An Adaptive Moving Mesh Method with Locally Refined Nested Grids for Partial Differential Equations

James. M. Hyman[1] and Shengtai Li[1,2]

[1] Theoretical Division, Mail Stop, B284
Los Alamos National Laboratory
Los Alamos, NM 87545

[2] Department of Computer Science
University of California
Santa Barbara, CA 93106

September 14, 2004

Abstract

We combine an adaptive moving mesh method with an adaptive mesh refinement (AMR) algorithm using a hierarchical locally refined nested grid to improve the efficiency and accuracy of numerical methods for solving evolutionary partial differential equations in one space dimensions. The local mesh velocity is defined to allow the mesh to track fronts and other dynamic features in the solution. Hierarchical nested grids locally adapt to resolve the spatial features in the solution. The nested grid data structure is also used to locally refine the grid in time. Small time steps are taken on the refined grids and imbedded in the larger time steps taken on the coarser grids. We propose and compare several approaches to provide explicit control over the minimum spacing for the adaptive moving mesh. Numerical experiments are presented to show the effectiveness of our method.

Keywords: Adaptive mesh refinement, moving mesh, numerical methods, partial differential equations.

1 Introduction

The moving mesh (MM) method and adaptive mesh refinement (AMR) method using locally refined nested grids are both effective strategies for adaptively solving partial differential equations (PDEs). In this paper we investigate how to combine these two powerful approaches where the spatial accuracy is maintained by local mesh refinement with a hierarchical data structure.

In [16], an efficient MM method is combined with a static rezone (SR) method based on equidistributing the error estimate on a single (nonhierarchical) global grid. The global data structure for the combined MM-SR method is simple and easy to manage, but the time steps are limited by the smallest grid spacing when implemented via explicit time integration.

When using an AMR hierarchical spatial grid, the MM method must define the grid velocities consistently on all of the levels to prevent the grid from crossing. When the grid is stationary in 1D, the hierarchical data structure can change back to a linear structure [17]. Although this allows more control to prevent the mesh from crossing, it introduces the difficult problem of constructing the base grid of the hierarchical data structure from the information in the linear structure of the MM. These difficulties can be avoided by directly incorporated the hierarchical data structure of AMR into the linear structure of the MM. We achieve this by treating each patch in the AMR hierarchical data structure as a separate problem on a single grid with prescribed moving boundaries. The mesh velocities are then constrained to keep the mesh points within the boundary of the moving patch.

When solving a PDE on a single grid if the grid spacing becomes extremely small, then a global time step for an explicit integration method is limited by stability, not accuracy considerations. Within the AMR nested grid framework we can avoid this problem by using the local time steps commonly used in AMR methods and developing an explicit MM method that can be applied independently on each grid level. Then we treat the mesh position as another solution variable in our AMR algorithm and data structure developed in [17], and compute the mesh velocity for each mesh point by the explicit MM method.

1.1 Gropp's Method

In [8], Gropp proposed another MM-AMR strategy which moves the whole patch as a single unit instead of giving each node a velocity and moving them separately within the patch. Gropp's approach eliminates the possibility of the mesh crossing and, in fact, each patch could remain locally uniform. In [8], the velocity of the patch is specified by the user. For simple problems, where the dynamics of the solution can be anticipated, this is a reasonable, effective and simple strategy. However, for a complex systems of equations, velocity of the patch should be determined dynamically.

A mesh velocity for each node could first be determined by a standard MM method based on equidistributing an error estimate of the spatial error. These velocities could then be averaged to define a single velocity to the patch. Unfortunately, in numerical experiments, this simple approach has not resulted in a good adaptive strategy.

A difficulty with this approach arises when different patches with different velocities collide and produce overlapping grids. Handling the overlapping grids adds significant complexity to the approach, particularly if each grid has a different velocity. The simple solution [8], forcing them to have the same velocity, is not appropriate when the patches are moving in different directions.

Also, the data structure is complex because the node positions of the grids on different levels may not coincide with each other. Because the fine grids do not align with the coarser levels, the regridding and projection steps (see [17]) require more interpolation rather than just copying the solution from different patches.

1.2 Explicit Time Integration

The refined regions with small grid spacing that accompanies most adaptive MM methods (see [11]) and the equation that controls the MM results in a stiff system of differential equations. When the mesh spacing can become extremely small, most efficient general purpose MM methods and software use implicit temporal integration methods. The computational cost, storage and memory requirements to solve the nonlinear system for the implicit method usually dominates the cost of these computations.

Explicit integration methods have more stringent stability requirements, but require much less storage and are more efficient than implicit methods for nonstiff systems. For PDEs, the Courant-Friedrichs-Lewy (CFL) condition limits the maximum size of the time step for an explicit integration method to be proportional to the minimum grid spacing. Because the CFL condition is in the reference frame of the MM, when there are isolated waves in a hyperbolic PDE the condition is not as severe as pointed out by Bell and Shubin [4], than in static mesh methods. However, when there are multiple characteristic velocities going in different directions, then the time step size can still be required to be extremely small. An advantage of MM methods for hyperbolic conservation laws is that some numerical schemes, such as the Godunov scheme [7], have better resolution when the shocks are nearly stationary.

Most of the shock-capturing schemes also contain logical operations and are not suitable for implicit temporal integration.

When only a small portion of the region has a fine grid, it is inefficient to integrate all the grid points with the same time step. To improve the efficiency of the MM-AMR method exploit the AMR hierarchical data structure and take separate time steps on the fine and coarse grids. Even when using this hierarchical time stepping, the explicit integration method

for the MM method will be inefficient unless we restrict the minimum spacing on the coarse grid.

To improve the efficiency of explicit MM methods one can set a relatively large minimum spacing so that the space interval will stop shrinking as soon as it reaches the minimum spacing. The length of the minimum spacing depends on what resolution you want to resolve the fine scale structure. The control of the minimum spacing should have minimal affect on the MM when there is no chance the minimum spacing will be violated on a time step. We will propose two strategies to control the minimum spacing and discuss how they can be applied to MM method and AMR data structure.

After reviewing the basic ideas of MM methods, we discuss and compare the accuracy of exploit integration methods for MM methods. We then describe how we restrict the minimum and maximum grid spacing. Next we describe the AMR hierarchical data structure used to separate the fine and coarse grids. Finally, we illustrate the effectiveness of the MM-AMR methods in numerical experiments.

2 Overview of MM methods

We will describe the MM method for systems of one-dimensional initial value PDEs

$$u_t = f(u, x, t), \qquad x \in [x_0, x_N]. \tag{1}$$

Here the subscripts indicate partial derivative operators, and f is a spatial differential operator (e.g., $f = u_{xx}$ for the heat equation).

In a moving reference frame at the mesh point $x_i(t)$, the solution $u_i(t) = u(x_i(t), t)$ satisfies the equation

$$\dot{u}_i(t) = u_t(x_i(t), t) + u_x(x_i(t), t)\dot{x}_i = f(u, x) + u_x\dot{x}. \tag{2}$$

In the time variation (TV) approach (see [13, 14, 21, 16]) the goal is to move the mesh points so that in the reference frame of the MM the solution is slowly varying and larger time steps can be taken without compromising accuracy. In this approach the grid velocity $\dot{x}$ is chosen to minimize the time rate of change of u and x in the new coordinates,

$$\begin{aligned} \min_{\dot{x}}[|\dot{u}|^2 + \alpha|\dot{x}|^2] &= \min_{\dot{x}}\Big[\sum \dot{u}^2 + \alpha\dot{x}^2\Big] \\ &= \min_{\dot{x}}\Big[\sum (u_t + u_x\dot{x})^2 + \alpha\dot{x}^2\Big]. \end{aligned}$$

Solving this quadratic in $\dot{x}$ at each mesh point for the minimum gives

$$\dot{x} = \frac{-f(u, x, t) \cdot u_x}{\alpha + u_x \cdot u_x}. \tag{3}$$

The mesh can also be moved to reduce spatial errors using the equidistributing mesh (EM) method. In this approach a mesh function is defined to reflect the local spatial truncation error of the PDE. The EM approach requires smaller time steps but is more robust than the TV method because when integrated with a time integration method that controls the temporal truncation errors, the combined method provides a direct control over truncation errors in both time and space (see the review [9] and its references).

Most of the analysis and implementations of the EM-MM method map the solution from the nonuniform moving grid (x) in the physical coordinates to a uniform grind in the stationary computational coordinates (ξ). By introducing a coordinate transformation $x \longmapsto \xi$, the uniform mesh methods can be used as usual in the computational domain, though the grid is nonuniform in the physical domain. Often the computational grid is chosen based on a mesh function such as the arclength of the solution, *i.e.*

$$\xi = \xi(x,t) = \int_{x_0}^{x} \sqrt{1+u_x^2}dx/\theta, \quad \theta = \int_{x_0}^{x_N} \sqrt{1+u_x^2}dx. \tag{4}$$

for a grid with $N+1$ mesh points. The inverse transformation is $x = x(\xi,t)$ and the total time derivative of $u(x(\xi,t),t)$ is

$$\dot{u} = u_t + u_x\dot{x} = f(x,u) + u_x\dot{x}. \tag{5}$$

as in Eq. (2).

After the transformation, we compute the value of u on the uniform computational grid $\{\xi_0, \xi_1, \cdots, \xi_N, \}$, corresponding to the nonuniform physical grid

$$\{x_0, x_1, ..., x_n\} = \{x(\xi_0,t), x(\xi_1,t), \cdots, x(\xi_N,t)\}.$$

Because the grid spacing in the computational domain is uniform the change in ξ is equidistributed between the mesh points

$$\xi(x_i,t) - \xi(x_{i-1},t) = \xi(x_{i+1},t) - \xi(x_i,t) = \frac{\theta}{N}. \tag{6}$$

This approach works for general positive monitor functions $M(x,u)$, where Eq. (6) is equivalent to

$$\int_{x_{i-1}(t)}^{x_i(t)} M(x,t)dx = \int_{x_i(t)}^{x_{i+1}(t)} M(x,t)dx, \;\; i = 1,2,\cdots,N-1 \tag{7}$$

or into discrete form

$$g_i = M_{i-\frac{1}{2}}(x_i - x_{i-1}) - M_{i+\frac{1}{2}}(x_{i+1} - x_i) = 0, \;\; i = 1,2,\cdots,N-1. \tag{8}$$

We now describe some approaches to define a mesh velocity where the equidistributing mesh given by Eq. (8) is global attractor for the ED-MM methods.

3 MM Methods

3.1 TV-MM method

The local TV mesh velocities defined by Eq. (3) may not vary smoothly along the mesh and generates rough irregular grids. Figure 3.1 shows the TV mesh velocity for the solution of Burgers' equation (which we consider in more detail in Section 6). Notice that the mesh

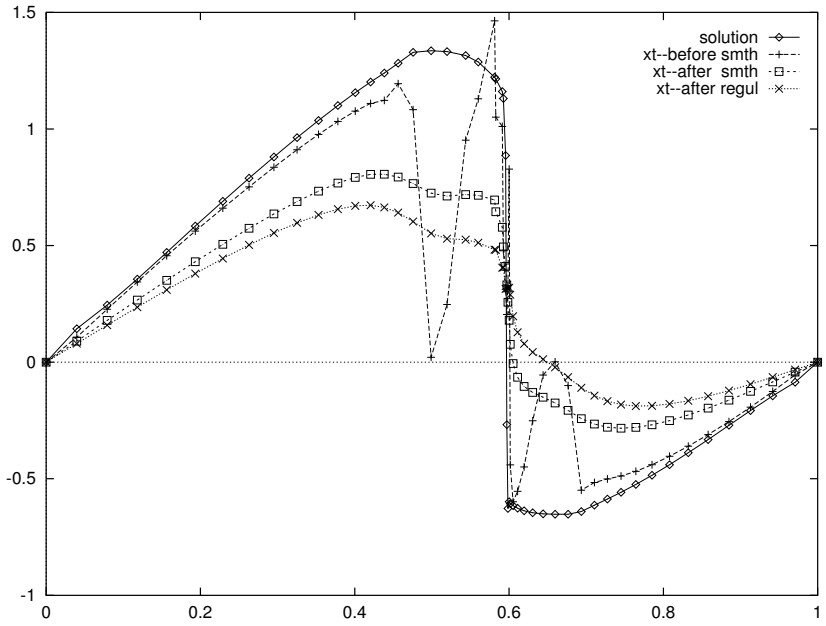

Figure 3.1: The mesh velocity $\dot{x}$ computed by time variation approach can be very rough. The original mesh velocity (+) is shown after smoothing (9) (box) and regularization by (19) (cross).

velocities defined by Eq. (3) changes rapidly where $u_x \approx 0$. We have observed that solutions $\widetilde{\dot{x}}$ of the second order boundary value problem

$$\widetilde{\dot{x}} - \lambda(\widetilde{\dot{x}})_{xx} = \dot{x}, \tag{9}$$

are a reliably smooth mesh velocities. Here $\widetilde{\dot{x}}$ is the new mesh velocity, and λ is a parameter that controls the smoothness of the mesh velocity $\dot{x}$. This approach also gives us explicit control over the local change in the mesh velocities. In particular, the smooth mesh velocity generated by Eq. (9) is shown in Fig. 3.1. Furthermore [12] if

$$\lambda = \kappa(\kappa + 1),$$

then

$$\frac{\kappa}{\kappa+1} \leq \frac{\dot{x}_i}{\dot{x}_{i+1}} \leq \frac{\kappa+1}{\kappa}, \qquad \forall i.$$

The TV approach is not sensitive to spatial errors [16] and may distribute the mesh points so the solution is not well approximated in space. This can be fixed by redistributing the mesh points based on the equidistributing principle.

3.2 ED-MM method

For the ED-MM method, we formulate a differential equation with Eq. (8) as a steady state attractor. The simple MM method [1],

$$\dot{x}_i = -\frac{1}{\tau}(M_{i-\frac{1}{2}}(x_i - x_{i-1}) - M_{i+\frac{1}{2}}(x_{i+1} - x_i)), \tag{10}$$

does this where τ is the time scale that determines how fast the mesh tends to the equidistributing mesh. The MMPDE5 of Huang, et al. [10], is an equivalent formulation and has been found not to be effective as other better scaled methods such as MMPDE6. A discretized form of MMPDE6 is

$$-\dot{x}_{i+1} + 2\dot{x}_i - \dot{x}_{i-1} = -\frac{1}{\tau}(M_{i-\frac{1}{2}}(x_i - x_{i-1}) - M_{i+\frac{1}{2}}(x_{i+1} - x_i)), \tag{11}$$

which yields a tri-diagonal system for the mesh velocity.

3.3 Hyman-Larrouturou ED-MM method

Hyman and Larrouturou propose another equidistributing mesh method [14], which considers the velocity of intervals.

$$\frac{\Delta\dot{x}_{i+\frac{1}{2}}}{\Delta x_{i+\frac{1}{2}}} = \frac{\beta}{\tau}\frac{\overline{m} - m_{i+\frac{1}{2}}}{\overline{m} + m_{i+\frac{1}{2}}}, \tag{12}$$

where $m_{i+\frac{1}{2}}$ is the value of the mesh function in the interval $[x_{i+1}, x_i]$ and is related to the monitor function $m_{i+\frac{1}{2}} = M_{i+\frac{1}{2}}\Delta x_{i+\frac{1}{2}}.$, $\overline{m}$ is the global average value of m, and β determines the relaxation time with respect to the problem time-scale τ. Equation (12) limits the relative mesh velocity to $\pm\beta/\tau$. Because

$$\sum_{0}^{N-1} \Delta\dot{x}_{i+\frac{1}{2}} = \dot{x}_N - \dot{x}_0 = 0, \tag{13}$$

when the value of the local mesh function is less than the average, the interval will increase otherwise, it will decrease.

To normalize Eq. (12) to insure Eq. (13) is satisfied, we solve the modified equation

$$\Delta\dot{x}_{i+\frac{1}{2}} = \frac{\beta}{\tau}\frac{\overline{m} - m_{i+\frac{1}{2}}}{\overline{m} + m_{i+\frac{1}{2}}}\Delta x_{i+\frac{1}{2}} + \varepsilon, \tag{14}$$

where

$$\varepsilon = -\sum(\frac{\beta}{\tau}\frac{\overline{m} - m_{i+\frac{1}{2}}}{\overline{m} + m_{i+\frac{1}{2}}}\Delta x_{i+\frac{1}{2}})/N,$$

Because $\dot{x}_0 = 0$, the mesh velocity can be explicitly solved for recursively in Eq. (14)

3.4 Dorfi-Drury MM method

The Dorfi and Drury method [5] solves Eq. (9) for the mesh concentration $n_i = 1/\Delta x_i$ and its time derivative $\dot{n}_i$. In this formulation, it is difficult to solve for the mesh velocity explicitly. However, because the method is equivalent to globally smoothing the monitor function $M(x,u)$ ([23]), the same mesh velocities can be found by first smoothing the monitor function by solving the second order equation

$$\widehat{M} - \lambda \widehat{M}_{xx} = M. \tag{15}$$

Then, using the new monitor function $\widehat{M}$, computing the mesh velocity by (11) or (14).

Because the smoothness of the grid affects the accuracy of the spatial derivative approximations it is advantageous to generate a smooth monitor function as in the Dorfi-Drury method (Eq. 15), or by a locally smoothing the mesh function such as

$$\widehat{M}_i = \sum_{j=i-p}^{i+p} \Big(\frac{\kappa}{\kappa+1}\Big)^{|i-j|} M_j. \tag{16}$$

This smoothing operator is equivalent to smoothing (15) if p is large enough [23], *i.e*, the smoothing is over all of the points. We typically chose $p = 1$ or 2. If $p \geq 3$ then Eq. (15) is more efficient.

4 Regularization of the Grid Spacing

When using an explicit integration method all MM methods need to prevent the mesh points from coming too close together and creating severe stability restrictions. Mesh functions that monitor the mesh regularity [15] indirectly control the mesh spacing in ED-MM methods to avoid fast relative motions of neighboring mesh points and satisfy prescribed minimum or maximum mesh spacing.

To maintain a minimum spacing $\Delta x_{\min}$ and a maximum spacing $\Delta x_{\max}$, the mesh velocities should satisfy the following conditions

$$\Delta \dot{x}_{i+\frac{1}{2}} = \begin{cases} > 0, & \text{if } \Delta x_{i+\frac{1}{2}} < \Delta x_{\min}, \\ < 0, & \text{if } \Delta x_{i+\frac{1}{2}} > \Delta x_{\max}. \end{cases} \tag{17}$$

4.1 Modified Winkler method

Winkler et al. [24] proposed adding a penalty term to the mesh equation;

$$\widetilde{\dot{x}_i} = \dot{x}_i + \left(\frac{\Delta x_{\min}}{\Delta x_{i+\frac{1}{2}}}\right)^4 - \left(\frac{\Delta x_{\min}}{\Delta x_{i-\frac{1}{2}}}\right)^4 \tag{18}$$

to achieve the minimum spacing. Occasionally, even when $\Delta x_{i+\frac{1}{2}} < \Delta x_{\min}$, $\dot{x}_i$ dominates the penalty term and $\Delta\dot{x}_{i+\frac{1}{2}} > 0$. This situation is avoided by modifying $\Delta\dot{x}_{i+\frac{1}{2}}$ by adding a penalty term

$$\Delta\widetilde{\dot{x}}_{i+\frac{1}{2}} = \Delta\dot{x}_{i+\frac{1}{2}} + \max(|\Delta\dot{x}_{i+\frac{1}{2}}|, 1.0)\left[\left(\frac{\Delta x_{\min}}{\Delta x_{i+\frac{1}{2}}}\right)^4 - \left(\frac{\Delta x_{i+\frac{1}{2}}}{\Delta x_{\max}}\right)^4\right], \tag{19}$$

which ensures that the mesh interval will enlarge as soon as it is less than $\Delta x_{\min}$ or decrease when greater than $\Delta x_{\max}$. The power 4 was chosen based on numerical experiments and allows the mesh to react quickly once the spacing is less than the minimum spacing. The velocity $\dot{x}_i$ or $\Delta\dot{x}_{i+\frac{1}{2}}$ can be defined by any moving mesh method discussed in Section 3 and the prefactor $\max(|\Delta\dot{x}_{i+\frac{1}{2}}, 1.0)|$ is to ensure the factor is sufficiently large if $|\Delta\dot{x}| \approx 0$. Numerical experiments have verified that Eq. (19) offers better control on the minimum and maximum spacing than Eq. (18).

To ensure that mesh velocities (19) satisfy (13) they are normalized as in Eq. (14). Since Eq. (13) cannot be satisfied locally, the regularization (19) may have problem when combined with the AMR hierarchical structure and algorithms.

4.2 Indirect control by monitor function method

On an ED mesh, the maximum of the monitor function $M_{\max}$ occurs at the minimum mesh spacing $\Delta x_{\min}$ so that

$$\Delta x_{\min} M_{\max} = \Delta x_{\max} M_{\min} = \overline{m}, \tag{20}$$

where $\overline{m}$ is the average value of the mesh function and has the same meaning as in (12). To enforce $\Delta x_{\min} \le \Delta x \le \Delta x_{\max}$, $M_{\max}$ or $M_{\min}$ can be easily computed from Eq. (20). Then we enforce these restrictions on the monitor function, so that

$$M_{\min} \le M_{i+\frac{1}{2}} \le M_{\max}, \quad \forall i. \tag{21}$$

This can be achieved by modifying the monitor function as

$$\widetilde{M}_{i+\frac{1}{2}} = \frac{M_{i+\frac{1}{2}} + M_{\min}}{1 + \dfrac{M_{i+\frac{1}{2}} + M_{\min}}{M_{\max}}} - \varepsilon. \tag{22}$$

The rescaled mesh function $\widetilde{M}_{i+\frac{1}{2}}$ has the same monotonicity properties as $M_{i+\frac{1}{2}}$ and satisfies condition (21). The scalar ε is chosen so that $\overline{m}$ is the same after rescaling (22). This is easily accomplished by first defining $\widetilde{M}_{i+\frac{1}{2}}$ with $\varepsilon = 0$, then computing ε by enforcing the sum of mesh function unaltered during the transformation (22).

Although this strategy requires a global variable $\overline{m}$, it can be computed locally and taken as an approximate value to $\overline{m}$ if a local fine grid is given. We will use it in our hierarchical data structure.

5 MM with hierarchical data structure

The minimum grid spacing may have to be small to resolve the solution. If the solution at all the grid points take the same time step, then small refined region can force a small time-step even though we could take a larger step size at other grid points without violating the stability and accuracy restrictions.

The adaptive mesh refinement (AMR) algorithm uses an [3, 2] hierarchical data structure, in which the fine grid is nested in the coarse grid completely. The AMR method allows us to separate the fine grid from the coarse grid and do the integration steps separately. The solution is updated recursively and the coarse grid always has the most accurate solution (projected from the fine grid) for the next time step. The internal boundaries of the fine grid are internal nodes of the coarse grid.

The hierarchical data structure allows us to advance the MM grid equations (described in Sec.3) as just another solution variable. However, the mesh function for spatial refinement no longer includes the smoothness of the grid, except on the coarsest grid. Because the mesh moves with the solution, far fewer SRs are needed than for a fixed grid. The AMR refinement procedure is performed as needed using the data structures described in [17] (based on the approach described in [3]).

The mesh velocities for the nodes of the base grid are computed using one of the MM strategies in Section 3. The mesh velocities for the boundary nodes of the fine grid are interpolated from the immediate (parent) coarse grid. The mesh velocities for the internal nodes on each finer level should be moved to track the internal features of the solution, respect the velocities of the boundary points and remain smoothly spaced.

When combined with AMR hierarchical data structure, the mesh velocities must be computed locally. The MM equations derived with a global normalization step similar to (13) have difficulty to be applied locally. Similarly, the minimum spacing regularization (19) must be normalized globally and is difficult to apply to a local patch. The time variation MM method is local, but the mesh velocities must still be smoothed globally to get a good result. The MMPDE6 (11) does not require smoothing or normalization and the grid velocities are local. Therefore, we use MMPDE6 to compute the mesh velocities in our MM-AMR.

The minimum spacing for the finest grid $\Delta x_{\min}$ (specified by the user) is used to compute the minimum spacing for the coarse grid based on the refinement ratio. For example, if the refinement ratio is r, the minimum spacing for the next finest level grid is $r \times \Delta x_{\min}$.

The scalar hyperbolic conservation equation

$$u_t + f(u)_x = 0,$$

in a moving reference frame

$$\dot{u} + f(u)_x - u_x \dot{x} = 0,$$

has the CFL time step criterion

$$\frac{|\dot{x} - f_u|\Delta t}{\Delta x} \leq 1, \tag{23}$$

where f_u denotes the derivative of f with respect to u. The time step can be relatively large even if Δx is very small, when the mesh velocity $\dot{x}$ is close to the characteristic speed f_u. This often happens in scalar problems where $|\dot{x} - f_u| \approx 0$ near in a step wave front where the mesh spacing is very small. However, for systems it is the spectral norm of $||\dot{x} - f_u||$ that determines the CFL condition. When f_u is a diagonal operator, this is the maximum of the difference between $\dot{x}$ and any of the character velocities. We have only one mesh velocity $\dot{x}$ at each node and even the optimal mesh velocity may only have a minimal improvement in the CFL time step condition when the characteristic velocities are widely separated.

6 Numerical Experiments

We integrate the equation with a second order Runge Kutta method,

$$\widetilde{u}^{n+1} = u^n + \Delta t f(u^n),$$

$$u^{n+1} = \frac{1}{2}(u^n + \widetilde{u}^{n+1}) + \frac{1}{2}\Delta t f(\widetilde{u}^{n+1}).$$

where u^n denotes the solution at the nth time step, and the 3-rd order WENO scheme [20] for the spatial discretization with specification. The MM methods use a smoothed arclength monitor function Eq. (9).

6.1 Burger's equation

Burgers' equation

$$u_t = -uu_x + 0.0001u_{xx}, \quad 0 < x < 1, \quad t > 0,$$

with the initial condition

$$u(x, 0) = 0.5\sin(\pi x) + \sin(2\pi x), \quad 0 \leq x \leq 1,$$

is solved with the boundary conditions $u(0, t) = u(1, t) = 0$.

For all the comparisons, we show the solution at times t = 0.2, 0.6, 1.0, 1,4, 2.0. The coarsest grid has 50 points and u_x in the MM term $u_x\dot{x}$ is discretized by first order central differences. We also supplied a reference solution (dashed line) computed by an implicit MM method with 601 nodes, and a very strict error tolerance.

6.1.1 Explicit MM method for a single grid

In the TV-MM we used Eq. (19) to control the minimum spacing. As expected, unless a static rezone is used with the TV-MM method it fails to adequately resolve the solution. Even with the static rezone, unless it is done every few time steps, the corner at $t = 0.2$ will

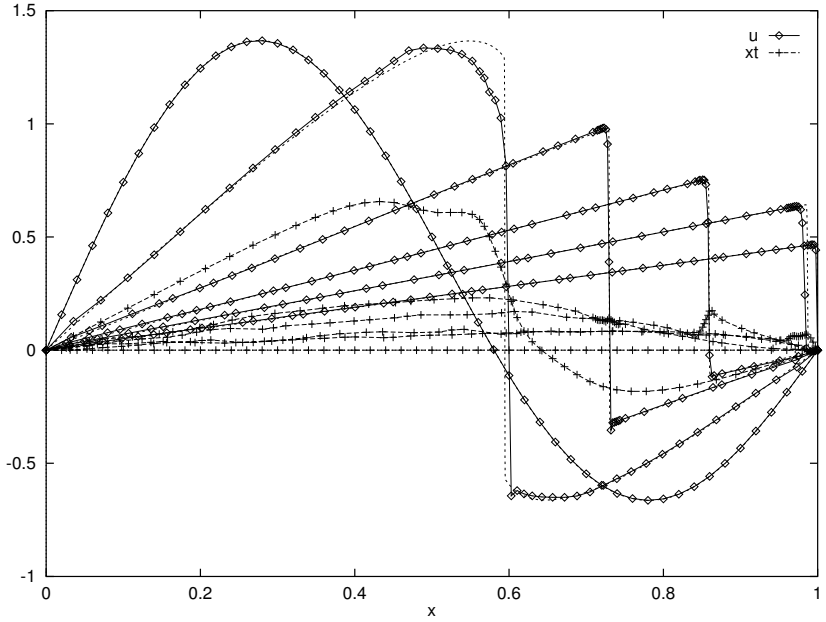

Figure 6.2: Result of time variation approach with static rezone after every 100 time steps.

still be under-resolved (see Fig. 6.2). This is partially because the TV-MM does not move the points to the corner to maintain spatial accuracy during the formation of the shock.

Fig. 6.3-a shows the result of MMPDE6 (11) with $\tau = 0.5$ and minimum spacing control Eq. (22). Fig. 6.3-e shows the result of MMPDE6 with minimum spacing control (19). Even though the mesh distribution was similar in Figs 6.3-a and 6.3-e. the integration was more sensitive to the time scale τ and the smoothness of the monitor function when using (19) than (22). When $\tau = 0.05$, MMPDE6 fails to solve the problem with Eq. (19), whereas it has no difficulty with (22). When the monitor function is not smoothed (see Fig. 6.3-c), the simulation fails when using (19) but still succeeds when using (22). This is because the monitor function transformed by (22) is smoothed simultaneously (Fig. 6.3-d).

The minimum spacing control is essential to the success of the explicit MM method. Figure 6.3-b shows how the minimum spacing control smoothes the mesh velocities near the steep wave front and prevents the mesh from crossing while allowing for large time steps.

The ED-MM methods were insensitive to the time scale τ in (11) or (14) for $[0.01, 1.0]$ if the monitor function is smoothed by Eq. (15). Fig. 6.3-f shows the results of the Hyman-Larrouturou formulation (14) with $\tau = 0.05$ and minimum spacing control (22). We observed that the normalization (13) was needed to keep the mesh from crossing when using the

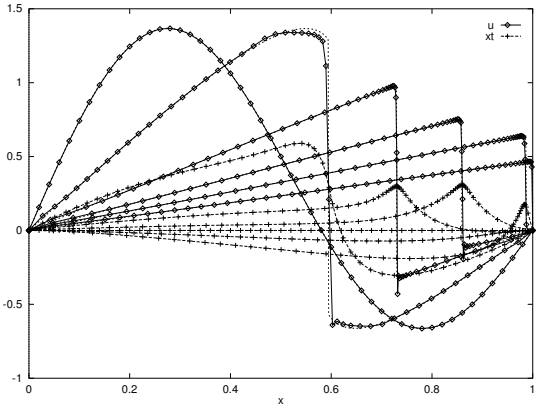

Figure 6.3-a: Result of MMPDE6 (11) with $\tau = 0.5$ and minimum spacing control (22).

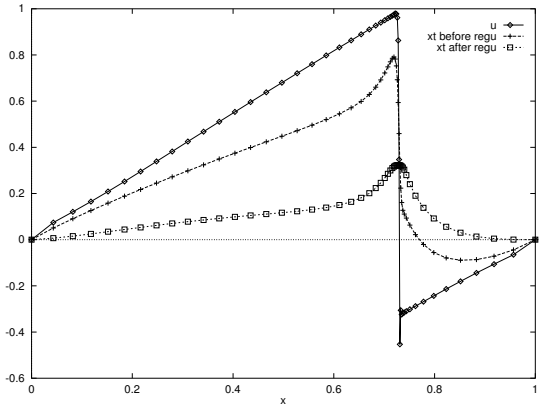

Figure 6.3-b: Result of minimum spacing control (22) at $t = 0.6$ for Fig. 6.3-a.

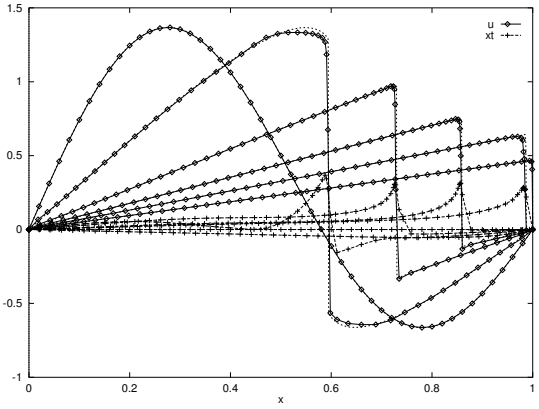

Figure 6.3-c: Result of MMPDE6 (11) with $\tau = 0.5$, minimum spacing control (22) and no smoothing on monitor function.

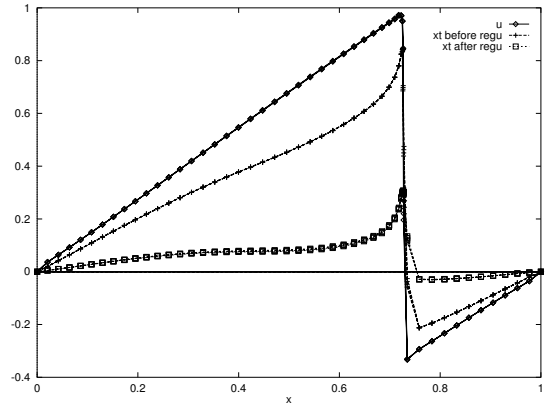

Figure 6.3-d: Result of the regularization (22) at time $t = 0.6$ for Fig. 6.3-c.

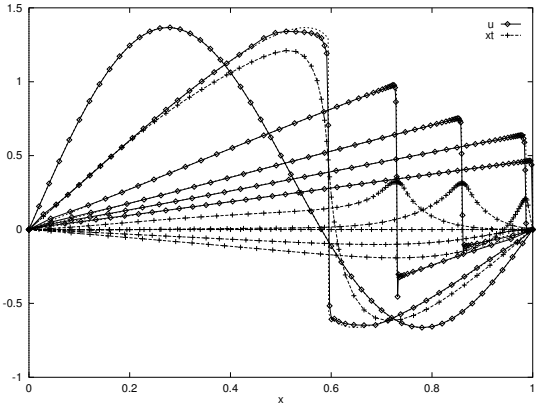

Figure 6.3-e: Result of MMPDE6 (11) with $\tau = 0.5$ and minimum spacing control (19).

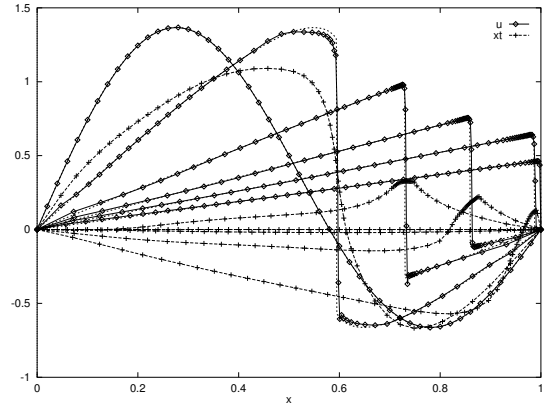

Figure 6.3-f: Result of Hyman and Larrouturou formulation (14) with $\tau = 0.05$ and minimum spacing control (22).

Hyman-Larrouturou Eq. (12).

The normalization (13) must be done globally for the approach (19), because (13) is not satisfied locally. However, the regularization (22) can be done locally without any difficulty. Therefore, we use (22) in all of our MM-AMR simulations.

6.1.2 MM-AMR with hierarchical data structure

When using a single nonuniform mesh, the global time step is chosen based on the CFL conditions for all of the mesh points. In this example, only $\frac{1}{4}$ of the mesh points that have a spacing less than three-fourths of the spacing of the uniform mesh that has the same number of nodes. An advantage of the AMR hierarchical data structure is that it allows smaller time steps on the fine scale mesh points.

We use a three-level hierarchical data structure with $r = 2$. We set the minimum spacing for the finest grid to 0.001 so the minimum spacing is 0.002 for the second level and 0.004 for the base grid. The integration for each level is done recursively and the time step for the fine grid is computed based on the CFL condition in that level. The result shown in Fig. 6.4-a is for 50 points in the base grid using the MM method (11) with minimum spacing control (22). The CPU time cost is 65% of that of a single mesh with 100 grid points without using the hierarchical data structure. The solution (Fig. 6.4-a) is more accurate than the MMPDE6 result in Fig. 6.3-e.

The accuracy is maintained even when the number of nodes in the base grid is reduced to 20 (see Fig. 6.4-b).

The mesh velocity $\dot{x}$ is not as smooth as before the node positions in the coarse grid are projected from the fine grid(see Fig. 6.4-c). The mesh velocity for each level at time $t = 0.2$ is plotted in Fig. 6.4-d.

The smoothness of the mesh velocity can be retained by adopting a relaxed time scale τ. Fig. 6.5 shows the mesh velocity at $t = 0.2$ with $\tau = 1.0$.

In our numerical experiments on Burger's equation we found that although the numerical solutions were insensitive to $\tau \in [0.1, 1.0]$, the discontinuity was better resolved if τ increased with the number of grid points and/or a number of refinement levels.

The numerical error at $t = 2.0$ (Table 1) was defined by interpolating the numerical solutions to a fine grid and comparing them to a highly resolved reference solution. The maximum error is relatively large because of the errors near the discontinuity.

6.2 Euler equations of gas dynamics

The Euler equations for gas dynamics

$$\rho_t + (\rho u)_x = 0,$$

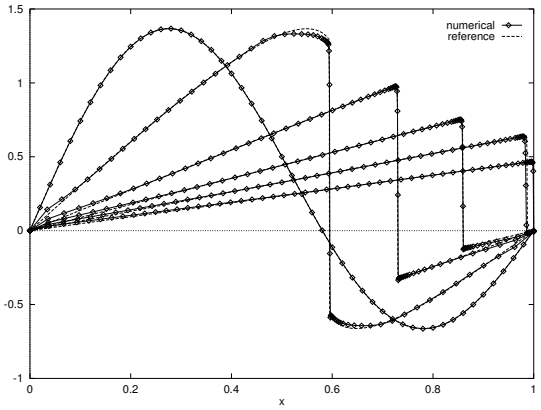

Figure 6.4-a: Result of MM combined with the hierarchical data structure. The number of base grid points is 50. $\tau = 0.2$

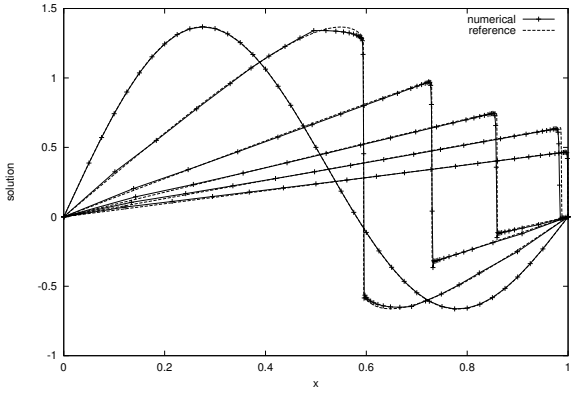

Figure 6.4-b: Result of MM combined with the hierarchical data structure ($\tau = 0.1$). The number of base grid points is 20.

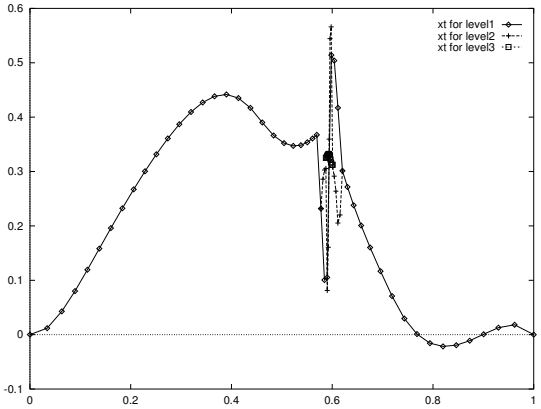

Figure 6.4-c: Mesh velocity computed by MM-AMR ($\tau = 0.2$)

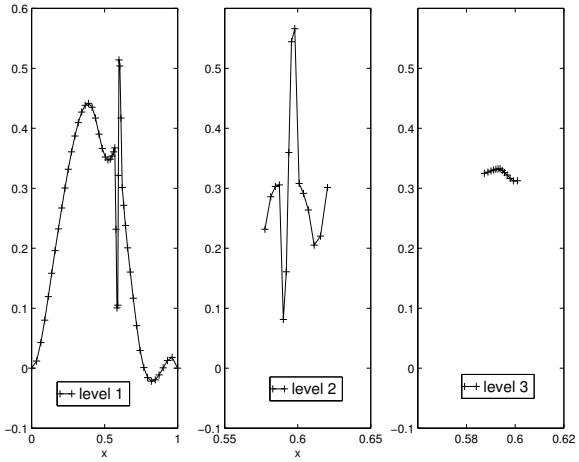

Figure 6.4-d: Mesh velocity for each level at time $t = 0.2$ ($\tau = 0.2$).

ndx	numlev	NSTEP	L_2-err	M-err	CPU
100	1	2300	0.0061	0.23	7.547
50	2	1178	0.0064	0.22	3.328
25	3	495	0.0073	0.24	2.222

Table 1: Performance comparison for difference numbers of grid points with different refinement levels at time $t = 2.0$.

$$\begin{aligned} (\rho u)_t + (p + (\rho u)^2/\rho)_x &= 0, \\ e_t + ((e+p)(\rho u)/p)_x &= 0, \end{aligned}$$

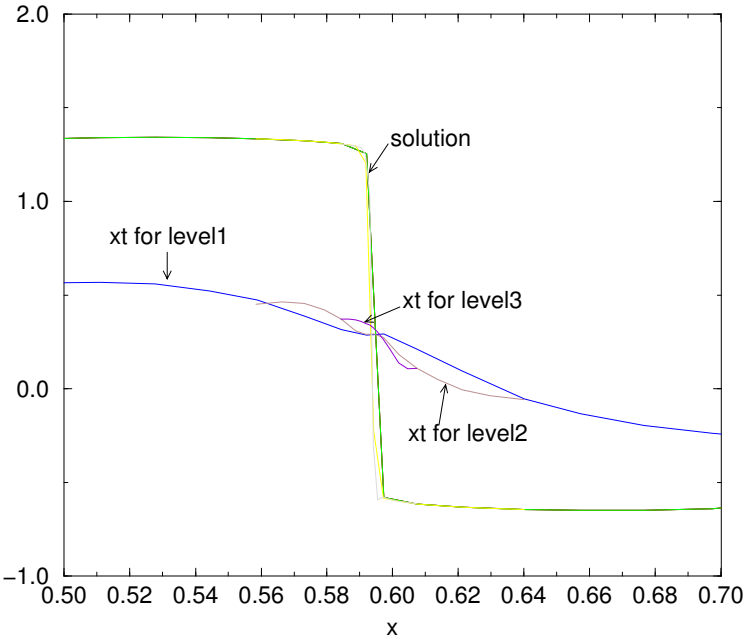

Figure 6.5: Mesh velocity for each level at time $t = 0.2$ ($\tau = 1.0$).

with initial conditions [22]

$$(\rho, (\rho u), e) = \begin{cases} (1.0, 0.0, 2.5), & \text{if } x \leq 0.5, \\ (0.125, 0.0, 0.25), & \text{if } x > 0.5. \end{cases}$$

is solved by the AMR method in [17]. We solve the equations with the MM-AMR method with 100 base grid points and three refinement levels. The minimum spacing for the finest level is 0.001.

In Fig. (6.6-a) the solution decomposes into a shock wave, a contact discontinuity and a long rarefaction wave. If the arc length monitor is used, more points will be distributed to the rarefaction wave, which is not necessary and leave the contact and shock under-resolved. Hence we use the modified curvature monitor in our computations, which is

$$M(x, u) = \sqrt{\alpha + N(\Delta u_x)^2}, \tag{24}$$

where we choose $\alpha = 1.0$. Numerical results show that this monitor greatly improves the accuracy at the rarefaction corner, contact discontinuity and shock.

The contact discontinuity is not resolved as well as the shock because the shock is discontinuous in all the variables and has more weight than the contact in our mesh function.

We also solved this problem with one refinement level, *i.e.*, pure explicit MM method without using the hierarchical data structure. The minimum spacing is 0.001. The results

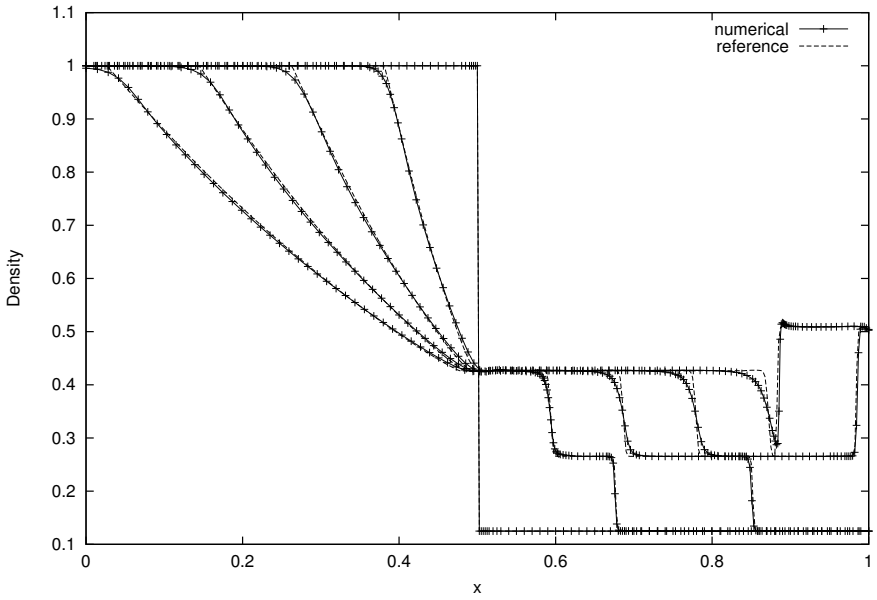

Figure 6.6-a: MM-AMR method for the shock tube problem. The number of base grid points is 100 with three refinement levels are used. $\tau = 1.0$.

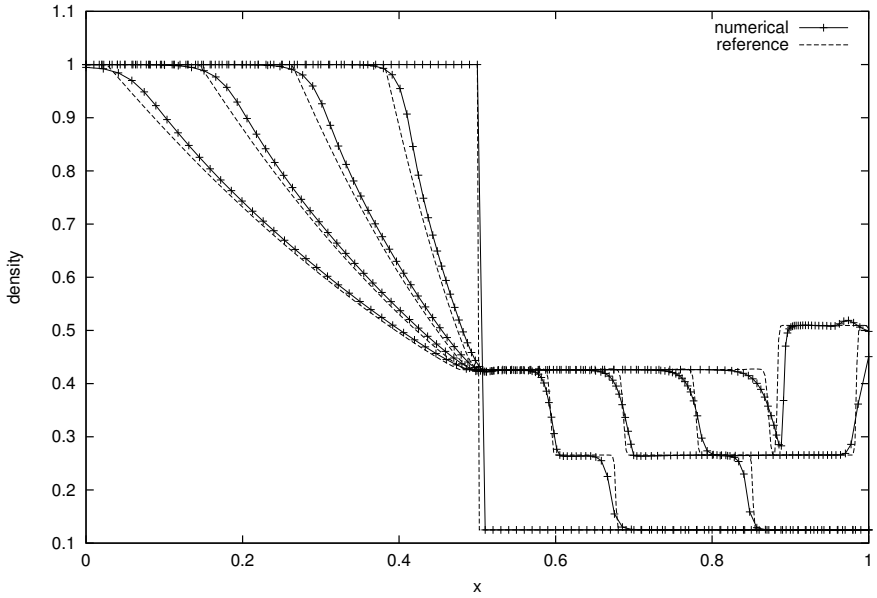

Figure 6.6-b: Explicit MM-AMR method for shock tube problem with only one refinement level. The number of grid points is 100. $\tau = 0.5$

with the time scale $\tau = 0.5$ are shown in Fig. 6.6-b. For this problem the results were even worse for other values of τ. The single level ED-MM method failed to solve this problem with $\tau = 0.2$ because of mesh crossing, but the MM-AMR method with three refinement levels had no problem.

The MM-AMR method is more robust and accurate than the single level ED-MM method. The CPU time increases only by 1/2.

7 Conclusion

We have described several approaches to formulate the MM methods and to control the minimum or maximum spacing. An improved explicit MM method which exploits the AMR hierarchical data structure is proposed and demonstrated to be effective in solving Burgers' equation and Euler equations. The minimum spacing control and hierarchical data structure overcome the small time-step problems with the explicit MM method.

The MM-AMR introduces the overhead of both AMR and MM method. We had compared the results of the AMR, moving mesh, MM-AMR methods for the same Burgers' equation. The MM-AMR takes more CPU time than either the MM or AMR method. For the Euler equations, the MM-AMR is slightly slower than the AMR method but much faster than the implicit MM method. We also noticed that the reduced time variation by the mesh moving is not fully-used by our MM-AMR. The time step size is mostly determined by the stability instead of accuracy. To effectively take the advantages of both AMR and MM method, an implicit AMR algorithm should be developed.

References

[1] J. Baumgarte, Stabilization of constraints and integrals of motion in dynamical systems, *Comp. Meth. Appl. Mech. Eng.* **1** (1972), 1-16.

[2] M. J. Berger and P. Colella, Local adaptive mesh refinement for shock hydrodynamics, *J. Comput. Phys.* **82** (1989), 64-84.

[3] M. J. Berger and J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, *J. Comput. Phys.* **53** (1984), 484-512.

[4] J. B. Bell and G. R. Shubin, An adaptive grid finite-difference method for conservation laws, *J. Comput. Phys.* **52** (1983), 569-591.

[5] E. A. Dorfi and L. O'c. Drury, Simple adaptive grids for 1-D initial value problems, *J. Comput. Phys.* **69** (1987), 175-195.

[6] P. R. Eiseman, Adaptive grid generation, *Computer Methods in Applied Mechanics and Engineering* . **64** (1987), 321-376.

[7] S. K. Godunov, A finite-difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics, *Mat. S.* . **47** (1959), 271-290.

[8] W. D. Gropp, Local uniform mesh refinement with moving grids, *SIAM J. Sci. Stat. Comput.* **8** (1987), 292-304.

[9] D. F. Hawken, J. J. Gottlieb and J. S. Hansen, Review of some adaptive node-movement techniques in finite-element and finite-difference solutions of partial differential equations, *J. Comput. Phys.* **95** (1991), 254-302.

[10] W. Z. Huang, Y. Ren and R. D. Russell, Moving mesh partial differential equations (MMPDEs) based on the equidistribution principle, *SIAM J. Numer. Anal.* **31** (1994), 709-730.

[11] W. Z. Huang, Y. Ren and R. D. Russell, Moving mesh methods based on moving mesh partial differential equations, *J. Comput. Phys.* **113** (1994), 279-290.

[12] W. Huang and R. D. Russell, Analysis of moving mesh partial differential equations with spatial smoothing, Simon Fraser University Mathematics and Statistics Research Report 93–17 (Burnaby B.C. V5A 1S6, Canada, 1993).

[13] J. M. Hyman, Adaptive moving mesh methods for partial differential equations, in Advances in Reactor Computations, American Nuclear Society Press, La Grange Park, IL, 24-43, 1983.

[14] J. M. Hyman and B. Larrouturou, Dynamic rezone methods for partial differential equations in one space dimension, *Appl. Numer. Math.* **5** (1989), 435-450.

[15] J. M. Hyman and M. J. Naughton, Static rezone methods for tensor-product grids, in Proceedings of SIAM-AMS Conference on Large Scale Computations in Fluid Mechanics, SIAM, Philadelphia, 1984.

[16] J. M. Hyman, S. Li and L. R. Petzold, An adaptive moving mesh method with static rezoning for partial differential equations, Los Alamos National Laboratory Report (1998)

[17] J. M. Hyman and S. Li, Interactive and dynamic control of adaptive mesh refinement with nested hierarchical grids, Los Alamos National Laboratory Report (1998)

[18] S. Li, L. Petzold and Y. Ren, Stability of moving mesh systems of partial differential equations, Report TR96-027, Department of Computer Science, University of Minnesota, 1996. To appear in SIAM *J. Sci. Comput.*

[19] S. Li and L. R. Petzold, Moving mesh methods with upwinding schemes for time-dependent PDEs, *J. Comput. Phys.* **131** (1997), 368-377.

[20] X-D. Liu, S. Osher and T. Chan, Weighted essentially non-oscillatory schemes, *J. Comput. Phys.* **115** (1994) 200-212.

[21] L. R. Petzold, Observations on an adaptive moving grid method for one-dimensional systems of partial differential equations, *Appl. Numer. Math.* **3** (1987), 347-360.

[22] G. A. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, *J. Comput. Phys. .* **43** (1978) 1-31.

[23] J. G. Verwer, J. G. Blom, R. M. Furzeland, and P. A. Zegeling, A moving-grid method for one-dimensional PDEs based on the method of lines, in Adaptive Methods for Partial Differential Equations, J. E. Flaherty, P.J. Paslow, M. S. Shephard, and J. D. Vasilakis, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1989.

[24] K. H. Winkler, D. Mihalas and M. L. Norman, Adaptive-grid methods with asymmetric time-filtering, Max-Plank-Institut fur Physik und Astrophysik, Munchen, 1984.