LA-UR-12-24140

Approved for public release; distribution is unlimited.

Title: Final Report - Los Alamos National Laboratory Computational Physics
Summer Student Workshop

Author(s): Lung, Tyler B.
Roe, Phil
Morgan, Nathaniel R.

Intended for: Report

Disclaimer:

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Final Report - Los Alamos National Laboratory Computational Physics Summer Student Workshop

Student: Tyler Lung
PhD Pre-candidate
Department of Aerospace Engineering
University of Michigan
`tblung@umich.edu`

Graduate Advisor: Professor Phil Roe
Department of Aerospace Engineering
University of Michigan
`philroe@umich.edu`

Mentor: Dr. Nathaniel Morgan
XCP-8: Verification and Analysis
Los Alamos National Lab
`nmorgan@lanl.gov`

August 17, 2012

Abstract

The numerical solution of highly compressible, multi-material flows is an ongoing research area. These types of flows can be solved with a Lagrangian type mesh which moves with the material in a simulation to allow precise material interface tracking. Currently, researchers at Los Alamos National Laboratory and elsewhere are investigating cell-centered Lagrangian algorithms with the aim of producing methods that have second-order accuracy, preserve symmetry, and do not generate spurious vorticity. The new cell-centered algorithms solve a Riemann-like problem at the vertex of a cell. Professor Phil Roe at the University of Michigan has proposed a new structure for Lagrangian hydrodynamic algorithms that does not rely on the solution of the Riemann problem. The new approach utilizes Flux Corrected Transport (FCT) and it implements a form of vorticity control. The first step in the development of this method has been to construct an algorithm that solves the acoustic equations on an Eulerian mesh. The algorithm, which builds on the work of Morton and Roe [1], calculates fluxes at cell vertices, attains second-order accuracy using FCT, and has the special property of preserving vorticity. Results are presented that confirm the second order accuracy of the scheme and the vorticity preserving properties. The results are compared to the output produced by a MUSCL-Hancock algorithm. Some discussion of limiting methods for the FCT algorithm is also given.

1 Introduction

The development of improved algorithms for the simulation of compressible multi-material flows is of high importance to Los Alamos National Laboratory. The algorithms form the basis for production computer codes which are heavily relied upon to successfully carry out the lab's mission. The long term goal of the research presented here is to develop a new Lagrangian compressible flow algorithm that has three distinguishing characteristics:

1. Uses Flux Corrected Transport (FCT) to achieve second order accuracy while incorporating multi-dimensional physics.
2. Does not rely on the solution of the traditional one dimensional Riemann problem.
3. Implements a form of vorticity control.

This is an ambitious goal, and in the opinion of the author and his graduate advisor, it would be foolish to jump straight to an effort to implement a multi-dimensional Lagrangian algorithm. A better procedure is to start with simple implementations of the ideas stated above and make jumps in complexity as ideas and methods are proven and understood. The first step in this process was implementing one dimensional linear advection solvers of FCT and MUSCL-Hancock types on Eulerian meshes and comparing their performance. The results obtained showed that the FCT algorithm matched or exceeded the performance of the MUSCL-Hancock algorithm in all areas. Next, one dimensional solvers for the acoustic equations were implemented in both the FCT and MUSCL-Hancock flavors, again on Eulerian meshes. The results here were sufficiently encouraging to warrant the extension of the algorithms to two dimensions. This work was begun about one month prior to the start of the 2012 Computational Physics Summer Workshop at LANL and continued for its duration. As such, the 2D vorticity preserving FCT algorithm for the acoustic equations is the focus of this report.

It is worth discussing the impetus for developing an Eulerian solver for the acoustic equations. After all, it has already been stated that the ultimate goal of this research is to produce a Lagrangian algorithm. Such an algorithm must incorporate mesh movement and non-linear physics, two things which are not present in the acoustic solver. The key here is to realize that the acoustic equations are a system of partial differential equations that exhibit symmetric wavespeeds. This is a characteristic they share with the Lagrangian equations of gas dynamics. The plan is to exploit this trait when developing a flux limiting procedure for the Lagrangian algorithm. The acoustic equations provide a simplified development environment for such a limiting procedure.

2 Vorticity Preserving Flux Corrected Transport Algorithm

The numerical method presented in the following section builds on the work of Morton and Roe [2], which was published in 2001. In that paper, the authors present a Rotated Richtmeyer scheme which is formulated using corner fluxes. The Rotated Richtmeyer scheme is a two dimensional variant of the Lax-Wendroff method. By computing and storing fluxes at cell vertices, Morton and Roe were able to produce a scheme that, by construction, guarantees that the discrete vorticity

$$\zeta = \delta_x \mu_y v - \delta_y \mu_x u \quad (1)$$

is preserved.¹ This fits the physics of the acoustic equations perfectly provided that the background state of the fluid of interest has a uniform density. Note that if the method were extended to

¹See Appendix A for a definition of the compact differencing operators used in equations throughout this document.

solve a system of governing equations that allow vorticity to evolve in time, a vorticity correction mechanism would need to be developed. There are benefits beyond the vorticity preserving property when working in terms of nodal fluxes. This eliminates the need to solve traditional one dimensional Riemann problems at cell interfaces as is customary in Godunov based finite volume methods. There are two main reasons why we seek to distance ourselves from using a Riemann solver to compute fluxes. First, strictly speaking, the Riemann problem is one dimensional in nature. In this research, emphasis is placed on making the new algorithm as multi-dimensional as possible. Second, when considering Lagrangian applications, the use of Riemann solutions at cell interfaces forces one to relate face velocities around a node to a unique nodal velocity. This can be a difficult task and it is one that we wish to avoid. By storing fluxes at vertices, the unique velocity of each node is already defined.

The ideas of Flux Corrected Transport, originally developed by Boris and Book [2] were chosen to achieve second order accuracy. This was due to the multi-dimensional and adaptive limiting framework that it provides. An FCT algorithm computes a first order solution which is known to be free of spurious oscillations and then uses that information to relax limiting constraints where it is appropriate. This stands in contrast to other limiting schemes which apply limiting whenever a new maximum or minimum could develop without giving a thought to whether the extremum should exist or not. The first order scheme used here is equivalent to the First Order Upwind Method, but it does not require explicit upwinding. The high order scheme used to define antidiffusive fluxes is the Rotated Richtmeyer scheme, mentioned previously.

2.1 Problem Statement

In two dimensions the acoustic equations read

$$p_t + \rho_0 a_0^2 (u_x + v_y) = 0 \quad (2)$$

$$u_t + \frac{1}{\rho_0} p_x = 0 \quad (3)$$

$$v_t + \frac{1}{\rho_0} p_y = 0 \quad (4)$$

Note that u, v , and p represent perturbations to a fluid with a uniform background state described by $u_0 = 0, v_0 = 0$ and $p_0 = \text{constant}$. Zero subscripts always denote a property of the background state. For example, ρ_0 and a_0 give the density and sound speed of the background state, respectively.

Assume a structured quadrilateral mesh with $\Delta x = \Delta y = h$. Define the timestep as k . The conserved variables $\mathbf{u} = (p, u, v)^T$ are stored at cell centers and the fluxes $\mathbf{f} = (\rho_0 a_0^2 u, 0, \rho_0^{-1} p)^T$ and $\mathbf{g} = (0, \rho_0 a_0^2 v, \rho_0^{-1} p)^T$ are stored at vertices.

2.2 Numerical Recipe

A step by step numerical recipe for the Vorticity Preserving Flux Corrected Transport Algorithm (VPFCT) will now be given.

2.2.1 Compute Low Order Solution

1. Calculate the fluxes at each vertex from adjacent cell centered values.

$$\hat{\mathbf{f}} = \mu_x \mu_y \mathbf{f} \quad (5)$$

$$\hat{\mathbf{g}} = \mu_x \mu_y \mathbf{g} \quad (6)$$

2. Evolve each vertex flux through a partial time step.

$$\hat{\mathbf{F}} = \hat{\mathbf{f}} - \frac{\nu h}{2k} \begin{pmatrix} \delta_x \mu_y u + \delta_y \mu_x v \\ 0 \\ \delta_x \mu_y p \end{pmatrix} \quad (7)$$

$$\hat{\mathbf{G}} = \hat{\mathbf{g}} - \frac{\nu h}{2k} \begin{pmatrix} 0 \\ \delta_x \mu_y u + \delta_y \mu_x v \\ \delta_y \mu_x p \end{pmatrix} \quad (8)$$

$$\nu \equiv \frac{a_0 k}{h} \quad (9)$$

3. Loop over cells to update.

$$\mathbf{u}^* = \mathbf{u}^n - \frac{k}{h} (\delta_x \mu_y \hat{\mathbf{F}} + \delta_y \mu_x \hat{\mathbf{G}}) \quad (10)$$

2.2.2 Compute High Order Solution

1. Define antidiffusive fluxes such that $\widehat{\mathbf{F}}^{\text{AD}*} = \widehat{\mathbf{F}}^{\text{H}*} - \widehat{\mathbf{F}}^{\text{L}*}$ and $\widehat{\mathbf{G}}^{\text{AD}*} = \widehat{\mathbf{G}}^{\text{H}*} - \widehat{\mathbf{G}}^{\text{L}*}$.

$$\widehat{\mathbf{F}}^{\text{AD}*} = \frac{a_0(\nu^2 - \nu)}{2\nu} \begin{pmatrix} \delta_x \mu_y u^* + \delta_y \mu_x v^* \\ 0 \\ \delta_x \mu_y p^* \end{pmatrix} \quad (11)$$

$$\widehat{\mathbf{G}}^{\text{AD}*} = \frac{a_0(\nu^2 - \nu)}{2\nu} \begin{pmatrix} 0 \\ \delta_x \mu_y u^* + \delta_y \mu_x v^* \\ \delta_y \mu_x p^* \end{pmatrix} \quad (12)$$

2. Compute limiting coefficients, ω , for the antidiffusive fluxes.

See Section 3 for more details.

3. Update the solution to the $n + 1$ time level.

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{k}{h} (\delta_x \mu_y \omega_{\mathbf{f}} \widehat{\mathbf{F}}^{\text{AD}*} + \delta_y \mu_x \omega_{\mathbf{g}} \widehat{\mathbf{G}}^{\text{AD}*}) \quad (13)$$

3 Flux Limiting

3.1 Procedure for One Dimension

The simplest flux limiting case, that which occurs in one dimension, will be considered first. The goal is to implement a method that will ensure that monotone data will remain monotone while advancing in time. In one dimension, the definition of monotonicity is simple and well defined. When computing the limiting coefficients for a given cell, the first order solution will be used to determine if the cell value falls on the interval defined by its left and right neighbors. If it does, the data is monotone and we wish to keep it that way. Mathematically, this can be expressed with the inequality

$$0 \leq \frac{u_j^{n+1} - u_{j-1}^*}{u_{j+1}^* - u_{j-1}^*} \leq 1 \quad (14)$$

By defining the quantities $\delta u_L^* = u_j^* - u_{j-1}^*$ and $\delta u_R^* = u_{j+1}^* - u_j^*$ and substituting them, along with the update equation, into the previous inequality we obtain

$$0 \leq \frac{2}{Q^+} + \omega_L - \omega_R \frac{\delta u_L^*}{\delta u_R^*} \leq \frac{2}{Q^+} \left(1 + \frac{\delta u_R^*}{\delta u_L^*}\right) \quad (15)$$

Where

$$Q^+ = Q - \nu^2 \quad Q = \nu \quad \nu \equiv \frac{a_0 k}{h} \quad (16)$$

Note that the limiting coefficients ω_L and ω_R have been applied to the fluxes in the update equation prior to substitution. The coefficients must fall between zero and one. A value of zero reduces the solution to first order, while a value of one gives the full second order method. The goal is to find the largest allowable coefficients that will still prevent non-physical oscillations from developing in the solution.

Noting that all the quantities in the inequalities are positive, one can use "worst case scenarios" to arrive at expressions for the limiting coefficients. The left inequality is

$$0 \leq \frac{2}{Q^+} + \omega_L - \omega_R \frac{\delta u_L^*}{\delta u_R^*} \quad (17)$$

The positive term ω_L can only help this inequality, so the worst case of $\omega_L = 0$ is assumed to give

$$\omega_R \leq \frac{2}{Q^+} \frac{\delta_L^*}{\delta_R^*} \quad (18)$$

Similar logic is used on the right inequality to give

$$\omega_L \leq \frac{2}{Q^+} \left(1 + \frac{\delta u_R^*}{\delta u_L^*}\right) \quad (19)$$

3.1.1 Procedure for Two Dimensions

The challenge now, which is of critical importance, is to develop a limiting method which can be applied to two dimensional, and eventually three dimensional, problems. The method and reasoning presented in the previous section becomes much more difficult in two dimensions since each flux depends on four neighboring cells. This adds a great deal of complexity to the inequalities and makes it difficult or impossible to find explicit expressions for the limiting coefficients. To begin analyzing the two dimensional case we will first consider the second order pressure correction

$$p^{n+1} = p^* - \frac{k}{h} (\delta_x \mu_y \omega_f \widehat{F^{AD*}} + \delta_y \mu_x \omega_g \widehat{G^{AD*}}) \quad (20)$$

Where the fluxes are given by

$$\widehat{F^{AD*}} = \frac{a_0(\nu^2 - \nu)}{2\nu} (\delta_x \mu_y p^*) \quad (21)$$

$$\widehat{G^{AD*}} = \frac{a_0(\nu^2 - \nu)}{2\nu} (\delta_y \mu_x p^*) \quad (22)$$

Performing a fair amount of algebra, the second order pressure update simplifies to

$$p^{n+1} = p^* - \frac{(\nu^2 - \nu)}{4} [\omega_1(p_1 - p_5) + \omega_2(p_3 - p_5) + \omega_3(p_9 - p_5) + \omega_4(p_7 - p_5)] \quad (23)$$

This is a notable result and shows that the fluxes for the second order pressure update depend only on the diagonal neighbors of the cell of interest. Furthermore, this implies that there are

actually two distinct pressure fluxes per node. Since these two fluxes depend only on the diagonal neighbors, they can be limited independently while still preserving the vorticity preserving properties of the scheme. Using this interpretation of the fluxes, it can be observed that we nearly have a one dimensional limiting problem along each cell diagonal. In fact, if the cell centered pressure could be intelligently split between the two sets of diagonal “flux pairs”, two one dimensional limiting problems would result. Since we know how to solve one dimensional limiting problems, it is this procedure we seed to implement. Note that this technique could be easily generalized to three dimensions.

In order to distribute the cell centered pressure between the two one dimensional limiting problems, define two weights w_a and w_b . The simplest way to approach this would be to assign values of 0.5 to each weight, which would evenly split the cell centered value between the limiting problems. However, it seems that this would be overly simplistic and that it would be better to assign more of the cell centered value to the flux pair that has the largest net flux value. The logic here is that the flux pair with the largest net value will have the greatest effect during the final cell update and therefore should be assigned a larger portion of the cell centered value for limiting purposes. Now define

$$f_a^{net} \equiv \text{abs}(f_1 + f_3) \quad f_b^{net} \equiv \text{abs}(f_2 + f_4) \quad (24)$$

From here the distribution weights can be defined as

$$w_a = \frac{f_a^{net}}{f_a^{net} + f_b^{net}} \quad w_b = \frac{f_b^{net}}{f_a^{net} + f_b^{net}} \quad (25)$$

and the pressures assigned to each limiting problem are given by

$$p_a = w_a p_5 \quad p_b = w_b p_5 \quad (26)$$

To summarize, the two dimensional pressure limiting problem has now been reduced to two one dimensional limiting problems which are solved along cell diagonals. This technique results in two limiting coefficients being computed per node, one for each diagonal flux. Implementation could be such that the two fluxes at each node are limited independently or the most conservative coefficient could be applied to both fluxes. Next it is necessary to turn our attention to the velocities. The natural question to ask here is if the velocity components can be limited in the same manner as the pressure. To investigate this question, we turn to the continuous form of the equations and analyze the second order terms.

The acoustic system is repeated here for convenience in vector form

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{1}{\rho_0} \nabla(p) = 0 \quad (27)$$

$$\frac{\partial p}{\partial t} + \rho_0 a_0^2 \nabla \bullet \mathbf{u} = 0 \quad (28)$$

Differentiating (1) with respect to time we have

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} + \frac{1}{\rho_0} \frac{\partial}{\partial t} \nabla(p) = 0 \quad (29)$$

Take the gradient of (2) to obtain

$$\nabla\left(\frac{\partial p}{\partial t}\right) + \rho_0 a_0^2 \nabla(\nabla \bullet \mathbf{u}) = 0 \quad (30)$$

Noting that partial differentiation is commutative we have

$$\frac{\partial}{\partial t} \nabla(p) = -\rho_0 a_0^2 \nabla(\nabla \bullet \mathbf{u}) \quad (31)$$

Substitute (5) into (3) to show

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} = a_0^2 \nabla(\nabla \bullet \mathbf{u}) \quad (32)$$

Recall the vector identity

$$\nabla(\nabla \bullet \phi) = \nabla \bullet (\nabla \phi) + \nabla \times (\nabla \times \phi) \quad (33)$$

Applying (7) to the RHS of (6) we have

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} = a_0^2 \nabla \bullet (\nabla \mathbf{u}) + \nabla \times (\nabla \times \mathbf{u}) \quad (34)$$

Noting that $\nabla \times \mathbf{u} = \boldsymbol{\omega}$ and $\nabla \bullet (\nabla) \equiv \nabla^2$ we write

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} = a_0^2 \nabla^2 \mathbf{u} + \nabla \times \boldsymbol{\omega} \quad (35)$$

Some vector manipulation will show that $\nabla \times \boldsymbol{\omega}$ is given in 2D by

$$\frac{\partial}{\partial y} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) \hat{\mathbf{i}} - \frac{\partial}{\partial x} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) \hat{\mathbf{j}} \quad (36)$$

The x and y components of the second order velocity equation are therefore

$$u_{tt} = a_0^2 (u_{xx} + u_{yy} + v_{yx} - u_{yy}) \quad (37)$$

$$v_{tt} = a_0^2 (v_{xx} + v_{yy} - v_{xx} + u_{xy}) \quad (38)$$

which can also be written as

$$\frac{\partial^2 u}{\partial t^2} = a_0^2 (\nabla^2(u) + \frac{\partial}{\partial y} \omega_i) \quad (39)$$

$$\frac{\partial^2 v}{\partial t^2} = a_0^2 (\nabla^2(v) - \frac{\partial}{\partial x} \omega_j) \quad (40)$$

Note at this point that the second order velocity corrections are composed of two terms. The first is Laplacian and the second term contains a component of the vorticity curl. The algebra will be omitted for brevity, but if one follows an analagous procedure for the pressure, you will find that

$$p_{tt} = a_0^2 (p_{xx} + p_{yy}) \quad (41)$$

It is now understood that the tidy form of the equation for the second order pressure update used in constructing the limiting scheme was due to the the Laplacian form of the continous equation. While the velocities have a Laplacian term also, there is an additional vorticity term that must also be accounted for. Perhaps it should be pointed out that this term is, obviously, only important for vortical flows. In the absence of vorticity the velocity terms could be limited in a way that is completely analagous to the pressure. In fact, the current implementation of the code does just

this, and vorticity is neglected when limiting the velocity. Improving this procedure is one focus of ongoing work.

There is one last issue related to velocity limiting that deserves some attention. The velocity limiting must be done in such a way that does not destroy the vorticity preserving property of the scheme. A natural question that arises here is whether or not different limiting coefficients can be applied to the velocity components at a given node while still keeping this important property intact.

Recall the compact vorticity, which we seek to preserve

$$\zeta = \delta_x \mu_y v - \delta_y \mu_x u \quad (42)$$

Vorticity preservation implies that the following condition should be enforced for any two successive time levels

$$\zeta^{n+1} = \zeta^n \implies \delta_x \mu_y v^{n+1} - \delta_y \mu_x u^{n+1} = \delta_x \mu_y v^n - \delta_y \mu_x u^n \quad (43)$$

This is accomplished, per the work of Morton and Roe [1], by evaluating the divergence of pressure via vertex fluxes. To understand how to keep this property while limiting the velocity components, first take a look at the first order scheme. Recall that the update equations for the first order scheme look like

$$u^* = u^n - \frac{k}{h} [\delta_x \mu_y (\mu_x \mu_y \rho a^2 u - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))] \quad (44)$$

$$v^* = v^n - \frac{k}{h} [\delta_y \mu_x (\mu_x \mu_y \rho a^2 v - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))] \quad (45)$$

Substitute (44) and (45) into (43) and cancel terms

$$\begin{aligned} & \delta_x \mu_y [v^n - \frac{k}{h} [\delta_y \mu_x (\mu_x \mu_y \rho a^2 v - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))]] \\ & - \delta_y \mu_x [u^n - \frac{k}{h} [\delta_x \mu_y (\mu_x \mu_y \rho a^2 u - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))]] \\ & = \delta_x \mu_y v^n - \delta_y \mu_x u^n \end{aligned} \quad (46)$$

$$\begin{aligned} & \delta_x \mu_y [-\frac{k}{h} [\delta_y \mu_x (\mu_x \mu_y \rho a^2 v - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))]] \\ & - \delta_y \mu_x [-\frac{k}{h} [\delta_x \mu_y (\mu_x \mu_y \rho a^2 u - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))]] \\ & = 0 \end{aligned} \quad (47)$$

$$-\delta_x \mu_y \frac{k}{h} [\delta_y \mu_x \mu_x \mu_y \rho a^2 v] + \delta_y \mu_x \frac{k}{h} [\delta_x \mu_y \mu_x \mu_y \rho a^2 u] = 0 \quad (48)$$

If vorticity is to be preserved, the LHS of (48) must evaluate to zero. Experiments with the existing code have shown that this is indeed the case. Now consider the second order FCT scheme. The update equations here are

$$u^{n+1} = u^* - \frac{k}{h} [\delta_x \mu_y \omega_u \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)] \quad (49)$$

$$v^{n+1} = v^* - \frac{k}{h} [\delta_y \mu_x \omega_v \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)] \quad (50)$$

Substitute (49) and (50) into (43)

$$\begin{aligned} & \delta_x \mu_y [v^* - \frac{k}{h} [\delta_y \mu_x \omega_v \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)]] \\ & - \delta_y \mu_x [u^* - \frac{k}{h} [\delta_x \mu_y \omega_u \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)]] \\ & = \delta_x \mu_y v^n - \delta_y \mu_x u^n \end{aligned} \quad (51)$$

Substitute (44) and (45) into (51) and then cancel terms

$$\begin{aligned} & \delta_x \mu_y [v^n - \frac{k}{h} [\delta_y \mu_x (\mu_x \mu_y \rho a^2 v - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))] \\ & \quad - \frac{k}{h} [\delta_y \mu_x \omega_v \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)]] \\ & - \delta_y \mu_x [u^n - \frac{k}{h} [\delta_x \mu_y (\mu_x \mu_y \rho a^2 u - \frac{Qh}{2k} (\delta_x \mu_y u^n + \delta_y \mu_x v^n))] \\ & \quad - \frac{k}{h} [\delta_x \mu_y \omega_u \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)]] \\ & = \delta_x \mu_y v^n - \delta_y \mu_x u^n \end{aligned} \quad (52)$$

$$\begin{aligned} & \delta_x \mu_y [-\frac{k}{h} [\delta_y \mu_x (\mu_x \mu_y \rho a^2 v)] - \frac{k}{h} [\delta_y \mu_x \omega_v \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)]] \\ & - \delta_y \mu_x [-\frac{k}{h} [\delta_x \mu_y (\mu_x \mu_y \rho a^2 u)] - \frac{k}{h} [\delta_x \mu_y \omega_u \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)]] \\ & = 0 \end{aligned} \quad (53)$$

From (48) we can infer that terms 1 and 3 of (53) combine to give zero, leaving

$$\begin{aligned} & -\delta_x \mu_y \frac{k}{h} [\delta_y \mu_x \omega_v \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)] \\ & + \delta_y \mu_x \frac{k}{h} [\delta_x \mu_y \omega_u \frac{aQ^+}{2\nu} (\delta_x \mu_y u^* + \delta_y \mu_x v^*)] \\ & = 0 \end{aligned} \quad (54)$$

Factoring like terms and operators gives

$$\delta_x \mu_y \delta_y \mu_x \frac{Q^+}{2} [\omega_v (\delta_x \mu_y u^* + \delta_y \mu_x v^*) - \omega_u (\delta_x \mu_y u^* + \delta_y \mu_x v^*)] = 0 \quad (55)$$

Inspection of (55) implies that the limiting coefficients must be equal for the equality to be true. This analysis suggests that vorticity will be generated if dissimilar limiting coefficients are applied to the velocity components. This conclusion agrees with experiments run to date with the research code.

4 Numerical Results

Results obtained from the most recent implementation of the code are now presented. First, convergence analysis will be discussed to gain confidence that the Vorticity Preserving FCT (VPFCT) and MUSCL-H algorithms are working properly. Then results will be presented that compare the performance of the VPFCT and MUSCL-H scheme for the following classes of problems: smooth initial data, discontinuous initial data, and vortical flows.

4.1 Convergence Analysis

Some convergence analysis was done using Richardson's deferred approach to the limit. This notes that the following equation can be used to relate an exact solution, its numerical approximation, and the design convergence rate of the numerical method. For both VPFCT and MUSCL-H we have

$$\phi^{exact} = \phi^{approx.} + O(h^2) \quad (56)$$

Recognizing that the exact solution is a constant we can rearrange this expression to get

$$\phi^{approx} = c_1 h^2 + c_2 \quad (57)$$

When plotting some code output as a function of the mesh parameter, h , a linear relationship should exist. This procedure was performed by taking the code output of interest to be the average velocity magnitude squared over the entire domain. Velocity magnitude squared was chosen so that a zero average would not be obtained on problems with cylindrical symmetry. The velocity magnitude was left squared to avoid a square root sign that complicates the analysis. It should be noted since the velocity magnitude was squared, there is a fourth order deviation from the perfect linear relationship shown above. This was assumed to be negligible. The test problem had smooth initial data: a Gaussian pressure disturbance to a fluid at rest.

A number of interesting conclusions can be reached from studying the convergence results presented in Figure 1. On an encouraging note, this figure shows that the unlimited versions of both VPFCT and MUSCL-H are converging at second order and to the same value. In addition, the constant in the leading error term of the VPFCT scheme is smaller than that of the MUSCL-H, as evidenced by the fact that the VPFCT line falls above that for MUSCL-H on the plot. The results for the limited VPFCT scheme were disappointing as the convergence failed on finer meshes. Understanding this failure and fixing it is an area of ongoing work. The VPFCT limiter which produced these results used the distribution weights defined using net flux values. In addition, it used the most conservative pressure limiting coefficient at each node. Experimentation has shown that this approach is preferred over using two limiting coefficients at each node. The implementation using distinct limiting coefficients has shown to increase the leading error constant during experimentation. Ironically, the convergence behavior of the VPFCT scheme can be improved by setting the distribution weights to 0.5. At the current time, it appears that using the more complicated weights introduces a large amount of noise into the velocity divergence and throws off the convergence of the scheme. Convergence results when the VPFCT limiting scheme uses 0.5 for the distribution weights are shown in Figure 2.

4.2 Smooth Initial Data

Tests were run using a problem with smooth initial data in which a Gaussian Pressure perturbation was introduced to a fluid at rest. The pressure field was initialized according to

Figure 1: Convergence Study - Convergence Problem with Limited VPFCT

$$p(x, y, 0) = 2\exp(x^2 + y^2) \quad (58)$$

Results were plotted as function of radius to help visualize the symmetry preservation ability of the algorithms. Figure 3 shows the pressure and velocity magnitude profiles for the the VPFCT and MUSCL-H schemes. Note that the VPFCT scheme has a slightly higher peak pressure and velocity. Perhaps of more interest was verifying the vorticity preserving properties of the VPFCT scheme. Figure 4 compares the compact vorticity contours produced by each scheme after three time units. The VPFCT maintains zero vorticity to double precision, while the MUSCL-H scheme creates vorticity in an interesting pattern.

4.3 Discontinuous Initial Data

A problem with discontinuous initial data was run in order to better exercise the limiting mechanisms of each scheme. Here a cylindrical pressure perturbation was introduced to a fluid at rest. The pressure field was initialized according to

$$p(x, y, 0) = 2 \quad r \leq 2 \quad p(x, y, 0) = 0 \quad r \geq 2 \quad (59)$$

Figure 2: Convergence Study - VPFCT Distribution Weights set to 0.5

Inspection of the pressure and velocity profiles in Figure 5 shows that while the VPFCT result is reasonable, more work is still needed on the limiting procedure. Some overshoots still develop, especially at the top of the shock front, in both the pressure and velocity magnitude plots. The vorticity plot shown in Figure 6 are even more encouraging than for the smooth problem. The vorticity preserving property of the VPFCT is again verified, while the MUSCL-H scheme manufactures significant vorticity.

4.4 Vortical Flows

To this point, all the results that have been introduced were for problems with zero vorticity. For completeness, a test problem with finite vorticity is also included. Here the Taylor-Green vortex solution is modified for inviscid flow. The pressure and velocity fields were intialized according to

$$p(x, y, 0) = \frac{2\rho_0}{4}[\cos(2x) + \cos(2y)] \quad (60)$$

$$u(x, y, 0) = 2\sin(x)\cos(y) \quad (61)$$

$$v(x, y, 0) = -2\cos(x)\sin(y) \quad (62)$$

Figure 3: Pressure and Velocity Magnitude Profiles for Gaussian Pressure Disturbance, $t = 3$ units

Figure 4: Compact Vorticity Contours for Gaussian Pressure Disturbance, $t = 3$ units

The vorticity contours at $t = 0$ are shown in Figure 7. The vorticity contours after 3 time units (or around 2000 timesteps) for the VPFCT and MUSCL-H schemes are shown in Figure 8. Note that the VPFCT scheme is able to maintain the vortices, while the MUSCL-H scheme distorts them. It is worth pointing out that the Taylor-Green solution assumes incompressible flow. Since the discrete operators in this scheme do not exactly evaluate the velocity divergence to zero and these algorithms are for compressible flow, small pressure waves propagate through the domain during the simulation. Even with the wave interactions, the VPFCT scheme is able to preserve vorticity.

5 Conclusions and Future Work

Significant progress has been made on the short term research goals discussed in the introduction. A two dimensional VPFCT algorithm has been implemented and compared to the performance of a MUSCL-H algorithm. The vorticity preserving properties of the VPFCT algorithm have been verified and its performance has shown promise relative to the MUSCL-H algorithm. Specifically,

Figure 5: Pressure and Velocity Magnitude Profiles for Cylindrical Pressure Disturbance, $t = 3$ units

Figure 6: Compact Vorticity Contours for Cylindrical Pressure Disturbance, $t = 3$ units

the unlimited version of VPFCT was shown to have a smaller leading error constant than the MUSCL-H scheme. There is still work to be done on the VPFCT limiting mechanism. The limiting scheme needs to be improved so that it does not adversely affect convergence. In addition, some overshoots are still appearing in the limited VPFCT solution so more aggressive limiting is needed in some instances. Once these issues are resolved in a satisfactory manner, work will begin on extending the method to the Lagrangian Euler equations.

Figure 7: Initial Compact Vorticity Contours: Taylor-Green Vortex

Figure 8: Compact Vorticity Contours for Taylor-Green Vortex, $t = 3$ units

References

- [1] Morton, K. W. and Roe, P.L.. *Vorticity-preserving Lax-Wendroff-type scheme for the system wave equation*. SIAM J. Sci. Comp., 2001, Vol. 23, No. 1, pp.170-191.
- [2] Book, D.L., Boris, J.P.. *Flux corrected transport: I. SHASTA, a fluid transport algorithm that works*. J. Comp. Phys., 1973, Vol. 11, pp. 38-69.

Appendix A

Definition of compact differencing operators:

$$\delta_x(\cdot)_{j,k} \equiv (\cdot)_{j+1/2,k} - (\cdot)_{j-1/2,k} \quad (63)$$

$$\delta_y(\cdot)_{j,k} \equiv (\cdot)_{j,k+1/2} - (\cdot)_{j,k-1/2} \quad (64)$$

$$\mu_x(\cdot)_{j,k} \equiv \frac{1}{2}[(\cdot)_{j+1/2,k} + (\cdot)_{j-1/2,k}] \quad (65)$$

$$\mu_y(\cdot)_{j,k} \equiv \frac{1}{2}[(\cdot)_{j,k+1/2} + (\cdot)_{j,k-1/2}] \quad (66)$$