstructured mesh. In *Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii*, 2012.

[7] I. L. Chern and P. Colella. A conservative front-tracking method for hyperbolic conservation laws. Technical Report UCRL-97200, Lawrence Livermore National Laboratory, 1987.

[8] P. Colella. Multidimensional upwind methods for hyperbolic conservation laws. *J. Comput. Phys.*, 87:171–200, 1990.

[9] P. Colella, D. T. Graves, B. Keen, and D. Modiano. A Cartesian grid embedded boundary method for hyperbolic conservation laws. *J. Comput. Phys.*, 211:347–366, 2006.

[10] Phillip Colella, Andrew Majda, and Victor Roytburd. Theoretical and numerical structure for reacting shock waves. *SIAM J/ Stat. Comput*, 7(4):1059–1080, 1986.

[11] M. S. Day and J. B. Bell. Numerical simulation of laminar reacting flows with complex chemistry. *Combust. Theory Modelling*, 4:535–556, 2000.

[12] Z. Dragojlovic, F. Najmabadi, and M. Day. "An embedded boundary method for viscous, conducting compressible flow". *J. Comp. Phys.*, 216(1):37–51, 2006.

[13] R. Ghias, R. Mittal, and H. Dong. A sharp interface immersed boundary method for compressible viscous flows. *Journal of Computational Physics*, pages 528–553, 2007.

[14] H. Glaz, P. Colella, and R. Deschambault. A numerical study of oblique shock-wave reflections with experimental comparisons. *Proc. R. Soc. London*, 398:117–149, 1985.

[15] Daniel Hartmann, Matthias Meinke, and Wolfgang Schroder. A strictly-conservative cartesian cut-cell method for compressible viscous flows on adaptive grids. *Comput. Methods Appl. Metch Engrg.*, 200:1038–1052, 2011.

[16] L. F. Henderson, K. Takayama, W. Y. Crutchfield, and S. Itabashi. The persistence of regular reflection during strong shock diffraction over rigid ramps. *Journal of Fluid Mechanics*, 431:273–296, 2001.

[17] L. F. Henderson, K. Takayama, W. Y. Crutchfield, and R. J. Virgona. The effects of thermal conductivity and viscosity of argon on shock waves diffracting over rigid ramps. *Journal of Fluid Mechanics*, 331:1–36, 1997.

[18] H. S. Johansen and P. Colella. A Cartesian grid embedded boundary method for Poisson's equation on irregular domains. *J. Comput. Phys.*, 147(2):60–85, December 1998.

[19] M.S. Kim, D. G. Jang, H.S. Uhm, S. W. Hwang, I. W. Lee, and H. Suk. Discharge characteristics of a gas-filled caplillary plama for laser wakefield acceleration. *IEEE Transactions on Plasma Science*, 39(8), 2011.

[20] W. P. Leemans, B. Nagler, A.J Gonsalves, C. S. Toth, K Nakamura, C.G.R Geddes, E. Esarey, C. B. Schroeder, and S. M. Hooker. Gev electron beams from a centimetre-scale accelerator. *Nature Physics*, 2:696–699, 2006.

[21] H. W. Liepmann, A. Roshko, and S. Dhawan. On reflection of shock waves from boundary layers. Technical Report NACA-1100, National Advisory Committee for Aeronautics, 1952.

[22] Michale N. Macrossan and Charles R. Lilley. Viscosity of argon at temperatures greater than 2000 k from measured shock thickness. *Phys. Fluids A*, 15:1363–1371, 2003.

[23] D. Martin, P. Colella, and D. T. Graves. A cell-centered adaptive projection method for the incompressible Navier-Stokes equations in three dimensions. *Journal of Computational Physics*, 227:1863–1886, 2008.

[24] R. B. Pember, J. B. Bell, P. Colella, W. Y. Crutchfield, and M. L. Welcome. An adaptive Cartesian grid method for unsteady compressible flow in irregular regions. *J. Comput. Phys.*, 120(2):278–304, September 1995.

[25] J. S. Saltzman. An unsplit 3d upwind method for hyperbolic conservation laws. *J. Comput. Phys.*, 115:153–168, 1994.

[26] Dr. Hermann Schlichting. *Boundary-Layer Theory.* McGraw-Hill, New York, NY, 1955.

[27] C. B. Schroeder, E. Esarey, C. Geddes, C. Benedetti, and W. P. Leemans. Physics considerations for laser-plasma linear colliders. *Phys. Rev. ST Accel. Beams 13*, 101301, 2010.

[28] P. Schwartz, M. Barad, P. Colella, and T. Ligocki. A Cartesian grid embedded boundary method for the heat equation and Poisson's equation in three dimensions. *Journal of Computational Physics*, 211(2):531–550, January 2006.

[29] D. J. Spence and S. M. Hooker. Investigation of a hydrogen plasma waveguide. *Physical Review E*, 63, 2000.

[30] E. Steinthorsson, D. Modiano, W.Y. Crutchfield, J.B. Bell, and P. Colella. An adaptive semi-implicit scheme for simulations of unsteady viscous compressible flow. In *AIAA Paper 95-1727-CP, in Proceedings of the 12th AIAA CFD Conference*, 1995.

[31] E.H. Twizell, A.B. Gumel, and M.A. Arigu. Second-order, l_0-stable methods for the heat equation with time-dependent boundary conditions. *Advances in Computational Mathmatics*, 6:333–352, 1996.

[32] Frank M. White. *Viscous Fluid Flow.* McGraw-Hill, New York, NY, 1974.