Adaptive mesh refinement for the compressible Euler equations using AMReX

Manlin Chawla

Cavendish Laboratory, Department of Physics, J J Thomson Avenue, Cambridge. CB3 0HE

1. Introduction

The Navier-Stokes equations form the basis of many mathematical models describing physical problems involving fluid flow and have a wide range of applications, ranging from the field of hydraulics, atmospheric flow modelling, aerospace, to simulation of physiological fluid flows in the cardiovascular system [1]. Under the assumption that viscosity $\tau = 0$, heat conductivity $\kappa = 0$ and other body forces are absent $\mathbf{f} = 0$, the Navier Stokes equations reduce to the inviscid compressible Euler equations given by [2],

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0$$
$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v} + \mathbf{I}p) = 0$$
$$\frac{\partial E}{\partial t} + \nabla \cdot [(E + p)\mathbf{v}] = 0$$

The inviscid compressible Euler equations form a system of non-linear hyperbolic PDE's comprising of three conservation laws for mass (density), momentum and total energy. In the above equations, ρ is the fluid density, $\mathbf{v}$ is the velocity vector, p is the fluid pressure, E is the total energy. This paper focuses on techniques related to solving these equations numerically using the AMReX [3] software package.

Due to the non-linearity of the governing equations, discontinuities and steep gradients can be generated in the solution of these PDE's even when starting with smooth initial conditions [2]. Over the years, many conservative shock-capturing finite volume methods have been developed to solve the Euler equations with an ideal gas, including centred schemes such as FORCE and Riemann problem-based schemes such as HLL [4], HLLC [5] etc. In general, these first-order schemes are combined with a method such as flux limiting or slope limiting to produce high-resolution TVD numerical methods [2]. These methods provide a combination of second-order accuracy and monotonicity. Furthermore, they can be combined with adaptive mesh refinement techniques to allow for more accurate capturing of flow features that are difficult to resolve with a coarse mesh [6].

Adaptive mesh refinement (AMR) is a widely used technique that is applied to the numerical solution of PDE's in order to increase the efficiency with which a numerical method can be used to obtain a solution for a given effective resolution or conversely to obtain a solution with a higher resolution at a particular computational cost [6]. AMR dynamically alters the resolution of a computational mesh as the numerical simulation progresses. The regions that require increased resolution are identified by criteria's based on flow properties such as high gradients or errors in each cell at each timestep. In doing so it tracks and refines flow features of interest without having to waste computational resources on refining the entire domain.

Many variants of AMR have been developed but this paper focuses on a hierarchical AMR approach which is used in the AMReX framework [7][3]. A high-resolution Riemann-problem based finite volume method scheme has been implemented and used as the basis to numerically solve the compressible Euler equations in the AMReX framework and to investigate the effects of adaptive mesh refinement.

The structure of the rest of this paper is as follows. The next section provides a brief introduction to the Euler equations, the properties of the solution and the numerical methods used to solve them. In particular, a description is given of one such high-resolution scheme, the dimensionally split MUSCL-Hancock with an approximate HLLC Riemann Solver which has been used to solve the compressible Euler equations in this study. Section 3 provides an overview of different AMR methods available and describes the AMR algorithm used in the AMReX framework. In Section 4, a series of test cases are presented in 1D and 2D to illustrate the accuracy and efficiency achieved with AMR over calculations carried out on a single, uniform, Cartesian grid. The paper is completed with some brief conclusions.

2. Mathematical and Numerical Formulation

A. Euler Equations

The one-dimensional inviscid compressible Euler equations are given by

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{f}(\boldsymbol{U})}{\partial x} = 0 \tag{1}$$

where the conservative variables and flux vectors are given by

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho v \\ E \end{bmatrix}, \quad \boldsymbol{f}(\boldsymbol{U}) = \begin{bmatrix} \rho v \\ \rho v^2 + p \\ v(E+p) \end{bmatrix} \tag{2}$$

Here ρ is the density, v is the velocity, p is the fluid pressure and E is the total energy given by a combination of the internal and kinetic energy

$$E = \rho\epsilon + \frac{1}{2}\rho v^2$$

where ϵ is the specific internal energy. The system is closed using the ideal gas equation of state given by

$$p = (\gamma - 1)\rho\epsilon$$

where γ is the ratio of specific heats.

The Riemann problem for the Euler equations (1)-(2) describes an initial value problem with piece-wise constant data with a single discontinuity centred at $x = x_0$ [2],

$$\boldsymbol{U}(x, t=0) = \begin{cases} \boldsymbol{U_L} & \text{if} \quad x \leq x_0 \\ \boldsymbol{U_R} & \text{if} \quad x > x_0 \end{cases}$$

In order to determine how many waves are present in the Riemann problem solution, we consider the Euler equations in quasi-linear form [8]

$$\frac{\partial \boldsymbol{U}}{\partial t} + \mathbf{A}(\mathbf{U})\frac{\partial \boldsymbol{U}}{\partial x} = 0$$

where the coefficient matrix $\mathbf{A}(\mathbf{U})$ is the Jacobian matrix $\partial \boldsymbol{f}/\partial \boldsymbol{U}$. Before explicitly defining $\mathbf{A}$, a new quantity referred to as the sound speed c_s needs to be introduced and pressure needs to be expressed in terms of the conserved variables which can be done using the equation of state. These are given by

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho v^2\right)$$

$$c_s = \sqrt{\frac{\gamma p}{\rho}}$$

As $\rho > 0$ and $p > 0$ are always satisfied for an ideal gas the quantity c_s is always real. Now we can evaluate the partial derivatives $\partial f_i/\partial u_j$ and express these in terms of the sound speed c_s to get the Jacobian matrix of the following form [2]

$$\mathbf{A}(\mathbf{U}) = \begin{bmatrix} 0 & 1 & 0 \\ \frac{1}{2}(\gamma-3)v^2 & (3-\gamma)v & \gamma - 1 \\ \frac{1}{2}(\gamma-2)v^3 - \frac{c_s^2 v}{\gamma - 1} & \frac{3-2\gamma}{2}v^2 + \frac{c_s^2}{\gamma-1} & \gamma v \end{bmatrix}$$

We can compute the eigenvalues of this matrix using the characteristic polynomial $|\mathbf{A} - \lambda\mathbf{I}| = 0$. The three eigenvalues obtained are

$$\lambda_1 = v - c_s$$
$$\lambda_2 = v$$
$$\lambda_3 = v + c_s$$

By considering the equation $\mathbf{AK} = \lambda\mathbf{K}$, the corresponding right eigenvectors are [8]

$$\mathbf{K}^{(1)} = \begin{bmatrix} 1 \\ -c_s/\rho \\ c_s^2 \end{bmatrix} \mathbf{K}^{(2)} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \mathbf{K}^{(3)} = \begin{bmatrix} 1 \\ c_s/\rho \\ c_s^2 \end{bmatrix}$$

These are important results, firstly we have shown that the eigenvalues are real and distinct meaning that the Euler equations are strictly hyperbolic, secondly, the eigenvectors $\mathbf{K}^{(1)}$, $\mathbf{K}^{(2)}$, $\mathbf{K}^{(3)}$ form a complete set of linearly independent eigenvectors. There are three waves associated with the three eigenvectors in the solution of the Riemann problem [8]. The wave associated with λ_2 is a contact discontinuity across which p and v remain constant but ρ jumps discontinuously. The waves associated with $\lambda_1 = v - c_s$ and $\lambda_3 = v + c_s$ can be rarefraction or shock waves. A rarefraction wave is a smooth wave across which ρ, v and p change continuously. A shock wave is a discontinuous wave across which all quantities ρ, v, p have a discontinuous jump.

B. Numerical Methods for solving the Euler Equations

As the Euler equation solutions can contain discontinuities such as shock waves, a conservative numerical method

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n - \frac{\Delta t}{\Delta x}(\mathbf{F}_{i+\frac{1}{2}}^n(\mathbf{U}) - \mathbf{F}_{i-\frac{1}{2}}^n(\mathbf{U}))$$

is required in order to capture the position of the shock correctly [8]. Over the years many conservative shock-capturing finite volume methods have been developed for numerically solving the Euler equations, the two main classes of such methods are referred to as centred schemes and Riemann-problem based schemes [2].

One example of a centred scheme is the FORCE scheme which is simply the average of the Lax-Friedrichs and Richtmyer schemes,

$$\mathbf{F}_{i+\frac{1}{2}}^{FORCE} = \frac{1}{2}\left(\mathbf{F}_{i+\frac{1}{2}}^{LF} + \mathbf{F}_{i+\frac{1}{2}}^{RI}\right)$$

The Lax Friedrich flux is defined as

$$\mathbf{F}_{i+1/2}^{LF} = \frac{1}{2}\frac{\Delta x}{\Delta t}\left(\mathbf{U}_i^n - \mathbf{U}_{i+1}^n\right) + \frac{1}{2}\left(\mathbf{f}\left(\mathbf{U}_{i+1}^n\right) + \mathbf{f}\left(\mathbf{U}_i^n\right)\right)$$

and the Richtmyer flux is defined as

$$\mathbf{U}_{i+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left(\mathbf{U}_i^n + \mathbf{U}_{i+1}^n\right) - \frac{1}{2}\frac{\Delta t}{\Delta x}\left(\mathbf{f}\left(\mathbf{U}_{i+1}^n\right) - \mathbf{f}\left(\mathbf{U}_i^n\right)\right)$$

$$\mathbf{F}_{i+\frac{1}{2}}^{RI} = \mathbf{f}\left(\mathbf{U}_{i+\frac{1}{2}}^{n+\frac{1}{2}}\right)$$

An advantage of this method is that it's comprised of a first-order scheme that is diffusive and a second-order scheme that has oscillations but the resulting scheme is a first-order scheme that is less diffusive and doesn't exhibit oscillatory behaviour.Furthermore, centred schemes like FORCE do not explicitly use wave propagation information, making them easier to implement. However, discontinuities are not captured as sharply [2].

Alternatively, Godunov methods have remained one of the most popular methods for solving the Euler equations as they are able to provide a solution with less diffusion and dissipation than centred schemes. The fundamental notion of a Godunov scheme is to solve a Riemann problem between two neighbouring cells and use this to compute the inter-cell flux $F_{i+\frac{1}{2}}$. From considering the nature of the characteristics of the Euler equations, at $x_{i+\frac{1}{2}}$ the solution to the Riemann problem is constant as it is either on one side of a shock wave or in the middle of a rarefraction [2] [8].

For the Euler equations with an ideal gas equation of state, an exact solution to the Riemann problem exists, however, this involves an iterative procedure thus making it computationally expensive. In order to improve computational efficiency, approximate Riemann solvers are often used. An example of such a method is the HLLC solver, devised by Toro, Spruce and Speares [5]. Given two constant states $\mathbf{u}_L$ and $\mathbf{u}_R$ that form a Riemann problem, this method makes the assumption that there are three waves in the solution moving with a speed S_L, S^* and S_R respectively separating the $x-t$ domain into four constant states. The two intermediate states $\mathbf{u}_L^{\mathrm{HLLC}}$ and $\mathbf{u}_R^{\mathrm{HLLC}}$ are separated by a contact discontinuity. Across the intermediate states the quantities

$$v_L^* = v_R^* = v^* = S^*, \qquad p_L^* = p_R^* = p^*$$

remain constant which is the case across a contact discontinuity. Then from considering these expressions and the jump conditions, the HLLC inter-cell flux is given by [2]

$$\mathbf{f}_{i+1/2}^{\mathrm{HLLC}} = \begin{cases} \mathbf{f}_L & 0 \le S_L \\ \mathbf{f}_L + S_L\left(\mathbf{u}_L^{\mathrm{HLLC}} - \mathbf{u}_L\right) & S_L < 0 \le S^* \\ \mathbf{f}_R + S_R\left(\mathbf{u}_R^{\mathrm{HLLC}} - \mathbf{u}_R\right) & S^* < 0 \le S_R \\ \mathbf{f}_R & 0 > S_R \end{cases}$$

where

$$S^* = \frac{p_R - p_L + \rho_L v_L\left(S_L - v_L\right) - \rho_R v_R\left(S_R - v_R\right)}{\rho_L\left(S_L - v_L\right) - \rho_R\left(S_R - v_R\right)}$$

The intermediate states $\mathbf{u}_K^{\mathrm{HLLC}}$ for $K \in [L, R]$ are given by

$$\mathbf{u}_K^{\mathrm{HLLC}} = \rho_K\left(\frac{S_K - v_K}{S_K - S_*}\right)\begin{bmatrix} 1 \\ S_* \\ \frac{E_K}{\rho_K} + \left(S_* - v_K\right)\left[S_* + \frac{p_K}{\rho_K\left(S_K - v_K\right)}\right] \end{bmatrix}$$

Several choices for defining the wave speed estimates S_L and S_R are given in the literature by Davis [9], Einfeldt [10] and others. A common choice for defining these quantities which provide good accuracy properties and is simple to implement is as follows [8]

$$\begin{aligned} S^+ &= \max\left(|v_L| + c_{s,L}, |v_R| + c_{s,R}\right) \\ S_L &= -S^+, \quad S_R = S^+ \end{aligned} \tag{3}$$

An important consideration for the stability of Godunov type methods is that the time step needs to be chosen such that propagation of the information from one cell boundary does not interact with other cell boundaries. The maximum possible stable time step that can be taken is given by

$$\Delta t = C\frac{\Delta x}{a_{\max}}$$

where

$$a_{\max} = \max_i\left(|\mathbf{v}_i| + c_s\right)$$

C. Second-Order Extension

As we assume piecewise-constant data in each cell, Godunov methods on their own are first-order and consequently are diffusive and smear out discontinuities. We now describe how to combine the MUSCL-Hancock approach with a Godunov method to achieve second-order accuracy.

The first step in the MUSCL-Hancock approach is data

reconstruction. In this step, we consider piece-wise constant data in each cell and introduce slope reconstruction across the cell boundaries. Using first-order reconstruction, we replace the piece-wise constant data with a piece-wise linear function. The boundary extrapolated values are given by

$$\mathbf{U}_i^L = \mathbf{U}_i^n - \frac{1}{2}\Delta_i \quad \mathbf{U}_i^R = \mathbf{U}_i^n + \frac{1}{2}\Delta_i$$

The slope vector is defined as

$$\Delta_i = \frac{1}{2}(1+\omega)\Delta_{i-\frac{1}{2}} + \frac{1}{2}(1-\omega)\Delta_{i+\frac{1}{2}}$$

$$\Delta_{i-\frac{1}{2}} \equiv \mathbf{U}_i^n - \mathbf{U}_{i-1}^n; \quad \Delta_{i+\frac{1}{2}} \equiv \mathbf{U}_{i+1}^n - \mathbf{U}_i^n$$

$\omega \in [-1, 1]$ and is often taken to be 0. It is important to consider what happens at discontinuities, at these points the derivative does not exist and this can cause spurious oscillations in regions of steep gradients [2]. In the TVD version of this scheme, slope limiting is used in the data reconstruction step. The slope vectors are replaced with limited slopes given by

$$\bar{\Delta}_i = \xi_i(r)\Delta_i$$

where r is defined as

$$r = \frac{u_i^n - u_{i-1}^n}{u_{i+1}^n - u_i^n}$$

and $\xi_i(r)$ is a slope limiter function. Some common slope limiters are [8]:

$$\xi_{\text{MINBEE}}(r) = \begin{cases} 0 & r \le 0 \\ r & 0 \le r \le 1 \\ \min\{1, \xi_R(r)\} & r \ge 1 \end{cases}$$

$$\xi_{\text{VANALBADA}}(r) = \begin{cases} 0 & r \le 0 \\ \min\left\{\frac{r(1+r)}{1+r^2}, \xi_R(r)\right\} & r \ge 0 \end{cases}$$

$$\xi_{\text{VANLEER}}(r) = \begin{cases} 0 & r \le 0 \\ \min\left\{\frac{2r}{1+r}\xi_R(r)\right\} & r \ge 0 \end{cases}$$

$$\xi_{\text{SUPERBEE}}(r) = \begin{cases} 0 & r \le 0 \\ 2r & 0 \le r \le \frac{1}{2} \\ 1 & \frac{1}{2} \le r \le 1 \\ \min\{r, \xi_R(r), 2\} & r \ge 1 \end{cases}$$

where

$$\xi_R(r) = \frac{2}{1+r}$$

In the case of large changes, the slope limiter reverts to piece-wise constant data. Alternatively for smooth changes, the reconstructed data is used. In between these two extremes, a spectrum of limiting is used [2].

Next, the boundary extrapolated values are evolved by $\frac{\Delta t}{2}$ as

$$\overline{\mathbf{U}}_i^L = \mathbf{U}_i^L - \frac{1}{2}\frac{\Delta t}{\Delta x}\left[\mathbf{F}\left(\mathbf{U}_i^R\right) - \mathbf{F}\left(\mathbf{U}_i^L\right)\right]$$

$$\overline{\mathbf{U}}_i^R = \mathbf{U}_i^R - \frac{1}{2}\frac{\Delta t}{\Delta x}\left[\mathbf{F}\left(\mathbf{U}_i^R\right) - \mathbf{F}\left(\mathbf{U}_i^L\right)\right]$$

Now we can use these values to compute the inter-cell flux by solving a Riemann problem at the interface, using the HLLC flux (or another Godunov method)

$$\mathbf{U}_L \equiv \overline{\mathbf{U}}_i^R \quad \mathbf{U}_R \equiv \overline{\mathbf{U}}_{i+1}^L$$

$$\mathbf{F}_{i+\frac{1}{2}} = \mathbf{F}_{i+\frac{1}{2}}^{\text{HLLC}}(\overline{\mathbf{U}}_i^R, \overline{\mathbf{U}}_{i+1}^L)$$

Finally, this inter-cell flux is used in the finite volume update formula

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n - \frac{\Delta t}{\Delta x}(\mathbf{F}_{i+\frac{1}{2}}^n(\mathbf{U}) - \mathbf{F}_{i-\frac{1}{2}}^n(\mathbf{U}))$$

D. Dimensional Splitting

In this study, we consider solutions to both the 1D and 2D Euler equations. The numerical methods described earlier can be extended to multiple dimensions by using an operator splitting approach, where we update the x-direction first and then update the y-direction

$$\overline{\mathbf{U}}_{i,j} = \mathbf{U}_{i,j}^n - \frac{\Delta t}{\Delta x}(\mathbf{F}_{i+\frac{1}{2},j}^n(\mathbf{U}) - \mathbf{F}_{i-\frac{1}{2},j}^n(\mathbf{U}))$$

$$\mathbf{U}_{i,j}^n = \overline{\mathbf{U}}_{i,j} - \frac{\Delta t}{\Delta y}(\mathbf{G}_{i,j+\frac{1}{2}}^n(\overline{\mathbf{U}}) - \mathbf{G}_{i,j-\frac{1}{2}}^n(\overline{\mathbf{U}}))$$

This approach is consistent with the underlying PDE (in the limit $\Delta x \to 0$, $\Delta y \to 0$) and is useful as it allows for the diagonal propagation of information over a single time step [2]. It should be noted that dimensional splitting is first-order accurate in time. This approach allows us to consider a one-dimensional problem in the x and y directions. The new stable time step is now given by

$$\Delta t = C\frac{\min(\Delta x, \Delta y)}{a_{\max}}$$

where the maximal wave speed in the system is given by

$$a_{\max} = \max_{i,j}\left(|\mathbf{v}_{i,j}| + c_{s,i,j}\right)$$

3. Adaptive Mesh Refinement and AMReX

In many compressible flow simulations, a high-resolution is often required to capture particular flow features accurately, otherwise resolving such features with a coarse mesh may lead to incorrect numerical results over a larger portion of the domain [6]. Although global refinement will increase the number of cells in regions of interest it results in a waste of computational resources elsewhere. Instead, we turn to adaptive mesh refinement (AMR), a technique that tracks features of interest and dynamically alters the resolution of a computational mesh as the numerical simulation progresses. This consequently leads to significant savings in CPU-time and memory for a given effective resolution.

In general mesh refinement strategies can be categorized into three types [11] [12], r-refinement, h-refinement and p-refinement. In r-refinement, existing cells in the computational domain are re-organised such that mesh resolution is increased locally in regions of interest, whilst simultaneously decreased elsewhere in the grid [11]. Specific examples of r-refinement include mesh stretching and the use of curvilinear meshes. In p-refinement the spacing is held fixed, but the order of the polynomial approximation within each element is changed in order to capture the solution more closely [12]. In h-refinement, the resolution is increased locally either by adding new computational points within the grid structure or by overlaying additional cells of finer resolution on top of a coarse base grid. The former is referred to as the quad-tree approach, the latter is referred to as hierarchical AMR and is the approach used in AMReX [7][3]. The rest of this section will focus on giving a description of the AMR algorithms used within AMReX.

A. Grid Structure

Hierarchical AMR is an approach based on the method developed by Berger and Oliger [13], and improved by Berger and Colella [14] and other researchers such as Quirk [15]. The method employs a hierarchical set of grids, at the bottom of the hierarchy lies a coarse grid G_0 which completely covers the computational domain. The resolution of the base grid is not altered during the simulation. Additional finer grid levels (G_1, G_2, ...) consist of patches that are nested within the internal boundaries of this underlying grid to increase the resolution locally. The size, location and number of these patches change over time but they are constructed such that on a given level, no two patches overlap [7]. Any number of levels of refinement is allowed but in practice, a maximum is usually set, depending on the particular simulation. In this investigation, we consider refinement ratios of $\times 2$ between consecutive levels.

B. Tagging Criteria

The additional levels of refinement are placed in areas that have been tagged by the refinement criteria of the model. The type of refinement criteria and thresholds values are set by the user at the time of running the simulation, independently for each problem. For the idealized simulations considered in this report one of the following refinement criteria's have been used to track discontinuities such as shocks and contact waves:

- $\rho_{i,j} >$
- $\frac{\mathrm{abs}(\rho_{i+1,j}-\rho_{i,j})}{\Delta x} > \eta \quad$ or $\quad \frac{\mathrm{abs}(\rho_{i,j+1}-\rho_{i,j})}{\Delta y} > \eta$

where η is a specified threshold. The tagged cells are augmented by additional buffer cells because of the CFL condition [6][16].

C. Clustering

Once the cells requiring refinement are tagged, these are clustered into patches based on a recursive algorithm developed by Bell et al [17]. The algorithm works as follows; starting off with a patch covering the entire domain, any rows or columns containing no tags are identified and the patch is sliced here. The efficiency of each patch (defined as =flagged cells/total cells) is calculated. If the efficiency is smaller than a pre-specified threshold α then the algorithm is applied again. The clustering algorithm is terminated when the efficiency criteria have been met for all patches or a specified minimum width is reached. The value of α is used to provide a balance between creating too many small patches that contain only flagged cells (which increases algorithmic overhead) or having few large patches that contain redundant untagged cells which end up being refined (and increases computational effort) [16].

D. Boundary Conditions

Numerical methods such as MUSCL-Hancock require two ghost cells around the edge of the computational domain. Each sub-patch within AMReX is surrounded by a layer of ghost cells to enforce the (internal) boundary conditions which are primed before each grid integration. If the ghost cells surrounding a patch cover the interior of a neighbouring patch on the same level then the values are directly copied. If not, then the ghost cells are interpolated from the underlying coarse mesh.

Any ghost cells that coincide with the domain boundary are set according to the physical boundary conditions [7][6].

E. Sub-Cycling

The CFL condition imposes the restrictions such that a smaller maximum stable time step needs to be used on the finer levels. Using this smaller timestep across all levels will introduce more diffusion and degrade the accuracy of the solution on the coarse mesh [6][16]. Instead, sub-cycling is used in AMReX to avoid unnecessarily prohibitive restrictions on the timestep.

The sequence of operations used to advance the solution using sub-cycling is best explained using an example. Consider a two-dimensional computational domain with a three level-grid hierarchy G_0, G_1, G_2 with a refinement ratio $\times 2$ between each level, in both the x and y direction [6].

Firstly, on the coarsest level integrate G_0 with the coarse grid time step Δt. On the next finer level G_1, interpolate the boundaries from G_0 and integrate G_1 with the timestep $\frac{\Delta t}{2}$. Now, on G_2, interpolate the boundaries from G_1 and integrate G_2 with its time step $\frac{\Delta t}{4}$ twice to advance the solution to the same time level as G_1. The next step is to project the data from G_2 to G_1 whilst maintaining conservation. This is done by taking cell averages in G_2 and updating the solution and fluxes in the cells they cover in G_1 [6]. The procedure is repeated – advance G_1 again with a timestep $\frac{\Delta t}{2}$. Advance G_2 by $\frac{\Delta t}{4}$ twice and project G_2 data to G_1. Finally, project G_1 data to G_0. Now all levels have been advanced to the same time level whilst using an appropriate time step on each refinement level.

F. Conservation

It is necessary to apply a flux correction to the coarse grid cells which lie at coarse-fine boundary interfaces. The flux correction is simply the difference between the two sets of fluxes computed by the coarse cell and the adjacent fine cells over a given time step. Applying the correction ensures that the numerical flux entering and leaving the cells balances and that conservation is preserved by the algorithm [16].

4. Numerical Experiments

In this section, we present various validation tests to ensure the correct working of our numerical method for solving the inviscid compressible Euler equations in both 1D and 2D within the AMReX framework. We also investigate the effects of AMR on accuracy and efficiency quantitively. As mentioned earlier the numerical method used is the MUSCL-Hancock HLLC approximate Riemann solver with wave speed estimates for S_L and S_R given by (3).

A. One dimensional tests

We begin by validating our underlying numerical method for solving the one-dimensional Euler equations. For this, we consider Toro's 5 benchmark tests [8], which have exact solutions against which we can compare our results. The initial data for these tests in terms of primitive variables is given in Table 1 along with the final time the simulation is run until.

Test		ρ	u	p	time
Toro1	$x < 0.5$	1.0	0.0	1.0	0.25
	$x > 0.5$	0.125	0.0	0.1	
Toro2	$x < 0.5$	1.0	-2.0	0.4	0.15
	$x > 0.5$	1.0	2.0	0.4	
Toro3	$x < 0.5$	1.0	0.0	1000	0.012
	$x > 0.5$	1.0	0.0	0.01	
Toro4	$x < 0.5$	1.0	0.0	0.01	0.035
	$x > 0.5$	1.0	0.0	100.0	
Toro5	$x < 0.5$	5.99924	19.5975	460.894	0.035
	$x > 0.5$	5.99242	-6.19633	46.0950	

Table 1: Initial data for Toro's Tests

The solutions of the five tests cover a range of scenarios [8]. The solution for Test 1 consists of a left rarefraction, a contact and right shock. Test 2's solution consists of two strong rarefractions and a contact discontinuity. Test 3's solution consists of a left rarefraction, a contact and a right shock. Test 4's solution consists of a left shock, a contact discontinuity and a right rarefraction. The solution for Test 5 consists of a left-facing shock and a right travelling contact discontinuity and shock wave.

All five tests are defined on a domain $x \in [0, 1]$, assuming an ideal gas with $\gamma = 1.4$ using a CFL number $C = 0.9$, using the Minbee slope limiter and transmissive boundary conditions. Three different resolutions are used namely $N = 128, 256, 512$ where N is the number of finite volume cells used. The results for density, velocity, pressure and internal energy for each test are given in figures 1-4, 7-10 and Appendix A.40-A.43, B.46-B.49, C.52-C.55. All of the results show good agreement with the exact solutions. In places where they do not such as the internal energy plot in figure 10, this is expected. This is because the low-density value caused by the two strong rarefraction causes unphysical states which the Riemann solver struggles with [18].

One-dimensional test: Toro Test 1

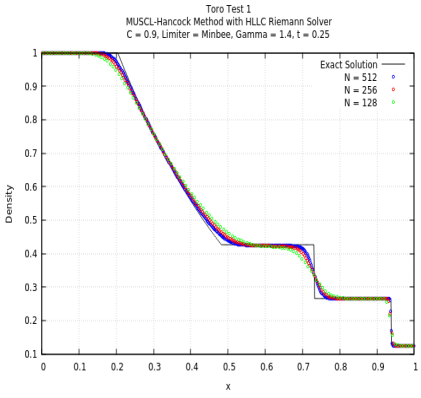

Figure 1: Density

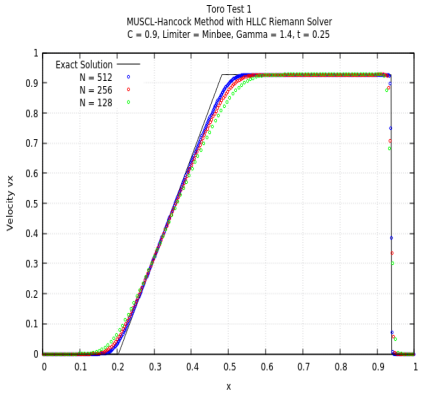

Figure 2: Velocity

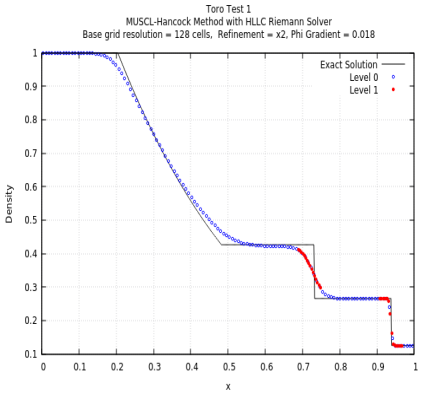

Figure 5: Density with AMR x2

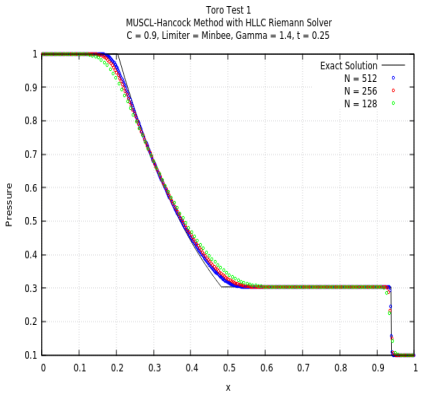

Figure 3: Pressure

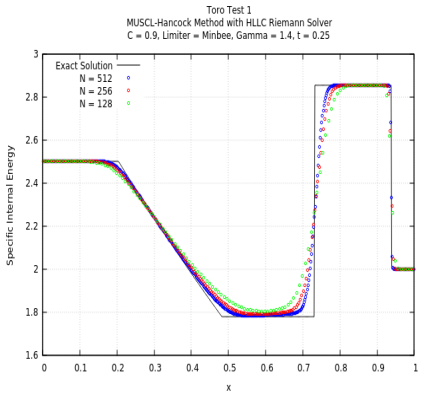

Figure 4: Internal Energy

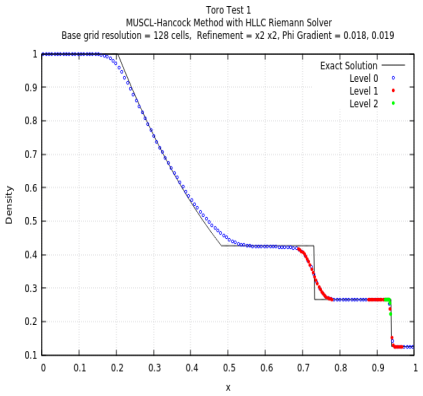

Figure 6: Density with AMR x2 x2

One-dimensional test: Toro Test 2

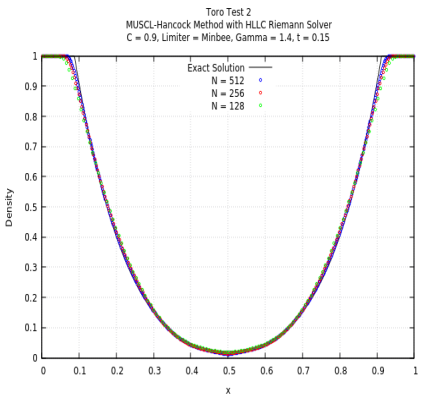

Figure 7: Density

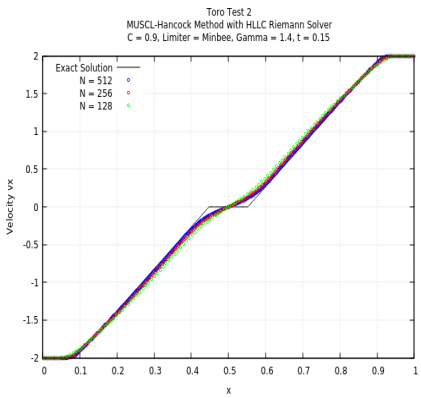

Figure 8: Velocity

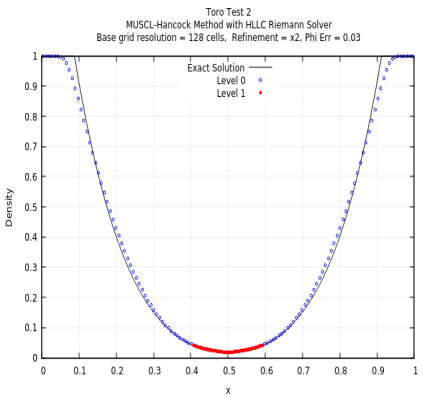

Figure 11: Density with AMR x2

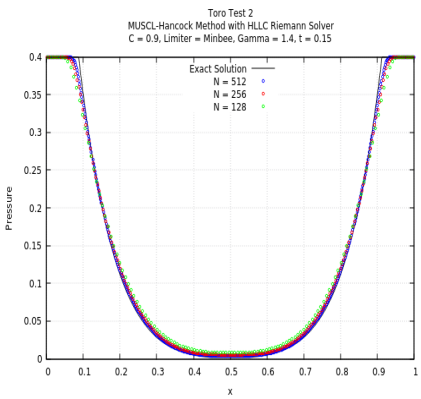

Figure 9: Pressure

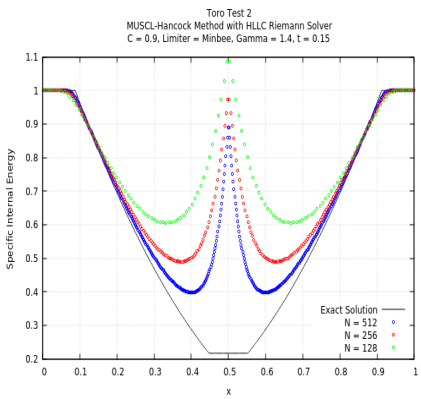

Figure 10: Internal Energy

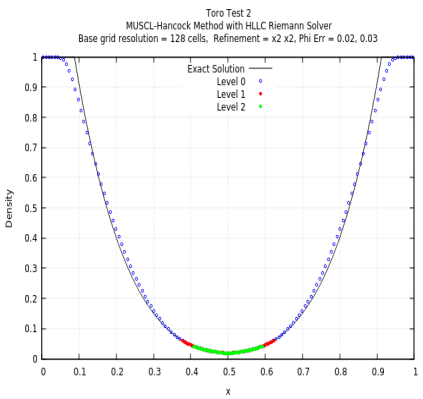

Figure 12: Density with AMR x2 x2

No AMR		
Test	Resolution	L_1 (5 dp)
Test 1	128	0.01601
	256	0.01026
	512	0.00653
Test 2	128	0.01447
	256	0.00964
	512	0.00636
Test 3	128	0.19665
	256	0.13981
	512	0.09629
Test 4	128	0.19827
	256	0.12707
	512	0.09113
Test 5	128	0.84346
	256	0.53365
	512	0.34676

With AMR				
Test	Resolution	Refinement Level	Refinement Parameters	L_1 (5 dp)
Test 1	128	-	-	0.01601
	128	$\times 2$	0.018	0.01054
	128	$\times 2 \times 2$	0.018, 0.019	0.00843
Test 2	128	-	-	0.01447
	128	$\times 2$	0.03	0.01383
	128	$\times 2 \times 2$	0.02, 0.03	0.00950
Test 3	128	-	-	0.19665
	128	$\times 2$	0.6	0.14179
	128	$\times 2 \times 2$	0.6, 0.61	0.11245
Test 4	128	-	-	0.19827
	128	$\times 2$	0.6	0.13970
	128	$\times 2 \times 2$	0.6, 0.65	0.11355
Test 5	128	-	-	0.84346
	128	$\times 2$	0.55	0.54084
	128	$\times 2 \times 2$	0.55, 0.56	0.36089

Table 2a (left): L_1 error between numerical solution for density and exact solution for each Toro Test at varying resolutions.
Table 2b (right): L_1 error for each Toro Test with 1 and 2 levels of adaptive mesh refinement.

Table 2a shows the errors for each test when compared to its exact solutions, the error norms have been calculated using

$$\text{Error}_{L1} = \Delta x \sum_i |u(x_i, t^n) - u_i^n|$$

It is clear to see both qualitatively and quantitatively that as the resolution increases (i.e. $\Delta x \to 0$), the solution of the numerical method converges towards the exact solution with increasing accuracy.

The tests have been repeated using AMR to resolve sharp features such as contact discontinuities and shocks. A coarse grid resolution of $N = 128$ cells has been used and each test has been refined first using 1 level and then 2 levels of refinement. This allows for a direct comparison of the accuracy between the non-refined tests and refined tests that have the same effective resolution. The refinement criteria for Tests 1, 3, 4 and 5 are based on the density gradient

$$\frac{\text{abs}(\rho_{i+1,j} - \rho_{i,j})}{\Delta x} > \eta$$

Due to the nature of the solution of Test 2, the refinement criteria for this test is based on the value of density which satisfy

$$\rho_{i,j} < \eta$$

The particular thresholds used for each test are problem dependant, these are listed in Table 2b. Figures 5-6, 11-12 and Appendix A.44-A.45, B.50-B.51, C.56-C.57 show the regions of the mesh which have been refined for each test. The convergence analysis for the refined tests has been repeated and is shown in Table 2b. The error measures show that when using a coarse mesh, increasing the effective resolution using AMR improves the accuracy of the solution. Furthermore, in each case the AMR test has a slightly higher error but more importantly similar accuracy compared to the corresponding resolution in the unigrid test. This is to be expected as AMR is targeted, so only certain areas have a higher effective resolution whereas the rest of the domain still has a lower resolution. From this convergence analysis, we can draw the conclusion that AMR is an effective tool in improving the accuracy of solutions on coarse meshes.

B. Two-dimensional tests

In order to validate our 2D numerical scheme we consider Toro's 5 tests [8] defined on a domain $[0, 1] \times [0, 1]$. These are effectively one-dimensional tests run in both the x and y direction to ensure the code's results are correctly obtained and direction independent. For the x-direction version of these tests, the left and right initial states are as defined in Table 1 with $v_y = 0$. For the y-direction version of the tests the x and y components of velocity are switched and the left and right initial states correspond to the discontinuity $y < 0.5$ and $y > 0.5$ respectively. Each of the x-direction and y-direction tests have been repeated with AMR. The same refinement

Two-dimensional tests - TORO TEST 1

x-direction test

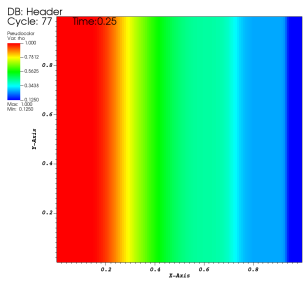

Figure 13: Density

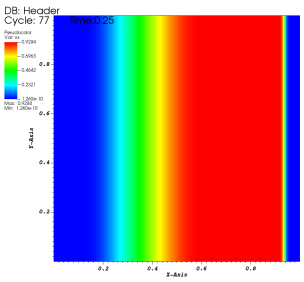

Figure 14: Velocity v_x

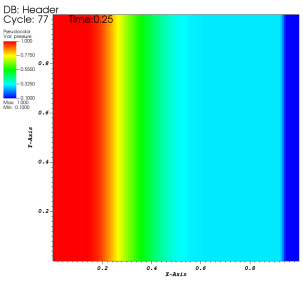

Figure 15: Pressure

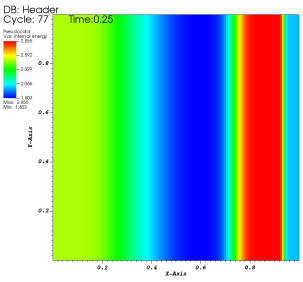

Figure 16: Internal Energy

y-direction test

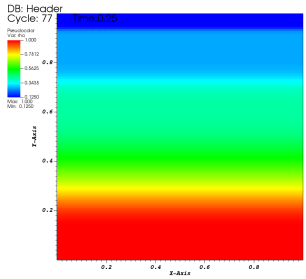

Figure 17: Density

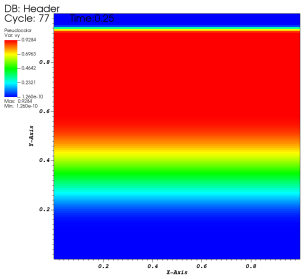

Figure 18: Velocity v_y

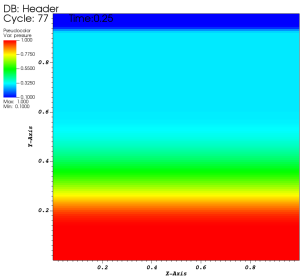

Figure 19: Pressure

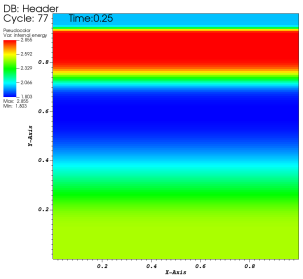

Figure 20: Internal Energy

Diagonal test

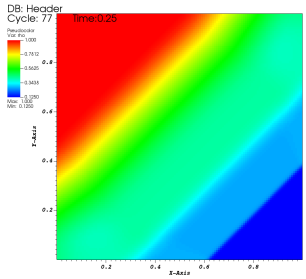

Figure 21: Density

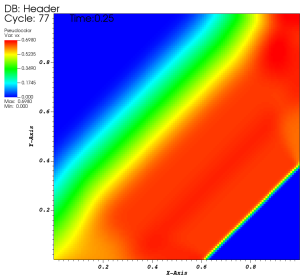

Figure 22: Velocity v_x

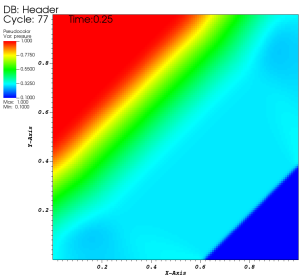

Figure 23: Pressure

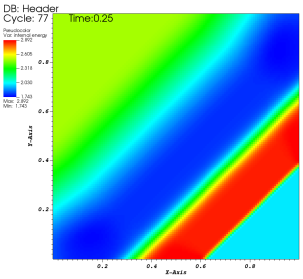

Figure 24: Internal Energy

AMR test

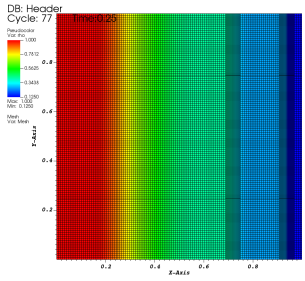

Figure 25: Density, x-direction with $\times 2$ AMR

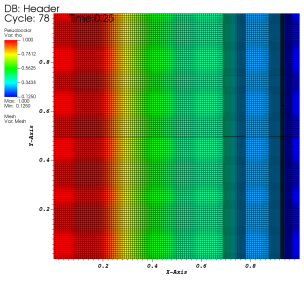

Figure 26: Density, x-direction with $\times 2 \times 2$ AMR

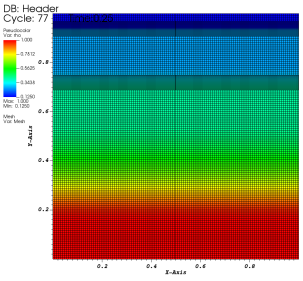

Figure 27: Density, y-direction with $\times 2$ AMR

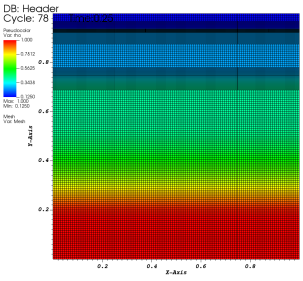

Figure 28: Density, y-direction with $\times 2 \times 2$ AMR

criteria and thresholds have been used as in the 1D tests but now extended to both the x and y direction. We also consider a modification of these tests where the initial discontinuity follows the line $y = x$ which is not perpendicular to either coordinate axis. In all versions of these tests, we assume an ideal gas with $\gamma = 1.4$, using a CFL number $C = 0.9$, with $N = 128 \times 128$ cells. Transmissive boundary conditions are used and the slope limiter used is Minbee. These are all of the same parameters used in the one-dimensional tests which allows for direct comparison between results. Note: The results for Test 1 are shown in Figure 13-28, the results for Test 2-5 can be found in appendix D-G.

The results in the x-direction are identical to those obtained in the one-dimensional case. It is clear to see that the x and y direction tests produce identical results when appropriately aligned. In the results for the diagonally aligned test, we see the same behaviour across most of the domain except at the boundaries/corners. This is attributed to the transmissive boundary conditions used. When using transmissive boundary conditions, the ghost cells are updated by extrapolating the values from cells within the physical domain where the flow is orthogonal and aligned. Whereas now the flow of motion is diagonal, this causes extra behaviour occurring at the corners of the domain when the waves in the simulation interact with the boundaries.

C. *Two Dimensional Timing Exercises*

In this section we investigate the effects AMR has on simulation time. We consider Toro's Cylindrical Explosion test [8] defined on a domain $[0, 2] \times [0, 2]$. The initial data consists of a region inside a circle centred at $(1, 1)$ with a radius $R = 0.4$ and the region outside this circle:

$$\left.\begin{array}{ll} \rho^{ins} = 1.0, & \rho^{out} = 0.125, \\ v_x^{ins} = 0.0, & v_x^{out} = 0.0, \\ v_y^{ins} = 0.0, & v_y^{out} = 0.0, \\ p^{ins} = 1.0, & p^{out} = 0.1. \end{array}\right\} \tag{4}$$

The test is run to a final time $t = 0.25$, with $\gamma = 1.4$, $C = 0.9$, using the Minbee slope limiter and transmissive boundary conditions. The solution consists of a circular shock wave and contact discontinuity travelling away from the centre and a circular rarefraction travelling towards $(1, 1)$. From figures 29-34 it is evident that increasing the resolution increases the accuracy of the solution.

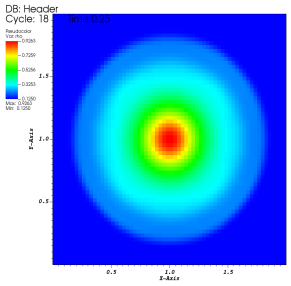

Figure 29: Density, $N = 64 \times 64$

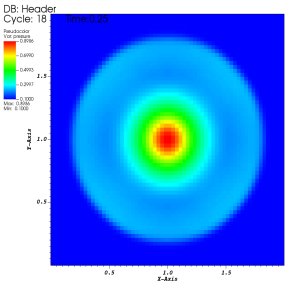

Figure 30: Pressure, $N = 64 \times 64$

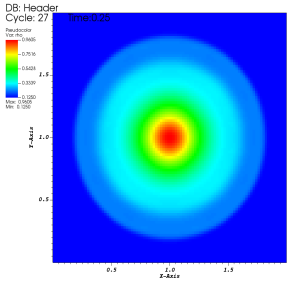

Figure 31: Density, $N = 96 \times 96$

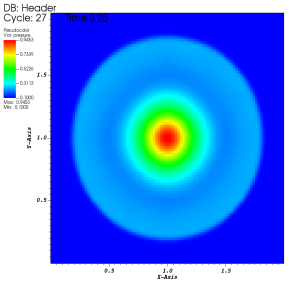

Figure 32: Pressure, $N = 96 \times 96$

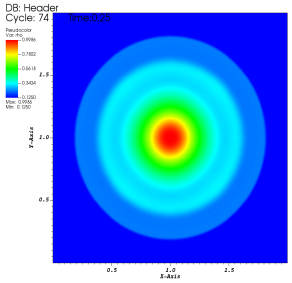

Figure 33: Density, $N = 256 \times 256$

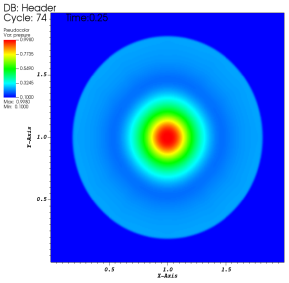

Figure 34: Pressure, $N = 256 \times 256$

Figure 29 - 34: Toro's Cylindrical Explosion at three different resolutions

Table 3-4 show the simulation timings for this test run with and without AMR, using various levels of parallelization. To ensure fairness, each test was run 3 times on the Hydra04 machine in the Lab of Scientific Computing and the average was taken to get a final runtime. In the non-refined tests, each simulation was run at three different resolutions namely $N = 256 \times 256$, 384×384, and 512×512 on a single CPU, the tests have been repeated with the AMReX code parallelized using MPI, OpenMP and a combination of MPI cores and OpenMP threads. AMReX has inbuilt functionality for parallelization of the code with MPI. In our code, OpenMP has been used to parallelise parts of the operations in the functions used to advance the solution and to estimate the time step.

No Mesh Refinement				
1 CPU Tests				
Resolution	Cores		Time (2dp) / s	Speedup
256×256	1		122.05	-
384×384	1		393.88	-
512×512	1		937.79	-

MPI Tests				
Resolution	MPI Cores	OMP Threads	Time (2dp) / s	Speedup
256×256	1	-	122.05	1.00
256×256	2	-	61.48	1.99
256×256	4	-	33.28	3.67
256×256	8	-	21.56	5.66
256×256	16	-	16.73	7.29
384×384	1	-	393.88	1.00
384×384	2	-	200.50	1.96
384×384	4	-	107.22	3.67
384×384	8	-	63.64	6.19
384×384	16	-	42.00	9.38
512×512	1	-	937.79	1.00
512×512	2	-	477.39	1.96
512×512	4	-	245.65	3.82
512×512	8	-	142.89	6.56
512×512	16	-	84.95	11.04

OpenMP Tests				
Resolution	Cores	OMP Threads	Time (2dp) / s	Speedup
256×256	1	1	122.05	1.00
256×256	1	2	62.19	1.96
256×256	1	4	33.89	3.60
256×256	1	8	21.16	5.77
256×256	1	16	14.05	8.69
384×384	1	1	393.88	1.00
384×384	1	2	209.38	1.88
384×384	1	4	106.06	3.61
384×384	1	8	65.90	5.98
384×384	1	16	41.81	9.42
512×512	1	1	937.79	1.00
512×512	1	2	479.94	1.95
512×512	1	4	250.35	3.75
512×512	1	8	152.61	6.15
512×512	1	16	93.79	10.00

MPI + OpenMP Tests				
Resolution	MPI Cores	OMP Threads	Time (2dp) / s	Speedup
256×256	1	1	122.05	1.00
256×256	2	8	36.91	3.31
256×256	4	4	12.90	9.46
256×256	8	2	14.26	8.56
384×384	1	1	393.88	1.00
384×384	2	8	90.14	4.37
384×384	4	4	38.15	10.32
384×384	8	2	38.35	10.27
512×512	1	1	937.79	1.00
512×512	2	8	187.50	5.00
512×512	4	4	81.88	11.45
512×512	8	2	83.54	11.23

Table 3: Average run-time of Toro's Cylindrical Explosion simulation with no adaptive mesh refinement, at various levels of parallelization. All tests have been run on Hydra04.

Adaptive Mesh Refinement						
1 CPU Tests						
Resolution	Refinement	Phi Gradient	Cores	OMP Threads	Time (2dp) / s	Speedup
512×512	-	-	1	-	937.79	1.00
256×256	$\times 2$	0.015	1	-	387.96	2.42
MPI Tests						
Resolution	Refinement	Phi Gradient	Cores	OMP Threads	Time (2dp) / s	Speedup
512×512	-	-	1	-	937.79	1.00
256×256	$\times 2$	0.015	2	-	211.34	4.44
256×256	$\times 2$	0.015	4	-	118.92	7.89
256×256	$\times 2$	0.015	8	-	81.46	11.51
256×256	$\times 2$	0.015	16	-	64.08	14.63
OpenMP Tests						
Resolution	Refinement	Phi Gradient	Cores	OMP Threads	Time (2dp) / s	Speedup
512×512	-	-	1	1	937.79	1.00
256×256	$\times 2$	0.015	1	2	232.04	4.04
256×256	$\times 2$	0.015	1	4	146.79	6.39
256×256	$\times 2$	0.015	1	8	104.06	9.01
256×256	$\times 2$	0.015	1	16	95.07	9.86
MPI + OpenMP Tests						
Resolution	Refinement	Phi Gradient	Cores	OMP Threads	Time (2dp) / s	Speedup
512×512	-	-	1	1	937.79	1.00
256×256	$\times 2$	0.015	2	8	157.80	5.94
256×256	$\times 2$	0.015	4	4	62.19	15.08
256×256	$\times 2$	0.015	8	2	62.67	14.96

Table 4: Average run-time of Toro's Cylindrical Explosion simulation with adaptive mesh refinement, at various levels of paralellisation. All tests have been run on Hydra04.

Adaptive Mesh Refinement - Adjusting Parameters						
1 CPU Tests - Adjusting Buffer Cells						
Resolution	Refinement	Phi Gradient	Blocking Factor	Cores	Time (2dp) / s	Speedup
512×512	-	-	8	1	937.79	1.00
256×256	$\times 2$	0.015	2	1	352.27	2.66
256×256	$\times 2$	0.015	4	1	358.00	2.62
256×256	$\times 2$	0.015	8	1	387.96	2.42
256×256	$\times 2$	0.015	16	1	450.38	2.08
1 CPU Tests - Adjusting Max Grid Size						
Resolution	Refinement	Phi Gradient	Max grid Size	Cores	Time (2dp) / s	Speedup
512×512	-	-	8	1	937.79	1.00
256×256	$\times 2$	0.015	8	1	513.70	1.83
256×256	$\times 2$	0.015	16	1	387.96	2.42
256×256	$\times 2$	0.015	32	1	359.84	2.61
256×256	$\times 2$	0.015	64	1	353.17	2.66
256×256	$\times 2$	0.015	128	1	359.72	2.61
1 CPU Tests - Adjusting Phi Gradient						
Resolution	Refinement	Phi Gradient		Cores	Time (2dp) / s	Speedup
512×512	-	-		1	937.79	1.00
256×256	$\times 2$	0.1		1	129.04	7.27
256×256	$\times 2$	0.05		1	226.36	4.14
256×256	$\times 2$	0.025		1	284.67	3.29
256×256	$\times 2$	0.02		1	334.75	2.80
256×256	$\times 2$	0.015		1	387.96	2.42
256×256	$\times 2$	0.01		1	432.50	2.17
256×256	$\times 2$	0.005		1	474.40	1.98

Table 5: Average run-time of Toro's Cylindrical Explosion simulation with adaptive mesh refinement, when varying blocking factor, max grid size and phi gradient parameters.

As expected, as the resolution increases the run-time for each simulation also increases. In the case of the lowest resolution $N = 256 \times 256$, the simulation took just over 2 minutes to run, with the highest resolution $N = 512 \times 512$ the simulation took almost 16 minutes to run. The same tests were repeated using multiple CPU's with MPI. Again as expected, using MPI parallelization causes a significant speed up but a greater speedup is achieved (approx. $\times 11$ faster) at the highest resolution. An interesting thing to note is that at lower resolutions there is still an increase in speedup but the speedup achieved slows down/plateaus as more cores are used. This is because the amount of work each processor has to do decreases and the overhead associated with communication as a proportion of run-time increases. The same trends are repeated when OpenMP is used to parallelize the code. For the non-refined tests, the fastest speedup overall was achieved by using a combination of 4 OpenMP cores and 4 MPI threads.

The next set of tests were run with AMR. The resolution of the coarse grid is taken to be $N = 256 \times 256$ and one level of refinement is used with a refinement ratio $\times 2$ and $\mathrm{blockingfactor} = 8$ to achieve an effective resolution of $N = 512 \times 512$. This allows us to compare the timings of the AMR simulations with $N = 512 \times 512$ unigrid simulation. The refinement criteria used is the density gradient over 0.015. After some preliminary tests, this threshold was found to be the best at capturing the contact discontinuity and shock waves whilst keeping other regions of the domain mostly unrefined.

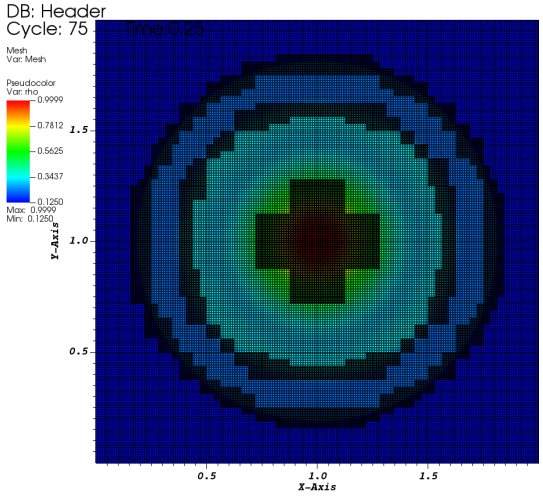

Figure 35: Density with AMR, phi gradient = 0.015

The effects of AMR are apparent even on a single CPU, a speedup of approximately $\times 2.4$ is achieved on a single CPU. Again when using AMR with parallelization we see the same trend where using more cores/threads gives a greater speedup but beyond a certain point, the marginal speed-up achieved plateau's because of the overhead with communication involved. At 16 cores the speed up achieved using MPI and mesh refinement is greater than the equivalent MPI unigrid test. With OpenMP, we get a similar speed up with AMR compared to running the simulation with 512×512 cells with a unigrid. Overall it is evident that significant savings in run-times can be achieved by combining AMR and parallelization. In particular, for Toro's cylindrical explosion, the highest speed up achieved is approximately $\times 15$ reducing the time taken for the simulation to run from over 15 minutes to just over a minute.

For completeness the effects of adjusting parameters relating to AMR on simulation time have been considered, see table 5. From the numerical experiments, it was found that increasing the blocking factor increases the simulation run-time. This makes sense as the blocking factor represents the number of buffer cells added when a cell is tagged. When using one level of refinement, really only 2 buffer cells are required to ensure the CFL condition is satisfied. Any more buffer cells used means more refinement and thus computational effort is involved.

The maximum grid size parameter enforces that no patch is longer in any direction than this value [7]. Decreasing this parameter means that during the clustering algorithm many smaller patches are created which increases the overhead involved. This is reflected in the timings, for smaller values the simulation takes longer to run. However, beyond a certain point increasing this parameter has little effect on the timings.

Finally, we consider the effects of changing the magnitude of phi gradient used. A range of values have been considered

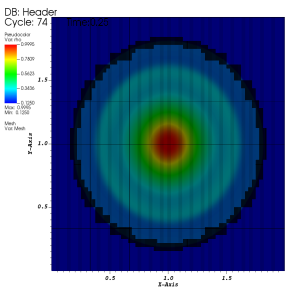

Figure 36: phi gradient > 0.05

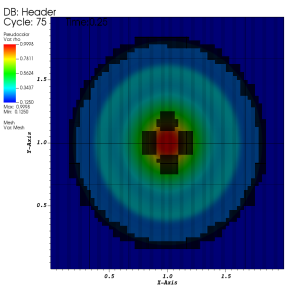

Figure 37: phi gradient > 0.02

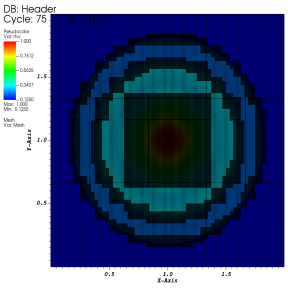

Figure 38: phi gradient > 0.01

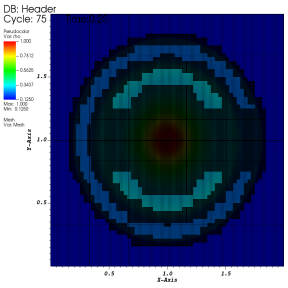

Figure 39: phi gradient > 0.005

Figure 36-39: Range of phi gradient values considered

As expected, run-times increase as more refinement is used. In order to gain the maximum benefit of AMR, it is important to choose a refinement criterion that targets regions of interest rather than simply covering the majority of the domain with refinement.

5. Conclusion

In this paper, a dimensionally split MUSCL-Hancock HLLC Riemann solver has been used to numerically solve the inviscid compressible Euler equations in 1D and 2D. The method has been combined with adaptive mesh refinement within the AMReX software framework to refine sharp features within the simulation. Throughout this study, we have investigated the effects of adaptive mesh refinement on accuracy and performance on simulations of Toro's standard 5 tests. These tests cover a wide range of scenarios and the results confirmed, that using AMR has no detrimental effect on the accuracy of the solution compared to unigrid simulations. Furthermore, the 1D and 2D tests demonstrate that AMR is able to track features of interest and reproduce the results of unigrid high resolution runs with fewer grid cells. However, AMR is a powerful technique and allows solutions to be obtained with much greater efficiency. The improvement is particularly dramatic when combined with parallelization. For example, when simulating Toro's cylindrical explosion speedups of approximately 15x were achieved with AMR.

The success of AMR heavily depends on the refinement criteria used to tag the cell and the choice of thresholds. Throughout this study, we have considered idealised simulations where the flow features of interest are clearly defined. The thresholds used for refinement have been tailored for each problem and are specified by the user at run-time. Several factors improve the effectiveness of AMR however no clear strategy currently exists for determining the best general refinement criteria. Further work involves developing algorithms that optimize for the best parameters to use for each simulation.

Overall we have seen that significant savings in CPU-time and memory can be achieved for a given effective resolution when solving the compressible Euler equations. It remains open to investigate whether these benefits extend to other situations. Further work involves exploring AMR used in conjunction with other numerical methods such as those mentioned in Section 2, on different geometries other than a regular cartesian mesh, with other refinement ratios other than powers of 2, with more than 2 levels of refinement and even on other physical problems that exhibit different flow characteristics and behaviours.

Appendix A. One-dimensional test: Toro Test 3

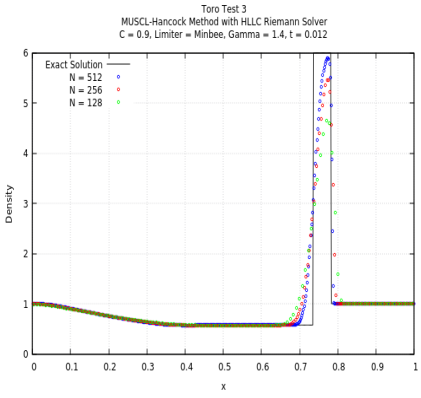

Figure A.40: Density

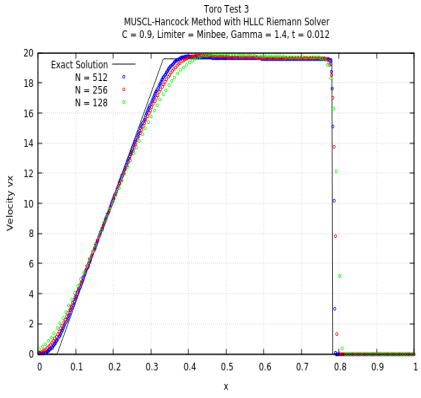

Figure A.41: Velocity

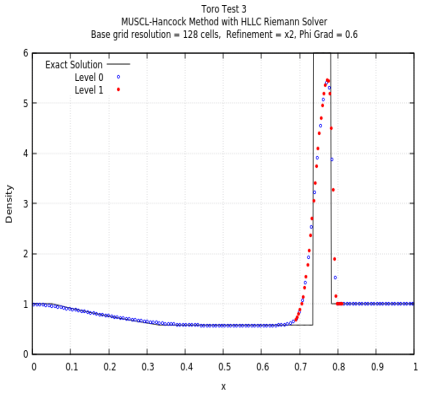

Figure A.44: Density with AMR x2

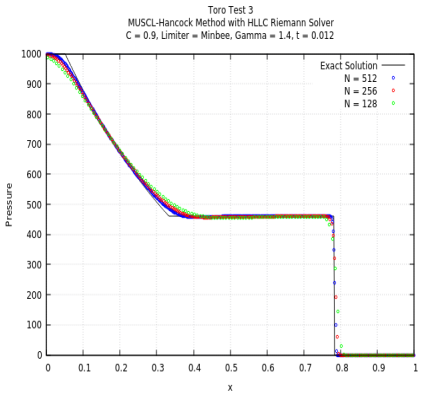

Figure A.42: Pressure

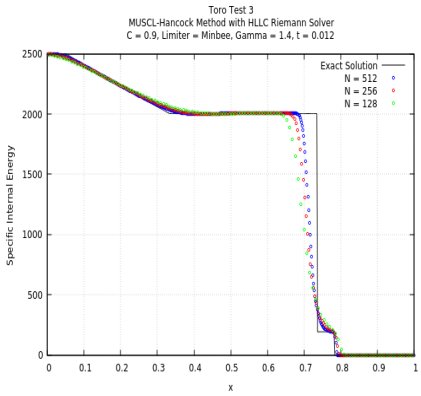

Figure A.43: Internal Energy

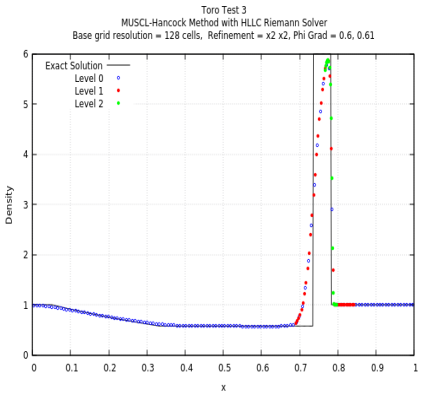

Figure A.45: Density with AMR x2 x2

Appendix B. One-dimensional test: Toro Test 4

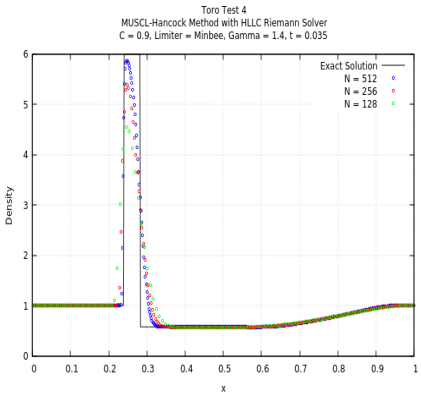

Figure B.46: Density

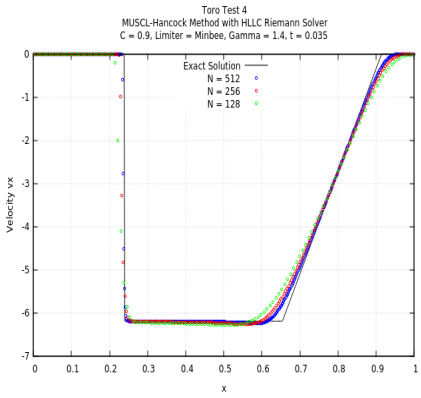

Figure B.47: Velocity

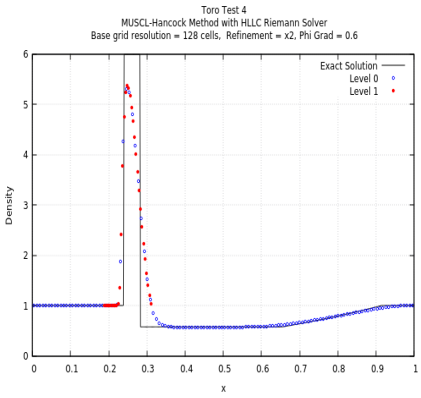

Figure B.50: Density with AMR x2

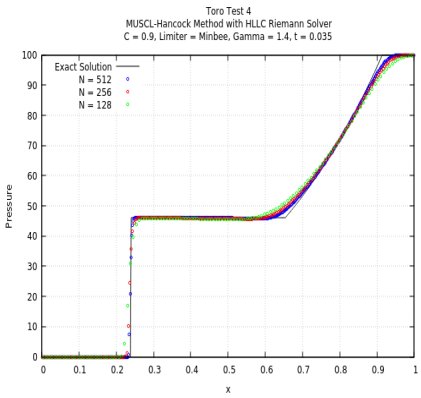

Figure B.48: Pressure

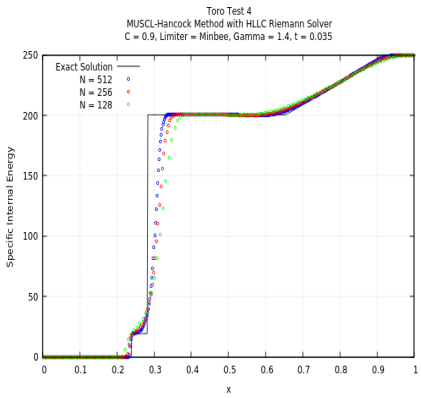

Figure B.49: Internal Energy

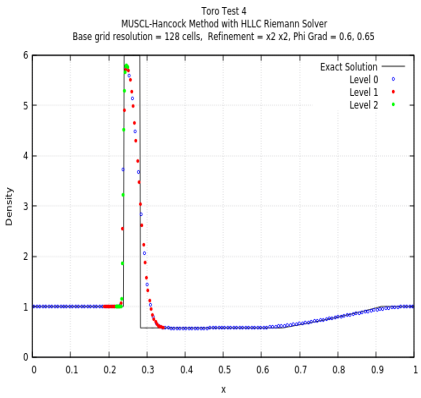

Figure B.51: Density with AMR x2 x2

Appendix C. One-dimensional test: Toro Test 5

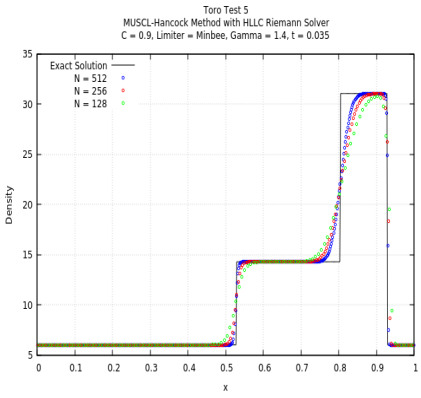

Figure C.52: Density

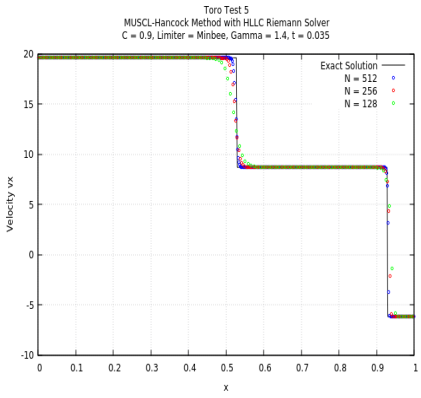

Figure C.53: Velocity

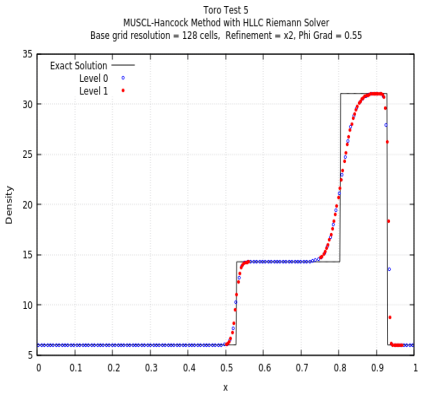

Figure C.56: Density with AMR x2

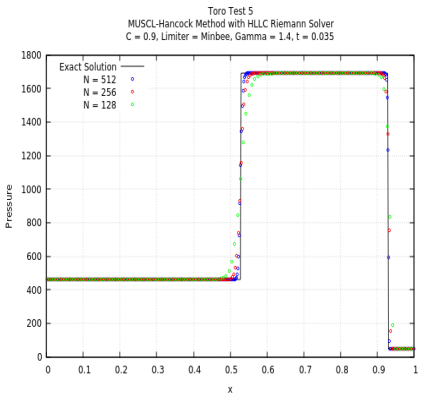

Figure C.54: Pressure

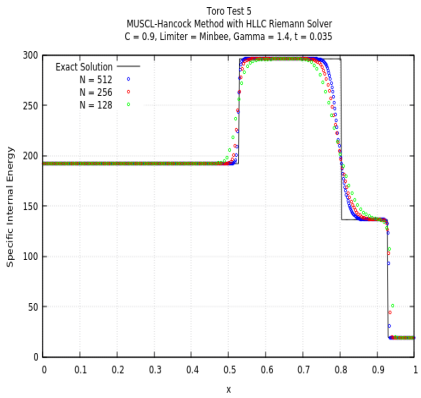

Figure C.55: Internal Energy

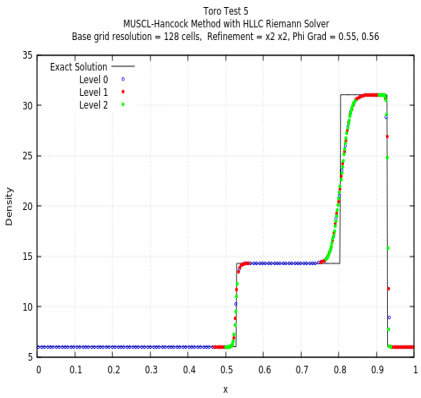

Figure C.57: Density with AMR x2 x2

Appendix D. Two-dimensional tests - TORO TEST 2

x-direction test

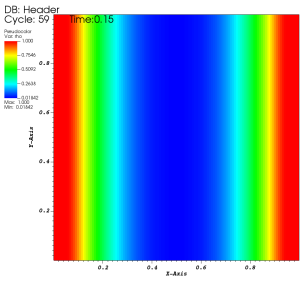

Figure D.58: Density

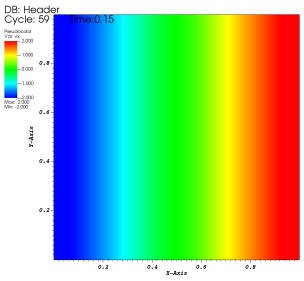

Figure D.59: Velocity v_x

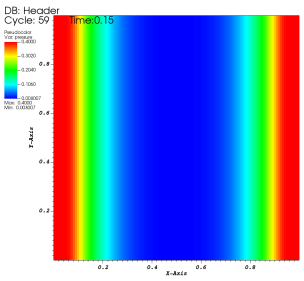

Figure D.60: Pressure

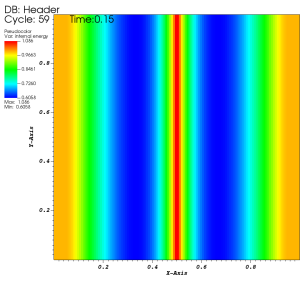

Figure D.61: Internal Energy

y-direction test

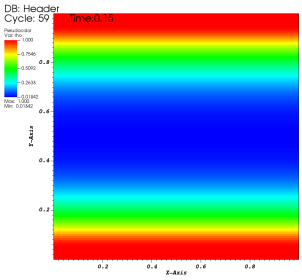

Figure D.62: Density

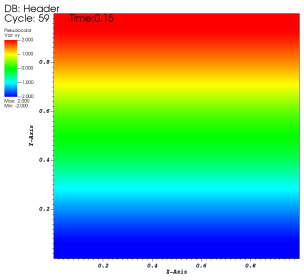

Figure D.63: Velocity v_y

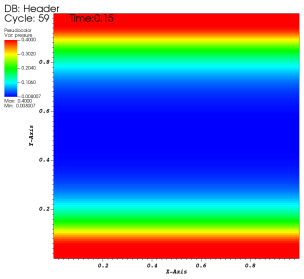

Figure D.64: Pressure

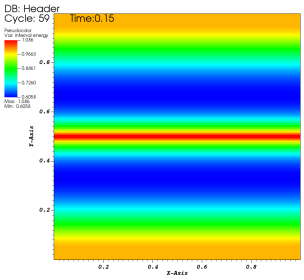

Figure D.65: Internal Energy

Diagonal test

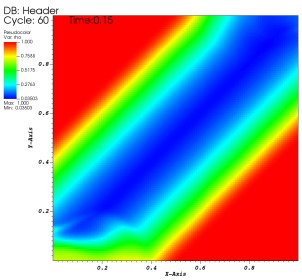

Figure D.66: Density

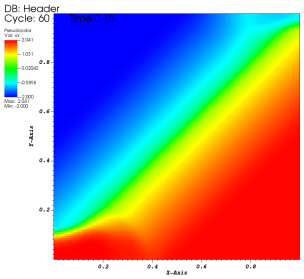

Figure D.67: Velocity v_x

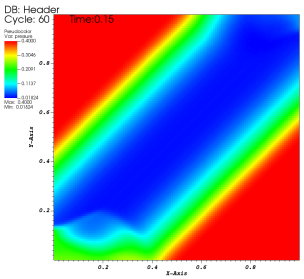

Figure D.68: Pressure

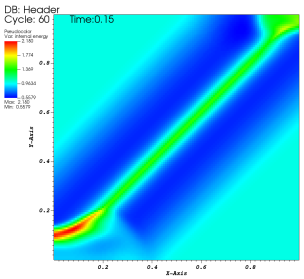

Figure D.69: Internal Energy

AMR test

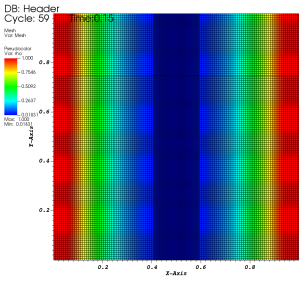

Figure D.70: Density, x-direction with $\times 2$ AMR

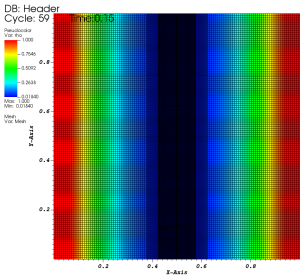

Figure D.71: Density, x-direction with $\times 2 \times 2$ AMR

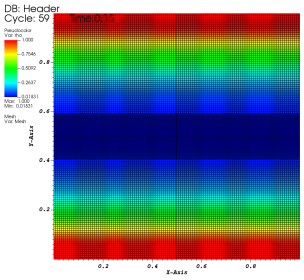

Figure D.72: Density, y-direction with $\times 2$ AMR

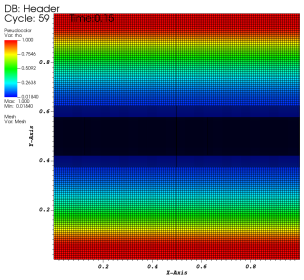

Figure D.73: Density, y-direction with $\times 2 \times 2$ AMR

Appendix E. Two-dimensional tests - TORO TEST 3

x-direction test

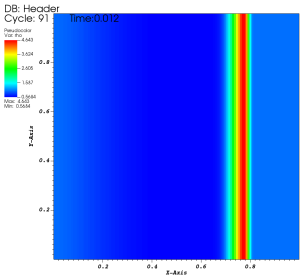

Figure E.74: Density

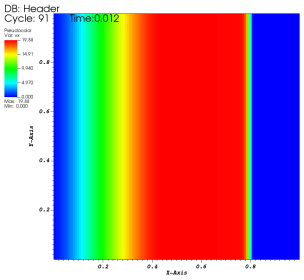

Figure E.75: Velocity v_x

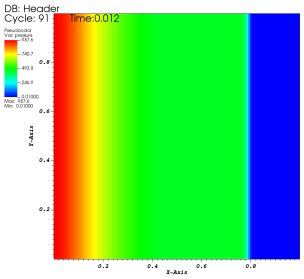

Figure E.76: Pressure

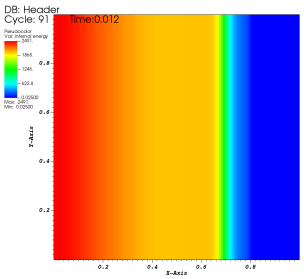

Figure E.77: Internal Energy

y-direction test

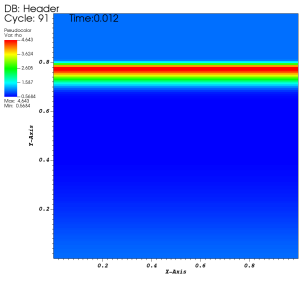

Figure E.78: Density

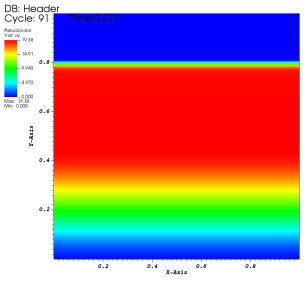

Figure E.79: Velocity v_y

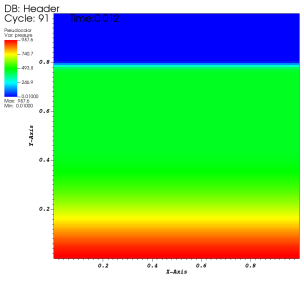

Figure E.80: Pressure

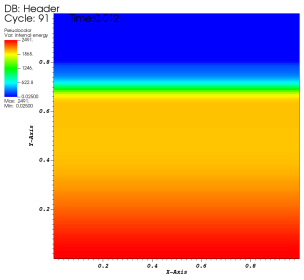

Figure E.81: Internal Energy

Diagonal test

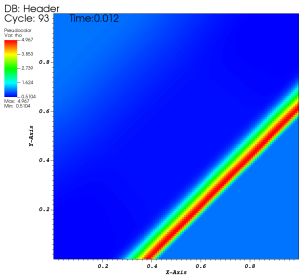

Figure E.82: Density

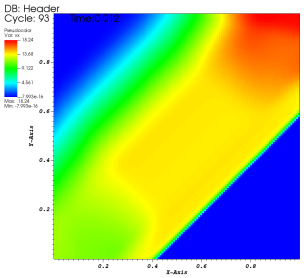

Figure E.83: Velocity v_x

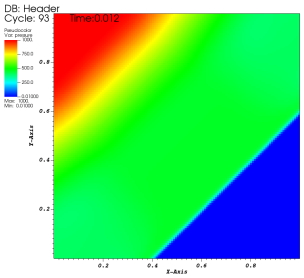

Figure E.84: Pressure

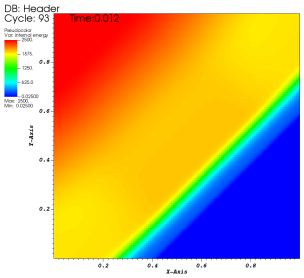

Figure E.85: Internal Energy

AMR test

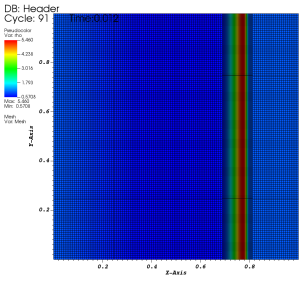

Figure E.86: Density, x-direction with $\times 2$ AMR

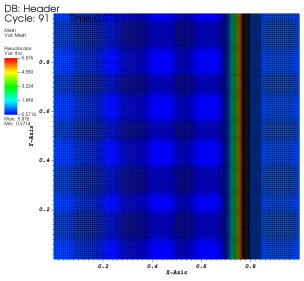

Figure E.87: Density, x-direction with $\times 2 \times 2$ AMR

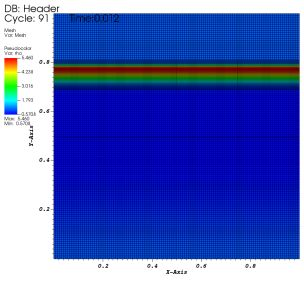

Figure E.88: Density, y-direction with $\times 2$ AMR

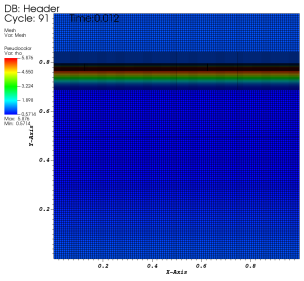

Figure E.89: Density, y-direction with $\times 2 \times 2$ AMR

Appendix F. Two-dimensional tests - TORO TEST 4

x-direction test

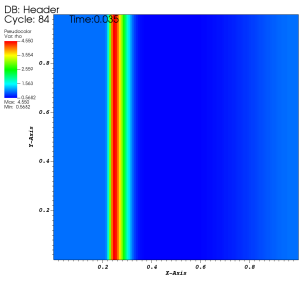

Figure F.90: Density

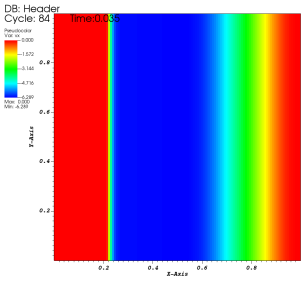

Figure F.91: Velocity v_x

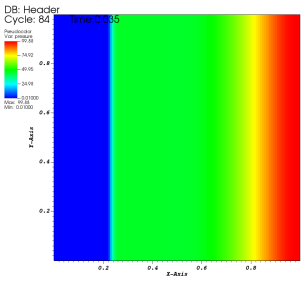

Figure F.92: Pressure

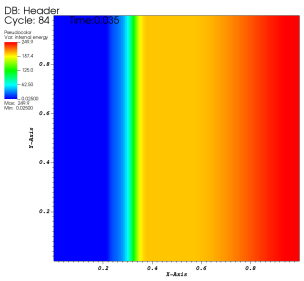

Figure F.93: Internal Energy

y-direction test

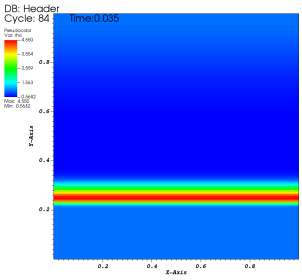

Figure F.94: Density

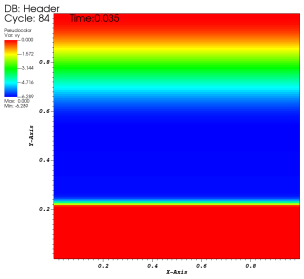

Figure F.95: Velocity v_y

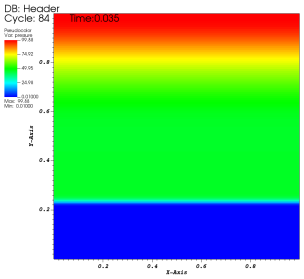

Figure F.96: Pressure

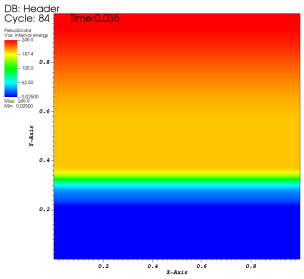

Figure F.97: Internal Energy

Diagonal test

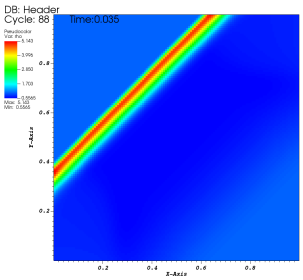

Figure F.98: Density

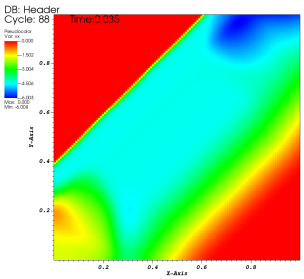

Figure F.99: Velocity v_x

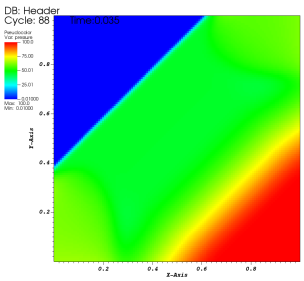

Figure F.100: Pressure

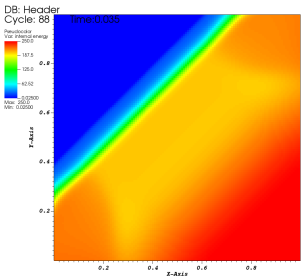

Figure F.101: Internal Energy

AMR test

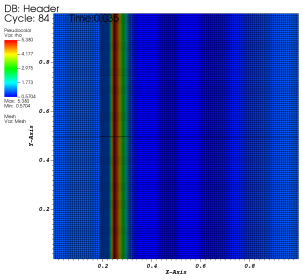

Figure F.102: Density, x-direction with $\times 2$ AMR

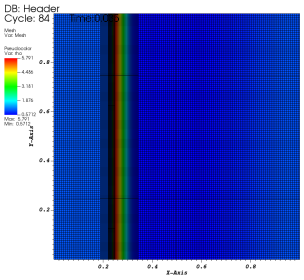

Figure F.103: Density, x-direction with $\times 2 \times 2$ AMR

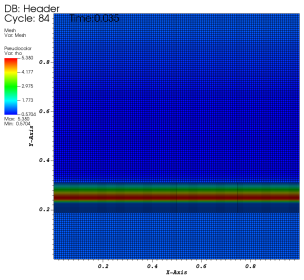

Figure F.104: Density, y-direction with $\times 2$ AMR

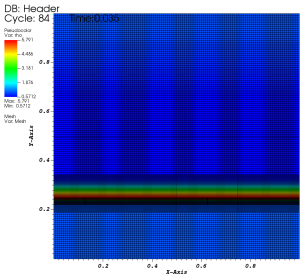

Figure F.105: Density, y-direction with $\times 2 \times 2$ AMR

Appendix G. Two-dimensional tests - TORO TEST 5

x-direction test

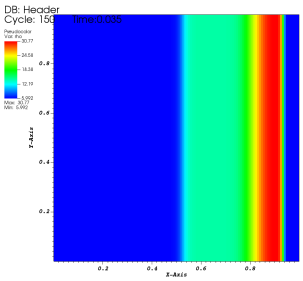

Figure G.106: Density

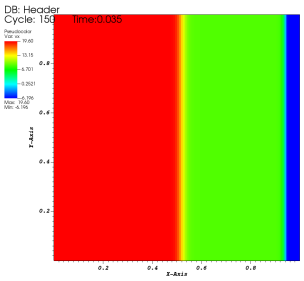

Figure G.107: Velocity v_x

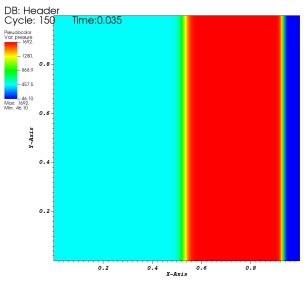

Figure G.108: Pressure

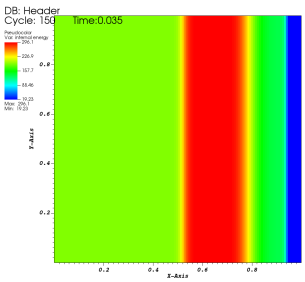

Figure G.109: Internal Energy

y-direction test

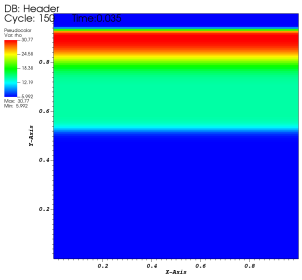

Figure G.110: Density

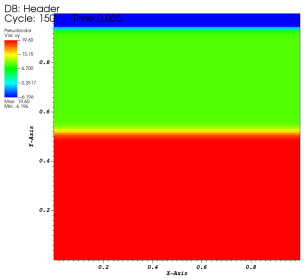

Figure G.111: Velocity v_y

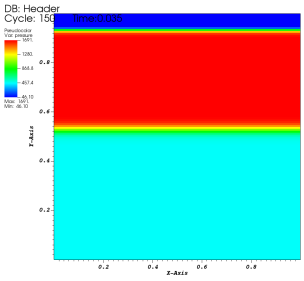

Figure G.112: Pressure

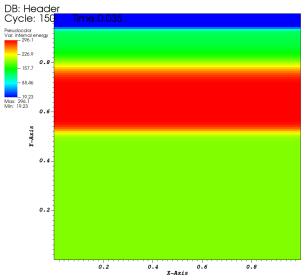

Figure G.113: Internal Energy

Diagonal test

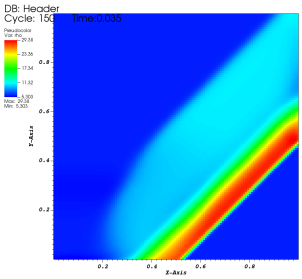

Figure G.114: Density

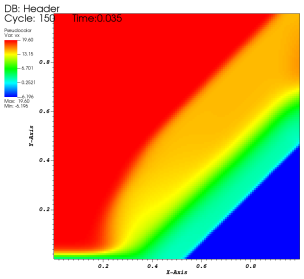

Figure G.115: Velocity v_x

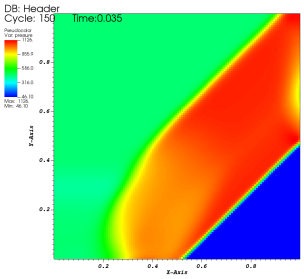

Figure G.116: Pressure

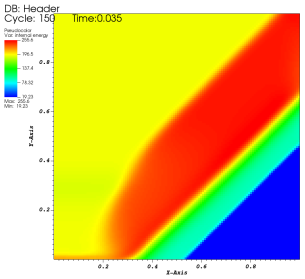

Figure G.117: Internal Energy

AMR test

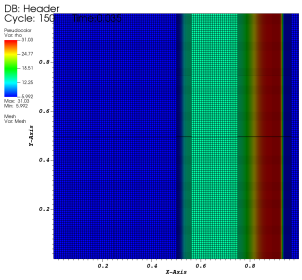

Figure G.118: Density, x-direction with $\times 2$ AMR

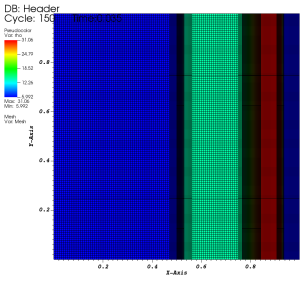

Figure G.119: Density, x-direction with $\times 2 \times 2$ AMR

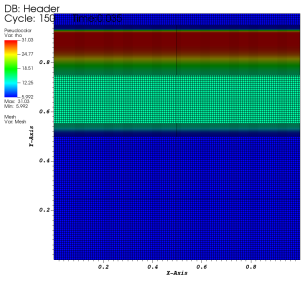

Figure G.120: Density, y-direction with $\times 2$ AMR

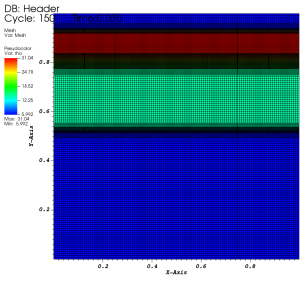

Figure G.121: Density, y-direction with $\times 2 \times 2$ AMR

References

[1] F. Fambri, M. Dumbser, O. Zanotti, Space–time adaptive ader-dg schemes for dissipative flows: Compressible navier–stokes and resistive mhd equations, Computer Physics Communications 220 (2017) 297–318. doi:https://doi.org/10.1016/j.cpc.2017.08.001.

[2] L. M. Stephen Millmore, N. Nikiforakis, Computational Continuum Modelling Lecture Notes, Laboratory for Scientific Computing, University of Cambridge (October 2020).

[3] W. Zhang, A. Almgren, V. Beckner, J. Bell, J. Blaschke, C. Chan, M. Day, B. Friesen, K. Gott, D. Graves, M. Katz, A. Myers, T. Nguyen, A. Nonaka, M. Rosso, S. Williams, M. Zingale, Amrex: a framework for block-structured adaptive mesh refinement, Journal of Open Source Software 4 (2019) 1370. doi:10.21105/joss.01370.

[4] P. D. L. Amiram Harten, B. van Leer, On upstream differencing and godunov-type schemes for hyperbolic conservation laws, SIAM 25 (1983) 35–61. doi:https://doi-org.ezp.lib.cam.ac.uk/10.1137/1025002.

[5] M. S. . W. S. E. F. Toro, Restoration of the contact surface in the hll-riemann solver, Shock Waves 4 (1994) 25–34.

[6] P. Blakely, Continuum modelling - amr lecture, Laboratory for Scientific Computing, University of Cambridge (January 2021).

[7] K. G. A. A. J. B. Weiqun Zhang, Andrew Myers, Amrex: Block-structured adaptive mesh refinement for multiphysics applications (09 2020).

[8] E. F. Toro, Riemann solvers and numerical methods for fluid dynamics: A practical introduction, 2nd Edition, Springer, 1999.

[9] S. F. Davis, Simplified second-order godunov-type methods, SIAM 9 (1987) 445–473. doi:https://doi.org/10.1137/0909030.

[10] B. Einfeldt, On godunov-type methods for gas dynamics, SIAM Journal on Numerical Analysis 25 (2) (1988) 294–318.

[11] H. Jasak, T. Uroic, Lecture notes: Computational continuum modelling ii (February 2020).

[12] J. O. Ferguson, C. Jablonowski, H. Johansen, P. McCorquodale, P. Colella, P. A. Ullrich, Analyzing the adaptive mesh refinement (amr) characteristics of a high-order 2d cubed-sphere shallow-water model, Monthly Weather Review 144 (2016) 4641–4666.

[13] M. J. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, Journal of Computational Physics 53 (3) (1984) 484–512. doi:https://doi.org/10.1016/0021-9991(84)90073-1.

[14] M. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, Journal of Computational Physics 82 (1) (1989) 64–84. doi:https://doi.org/10.1016/0021-9991(89)90035-1.

[15] J. J. Quirk, A parallel adaptive grid algorithm for computational shock hydrodynamics, Applied Numerical Mathematics 20 (4) (1996) 427–453, adaptive mesh refinement methods for CFD applications. doi:https://doi.org/10.1016/0168-9274(95)00105-0.

[16] M. E. Hubbard, N. Nikiforakis, A three-dimensional, adaptive, godunov-type model for global atmospheric flows, Monthly Weather Review 131 (8) (2003) 1848, copyright - Copyright American Meteorological Society Aug 2003; CODEN - MWREAB.

[17] J. Bell, M. Berger, J. Saltzman, M. Welcome, Three-dimensional adaptive mesh refinement for hyperbolic conservation laws, SIAM Journal on Scientific Computing 15 (1) (1994) 127–12, copyright - Copyright] © 1994 © Society for Industrial and Applied Mathematics; Last updated - 2012-02-16.

[18] F. J. Träuble, Multi-physics modelling of solid-plasma interaction, Master's thesis, University of Cambridge (2018).