Essentially Non-Oscillatory Adaptive Tree Methods

Thomas C. Cecil[*], Stanley J. Osher[†], Jianliang Qian[‡]

September 19, 2007

Abstract

We develop high order essentially non-oscillatory (ENO) schemes on non-uniform meshes based on generalized binary trees. The idea is to adopt an appropriate data structure which allows to communicate information easily between unstructured data structure and virtual uniform meshes. While the generalized binary trees as an unstructured data structure can store solution information efficiently if combined with a good adaptive strategy, virtual uniform meshes allow us to take advantage of many well-developed ENO numerical methods based on uniform meshes. Therefore, the ENO adaptive tree methods proposed here can leverage the merits from both tree structures and uniform meshes. Numerical examples demonstrate that the new method is efficient and accurate.

1 Introduction

Since its inception in [11], essentially non-oscillatory (ENO) schemes have been very influential in numerical solutions for nonlinear hyperbolic equations. The original design principle of ENO schemes for nonlinear conservation laws is to generalize the Godunov scheme: reconstruct an essentially non-oscillatory piecewise polynomial of the solution from its cell averages, evolve in time through an approximate solution of the resulting initial value problem, and average the approximate solution over each cell. Consequently, this pioneering work was built upon an adaptive algorithm to choose a local stencil among several possible candidates so that the resulting polynomial interpolant yields high-order accuracy whenever the function is smooth but avoids the Gibbs phenomena at discontinuities.

[*]tcecil@ices.utexas.edu ICES, UT Austin, Austin, TX 78712.

[†]sjo@math.ucla.edu Department of Mathematics, University of California, Los Angeles, 405 Hilgard Avenue, Los Angeles, CA 90095-1555.

[‡]qian@math.wichita.edu Department of Mathematics and Statistics, Wichita State University, Wichita, KS 67260-0033.

Following this beautiful idea, many researchers have improved the methodology and expanded the area of its applications; see [27] and references therein for an overview of ENO and its applications. Because it is costly to evaluate cell averages accurately in high dimensional spaces and it is desirable to have the time evolution operator that does not increase the total variation of the solution, Shu and Osher [28, 29] have introduced efficient ENO schemes based on nodal values rather than cell averages and total variation diminishing (TVD) Runge-Kutta time discretizations, which can save computational costs tremendously in multi-dimensional spaces; however, the efficiency of such approaches critically hinges on uniform meshes. In this work we propose an adaptive tree based generalization of ENO schemes so that uniform mesh based ENO schemes can be used on unstructured meshes naturally.

Multi-dimensional ENO schemes based on unstructured meshes were developed in [5, 1]. Casper and Atkins [5] have developed their finite volume approach by assuming a smooth rectangular transformation between a reference Cartesian mesh and their computational grid. Abgrall [1] has developed a class of ENO schemes on general unstructured meshes based on a quasi minimal family of candidate stencils; such ENO schemes on unstructured meshes are quite complicated, depending on the complexities of the grid structure.

As a consequence, the question is: can we develop ENO schemes on non-uniform meshes as efficient as those on uniform meshes designed by Shu and Osher [28, 29]? As pointed out by Merriman in his enlightening paper [17] on understanding the Shu-Osher conservative finite difference form, it is still an outstanding problem to make the Shu-Osher form compatible with robust adaptive gridding in the development of the highly successful class of ENO methods. To that end, in this work we extend ENO schemes to grids that have abrupt changes in cell size by making use of generalized binary trees in arbitrary dimension, retaining all the desirable properties of accuracy, simplicity and shock-capturing capability, so that the resulting family of ENO schemes is as efficient as the original Shu-Osher and is compatible with adaptive gridding.

In an earlier work [6] we have developed first order monotone finite difference schemes on generalized binary trees. To extend those first order schemes to high order ENO schemes on non-uniform meshes based on generalized binary trees, the idea is to adopt an appropriate data structure which allows to communicate information easily between unstructured data structure and virtual uniform meshes. While the generalized binary trees as an unstructured data structure can store solution information efficiently if combined with a good adaptive strategy, virtual uniform meshes allow us to take advantage of many well-developed ENO numerical methods based on uniform meshes [20, 13, 12, 25]. Therefore, the ENO adaptive tree methods proposed here can leverage the merits from both tree structures and uniform meshes.

2 Tree Data Structure

In this section we describe the tree data structure used. We use a generalized binary tree (e.g. quad-tree in 2d, octree in 3d, etc.) data structure, details of which can be found in numerous computer science texts [14],[24],[8]. We describe the portions of the implementation that are specific to our problem of solving a PDE in a bounded spatial domain.

Consider a computational domain, $\Omega = [0,1]^n$. At the k^{th} level of the tree, each node, c, represents a hypercube cell with sides of length $dx_c = 1/2^k$, and center x_c. We assume that the function value, $\phi(x_c)$, is stored at the center of mass of the nodes at the finest level of the tree, also known as the leaves. An alternate storage location could be the vertexes of the cells. We choose the centers because of the simplicity of implementation (e.g. there are no storage points belonging to multiple cells), and the fact that local interpolations can be done in a way that is easily generalized to higher dimensions.

When refinement of a cell is done, the cell is split into 2^n subcells with side lengths $1/2^{k+1}$. We do not allow any cells with side length ratio > 2 or < 0.5 to be neighbors. This restriction results in what is known as a balanced tree; see [19] for some results concerning tree balancing. For higher order numerical flux methods (such as 6-point WENO) we impose

more stringent balancing, restricting the above size ratio not only on the neighbors of a cell, but also on the neighbors of neighbors of a cell.

In the interface motion based on level set formulation, this balancing can be obtained by following the criterion of refining any cell whose distance to the interface, Γ, is less than a constant times its edge length [30]. In practice, if we are using a single level set function ϕ e.g. for co-dimension-one problems, and if ϕ is a signed distance function then we can set $\rho \geq (1+\sqrt{n}/2)$ and refine if $|\phi(x_c)| < \rho dx_c$. If the intersection of multiple level set functions, $\{\phi_j\}$, represents Γ, where the level sets of the ϕ_j are mutually orthogonal and each ϕ_j is a distance function measured along the level sets of the other $\phi_{i\neq j}$, we can compare $\|\phi\|_{l_2}$ to ρdx_c.

3 Brief Review of 1d ENO Polynomial Interpolation

In this section we briefly review the procedure of constructing 1d ENO polynomial interpolants. During the evolution of the grid we will use 1d ENO interpolation to determine function values at both parents of cells, as well as at newly created child cells.

For our purposes we will always assume that we have a point x_0 where we will evaluate the interpolant, $P(x)$, and that x_0 is between 2 values: $x_{-1} < x_0 < x_1$ that are on the stencil of points used to construct P. The stencil $\{x_{-1}, x_1\}$ is our starting stencil. We define $L = -1, R = 1$ as the left and right endpoints of our stencil. We denote the function values used in the interpolation by $f_i \equiv f(x_i)$. We predetermine a maximum polynomial interpolation degree r and construct a set of candidate stencil points of size $2r$: $x_{-r} < x_{-r+1} < \cdots < x_{r-1} < x_r$, where r of these points are less than x_0, and r are greater than x_0.

The procedure for constructing P is then as follows:

1. If $R-L < r+1$, then continue to step 2. Otherwise the interpolation stencil $\{x_L, \ldots x_R\}$ is completely found and $P(x_0)$ can be evaluated.

2. Form the divided differences $a = f[x_{L-1}, \ldots x_R]$, and $b = f[x_L, \ldots x_{R+1}]$.

3. If $|a| < |b|$ then set $L = L - 1$, otherwise set $R = R + 1$.

4. Go to step 1.

Note that this procedure is almost the same as the standard procedure for ENO interpolation; see [27].

4 Parent Cell Interpolation with ENO

In this section we describe the ENO interpolation method used to find a value at the center of a cell c when c is not a leaf cell. In [6] we used the 2^n point average of the value of all first children of c for the value at c. While this is a monotone interpolation scheme, it may smear discontinuities. Here, we introduce a more accurate reconstruction method using 1d ENO polynomial interpolations along the extended diagonals of the hypercube c.

The basic idea is that we will find a 1d ENO polynomial interpolation along each of the 2^{n-1} diagonals, $\{d_j\}$, of the n dimensional hypercube, c. This will give 2^{n-1} candidate values, $\{\phi_j(x_c)\}$, for $\phi(x_c)$. From this set of candidates we choose $\phi(x_c)$ to be the average of $\{\phi_j(x_c)\}$ such that

$$\phi(x_c) = \sum_{j=1}^{2^{n-1}} \alpha_j \phi_j(x_c) \tag{1}$$

where $\alpha_j = 2^{-(n-1)}$. Other convex combinations could be used in practice, such as choosing the α_j based on a measure of smoothness of the interpolating polynomial ϕ_j.

For example in 2d, Figure 1 shows the interpolation stencils used when the value at w_0 is to be found at the center of the cell with dashed edges. If we use 2-point linear interpolations the points used for the polynomial stencil for ϕ_1 would be $\{w^1_{-1}, w^1_1\}$ with function values $\{\phi(w^1_{-1}), \phi(w^1_1)\}$, and the points used for the stencil for ϕ_2 would be $\{w^2_{-1}, w^2_1\}$ with function values $\{\phi(w^2_{-1}), \phi(w^2_1)\}$. In order to put the interpolation into a 1d setting we reassign the independent variables using a function $d : \mathbb{R}^n \to \mathbb{R}$, given by $d_{\hat{e}}(w) = sgn(w \cdot \hat{e})\|w - w_0\|_2$. Here $\hat{e}$ is a chosen unit vector along a particular axis (in this example $\hat{e} = (0, 1)$, noting

that the choice of $\hat{e}$ does not affect the resulting interpolated value), and sgn is the signum function. The interpolated parent value in this case is

$$\phi(w_0) = 0.25(\phi(w^1_{-1}) + \phi(w^1_1) + \phi(w^2_{-1}) + \phi(w^2_1)).$$

If we instead use ENO interpolation with 3-point sub-stencils then ϕ_1 will use a stencil constructed from the points $\{w^1_k\}_{k=-2:2,k\neq 0}$, and ϕ_2 will use a stencil constructed from the points $\{w^2_k\}_{k=-2:2,k\neq 0}$.

Because of the grading of the tree, although the stencils are nonuniform, there are only a limited number of stencils which can occur, allowing for a lookup table to be constructed for faster interpolation speed.

WENO interpolations could also be constructed and one could use the smoothness indicators calculated in determining the WENO sub-stencil weights to help choose the α_j in equation (1).

The extension from 2d to other dimensions is straightforward.

Within the tree data structure a postorder traversal call of the interpolate parent function will recursively calculate the unknown parent values starting at the parents of the leaf cells and ending at the root of the tree. For example, this traversal would be called after a single forward-Euler PDE time-step advancement were completed at all leaves of the tree.

5 Child Cell Interpolation with ENO

In this section we describe the ENO interpolation method used to find a value at the center of a child cell c, with parent C, when c has been created but not yet assigned a value, e.g. when cell refinement has just occurred. In [6] we used the two-point interpolation based on a weighted average consisting of the value at x_C and the value of the nearest neighbor cell along the ray $r = x_C + \tau(x_c - x_C), \tau > 0$. Here, 1d ENO interpolation is used along r.

For example in 2d, Figure 2 shows the interpolation stencils used when the value at w_0 is to be found at the center of the cell with dashed edges. The method of [6] would use the

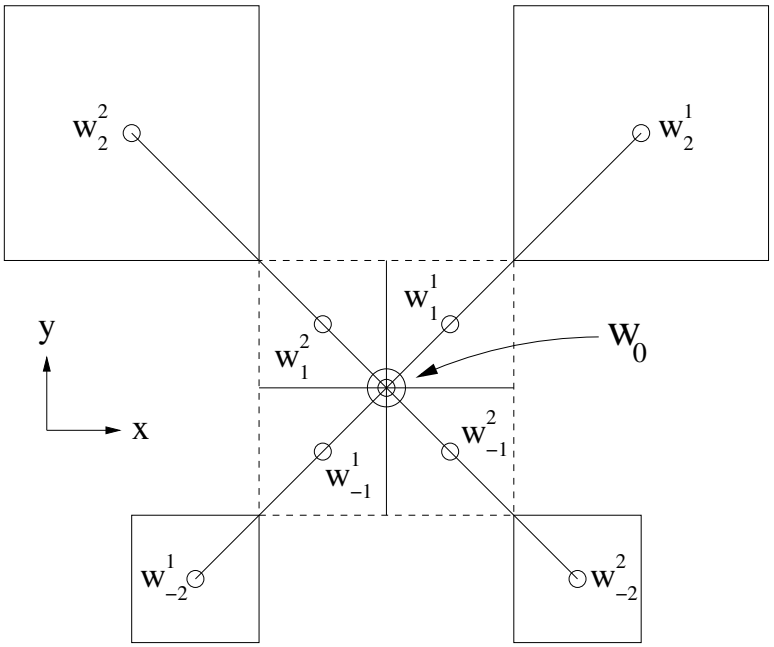


Figure 1: Interpolation stencils $\{w_k^1\}_{k=-2:2,k\neq 0}, \{w_k^2\}_{k=-2:2,k\neq 0}$ used in computing the value at the parent cell with dashed edges having center $x = w_0$.

2-point stencil $\{w_{-1}, w_1\}$ with function values $\{\phi(w_{-1}), \phi(w_1)\}$. For an ENO interpolation with three-point sub-stencils the big stencil would use stencils constructed from the points $\{w_k\}_{k=-2:2,k\neq 0}$, and for an ENO interpolation with four-point sub-stencils the big stencil would use stencils constructed from the points $\{w_k\}_{k=-3:3,k\neq 0}$.

As in the parent interpolation case we use the mapping $d_{\hat{e}}(w)$ to convert our n dimensional stencil to a 1d stencil (for the example in Figure 2 $\hat{e} = (1, 0)$). Again, there are only a limited number of possible stencils allowing for lookup tables and WENO interpolations to be generated, and the extension from 2d to other dimensions is straightforward.

6 Computing numerical fluxes and numerical gradients

When computing numerical fluxes for hyperbolic conservation laws or numerical gradients for Hamilton-Jacobi equations we are able to use high order approximations constructed for uniform meshes such as ENO and WENO because of the graded tree data structure.

For example in Figure 3 we wish to calculate an approximation $D_y^+\phi(w_0)$ of $\phi_y(w_0)$ using

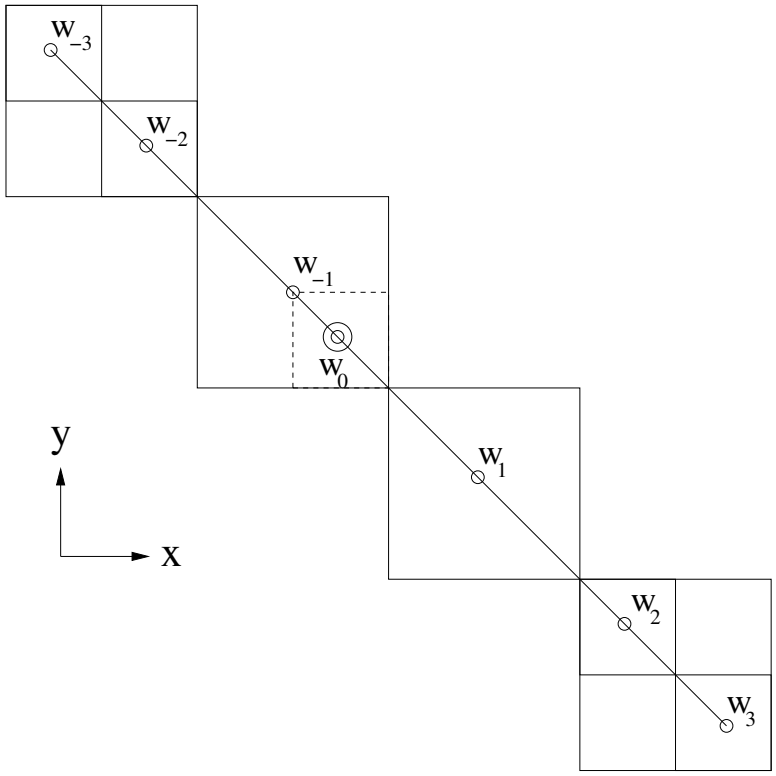


Figure 2: Interpolation stencil $\{w_k\}_{k=-3:3,k\neq 0}$, used in computing the value at the child cell with dashed edges having center $x = w_0$.

6-point WENO interpolation with the stencil $\{w_k\}_{k=-2:3}$ (this employs three different 4-point sub-stencils). This would be the case for solving a Hamilton-Jacobi problem if we have chosen a monotone numerical Hamiltonian which is a function of higher order approximations of $D_y^+\phi(w_0)$ and $D_y^-\phi(w_0)$ of $\phi_y(w_0)$; see [20] for examples of such numerical Hamiltonians.

In general we will need to obtain values at the points x_i^k that are offset in the i^{th} dimension from the cell with center x^0 ($\equiv w_0$ in Figure 3) in order to construct WENO interpolations of ϕ_{x_i}. The stencil used in calculating ϕ_{x_i} consists of the points, x^k, that are defined by

$$x^k \equiv x^0 - \, k\, dx_i\, e_i,\; k = -r : r+1. \tag{2}$$

Note that for fifth order WENO interpolations $r = 2$, and for third order WENO interpolations $r = 1$. The resulting value of ϕ_{x_i} is then calculated in exactly the same way as described in [21, 27].

In Figure 3 all cells with single circles at their center are leaf cells, while those cells with double circles at their centers are either nonexistent leaf cell children (in the case of w_1, w_2, w_3), or leaf cell parents (in the case of w_{-2}). Here, only the cell with center w_{-1} is of the same size as the cell centered at w_0. However, this is not a problem given that:

1. We have values at the parent cells of all nodes, so that $\phi(w_{-2})$ has already been calculated and is known.

2. We can easily construct values at the non-existing cells centered at the points w_1, w_2, and w_3, cells that are of the same size as the cell centered at w_0. For this example we show the stencils for two-point interpolations only; e.g. $\phi(w_1)$ is calculated using leaf cells centered at P_3, P_4, but higher order ENO interpolations could of course be used.

Here we can see why we require our tree to be graded. If it were not, interpolations could become much more complicated. For example, if the cells centered at points P_1, P_2, P_3, and P_5 were joined together, then the interpolations at points w_1, w_2, and w_3 could not be computed in the systematic way as described in Section 5.

We now describe the simplest general procedure for obtaining the neighboring values at w_k in Figure 3.

1. Assume that we need the value of ϕ at a point w_k which is a cell center at the same tree level as the cell c_0 with center w_0. We will say that c_0 has edge length dx_0 for ease of notation.

2. Attempt to find the finest existing cell , c, (that is at least as large as c_0) that contains w_k.

   (a) If c is at the same level as c_0 then we have $\phi(w_k)$ already calculated, which will happen if c contains children cells or is a leaf cell itself, e.g. w_{-2}, w_{-1} in Figure 3. Thus we are finished.

   (b) If c is larger than c_0 then we must use interpolation to determine $\phi(w_k)$, e.g. w_1, w_2, and w_3 in Figure 3. For 2-point interpolation as shown in the example we use the point w_c (the center of cell c), and the point at the center of the finest cell which contains

   $$w_k + \frac{w_k - w_c}{|w_k - w_c|}\sqrt{n}(dx_0 + \epsilon), \tag{3}$$

   where ϵ, a positive number, is smaller than the finest grid size used in the tree. Here ϵ is used to ensure that the point being found is in a neighboring cell (not the cell where w_c is). Note again that the grading of the tree allows this formula to be valid in all cases where c is larger than c_0.

Similarly, we may use the above strategy to evaluate numerical flux functions on a virtual uniform mesh so that well-established ENO/WENO type conservative numerical schemes constructed for uniform meshes [28, 29, 13] can be applied in the adaptive computation right away.

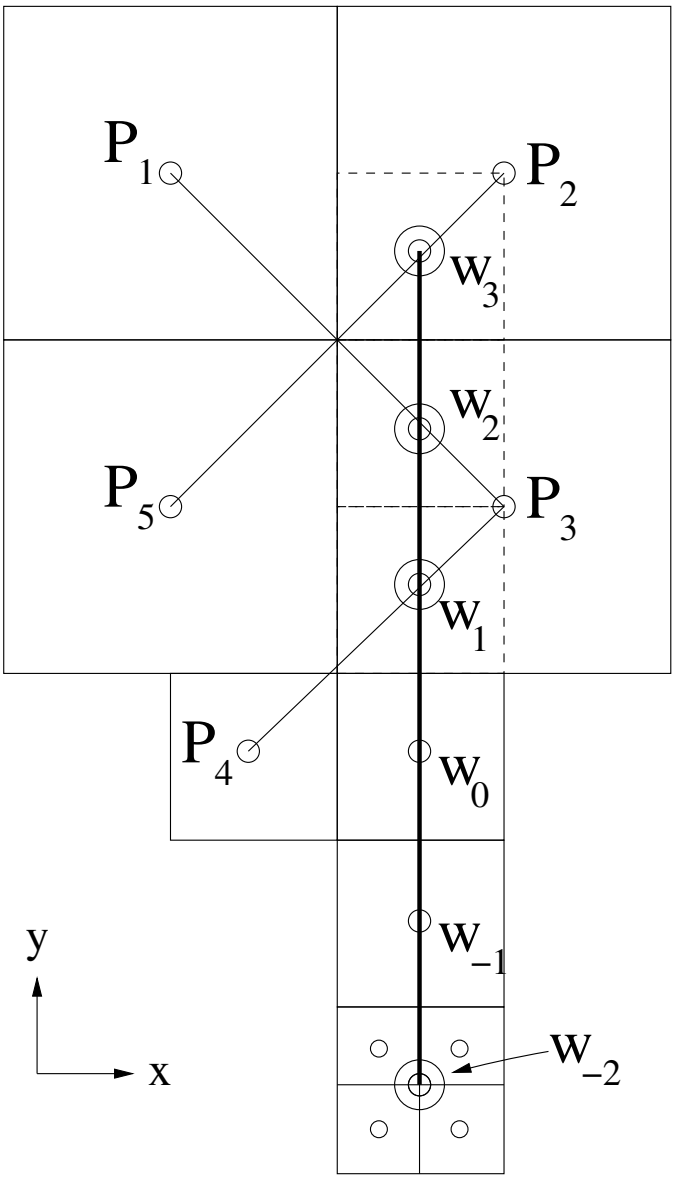


Figure 3: Stencil $\{w_k\}_{k=-2:3}$ used in calculating $D_y^+\phi(w_0)$.

7 Adaptivity

In this section we discuss some methods that can be used to determine where the mesh should be refined or coarsened. While this is an important part of any adaptive method, it is not the main focus of this paper, thus we will not go into great detail in this section.

As described in [30], [18],[6], when the problem is studied in a level set framework, there is a natural choice to refine the mesh near the particular level set of ϕ representing the interface being tracked. The details of this adaptive procedure are well documented in the above references and will not be further commented upon here.

For general problems, such as conservation laws, all values of ϕ are important, and a typical adaptive method is to attempt to determine a point-wise error estimate and then refine where the error is large and coarsen where the error is small. These error estimates can be problem dependent, such as assuming large errors near discontinuities and shocks [10, 15, 2, 7], or can be more general in nature, based on an estimation of truncation errors [3, 4].

For the examples presented here for conservation laws and Hamilton-Jacobi equations we have implemented an estimate of truncation errors found by comparing the numerical solution obtained using a low order discretization in space and time, such as two-point upwinding and forward-Euler time-stepping, with the solution obtained using a high order discretization, such as fifth order WENO and third order Runge-Kutta [22]. This method tends to pick out the shocks and contact discontinuities for refinement, as well as other areas.

Given a starting solution, $\phi(x, t = t_0)$, the method is implemented by evolving the solution for a period of time using a low order method to get $\phi_L(t = t_0+\Delta t)$, and a high order method to get $\phi_H(t = t_0 + \Delta t)$. Then we compute $E(x) = |\phi_H(x) - \phi_L(x)|$. We adopt an adaptive mesh strategy proposed in [22] to our setting. Given a problem and/or mesh dependent $\delta_R > \delta_C > 0$, if $E(x) > \delta_R$ then the leaf containing x is marked for refinement, while if $E(x) < \delta_C$ then the leaf containing x is marked for coarsening, and if a parent has all children marked for coarsening, then we coarsen it. In practice the more accurate solution

is the one used for the evolution of ϕ.

Because the evolving step in time is based on the Runge-Kutta time stepping on a virtual uniform mesh, the time step size is restricted by the finest mesh cell of the virtual uniform mesh at a particular time; for example, if there is a grid cell with $dx = 1/1024$ then even the big cells of size $dx = 1/256$ still evolve with a CFL condition determined by the finest cell of $dx = 1/1024$.

8 Numerical Examples

8.1 Linear advection equations

Example 1. We solve the model equation

$$u_t + u_x = 0, \quad 0 \le x < 1 \tag{4}$$

$$u(x,0) = u^0(x), \tag{5}$$

where u^0 is periodic with period 1,

$$u^0(x) = \begin{cases} 1, & 0.25 \le x \le 0.75, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Figure 4 shows the computational results by uniform meshes and adaptive meshes. For the uniform mesh, the solutions are computed with 129 and 513 uniformly distributed grid points, respectively. In terms of adaptive methods, we have used interpolation polynomials of degree 1 and 5, respectively, so that the solutions are computed over non-uniform meshes with the largest mesh size $dx = 1/64$ and the finest mesh size $dx = 1/512$, consisting of 122 grid points in total. The comparison results demonstrate that given a finest mesh size, the adaptive tree method with far fewer number of mesh points achieves the same accuracy as does the uniform method with the finest mesh size.

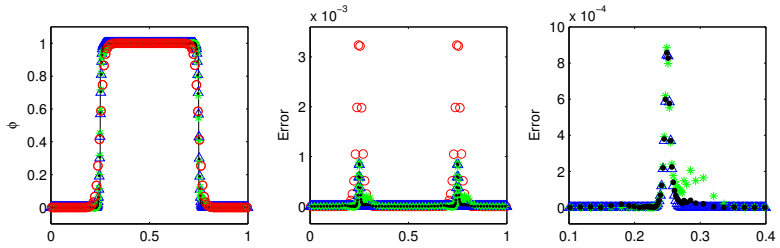


Figure 4: Transport problem in 1d after 2 periods. Red circle: uniform grid, $dx = 1/128$. Blue triangle: uniform grid, $dx = 1/512$. Green star: 122 point adaptive grid, interpolation degree $= 1$, largest $dx = 1/64$, finest $dx = 1/512$. Black dot: 122 point adaptive grid, interpolation degree $= 5$, largest $dx = 1/64$, finest $dx = 1/512$. Left: solution at $t = 2$. Center: error. Right: zoom-in on error of most accurate solutions.

8.2 Hamilton-Jacobi equations

Example 2. The Burgers equation We solve

$$u_t + H(u_x, u_y) = 0, \quad 0 \le x, y \le 1 \tag{7}$$

$$u(x, y, 0) = -0.25 \cos(2\pi(x + y)), \tag{8}$$

with a convex $H = H(p, q) = 0.5(p + q + 1)^2$.

We solve the equation by using both adaptive meshes and uniform meshes up to the final time T=0.15625. The Hamilton-Jacobi solver is based on Lax-Friedrichs monotone numerical Hamiltonians [20]. We test the adaptive tree method on both two-level and three-level adaptive meshes, with the WENO fifth order in space and the Runge-Kutta third order in time as the driving scheme and with the two-point upwinding and forward Euler time stepping as the prediction scheme. To calibrate our adaptive computation, we use the results computed on uniform meshes.

Figure 5 shows the computational results on two-level adaptive meshes of $dx = 1/32$ and $dx = 1/64$, respectively; the results shown are at the final time T=0.15625 after 41 time steps. As illustrated by the computational mesh, the adaptive mesh method uses more mesh points around the region where the solution changes rapidly and fewer mesh points around the region where the solution changes slowly.

Figure 6 shows the computational results on three-level adaptive meshes of $dx = 1/32$, $dx = 1/64$ and $dx = 1/128$, respectively; the results shown are at the final time T=0.15625 after 41 time steps. As illustrated by the computational mesh, the adaptive mesh method uses more mesh points around the region where the solution changes rapidly and fewer mesh points around the region where the solution changes slowly.

Example 3. A nonconvex equation We solve

$$u_t + H(u_x, u_y) = 0, \quad 0 \le x, y \le 1 \tag{9}$$

$$u(x, y, 0) = -0.25 \cos(2\pi(x + y)), \tag{10}$$

with a nonconvex $H = H(p, q) = -\cos(p + q + 1)$.

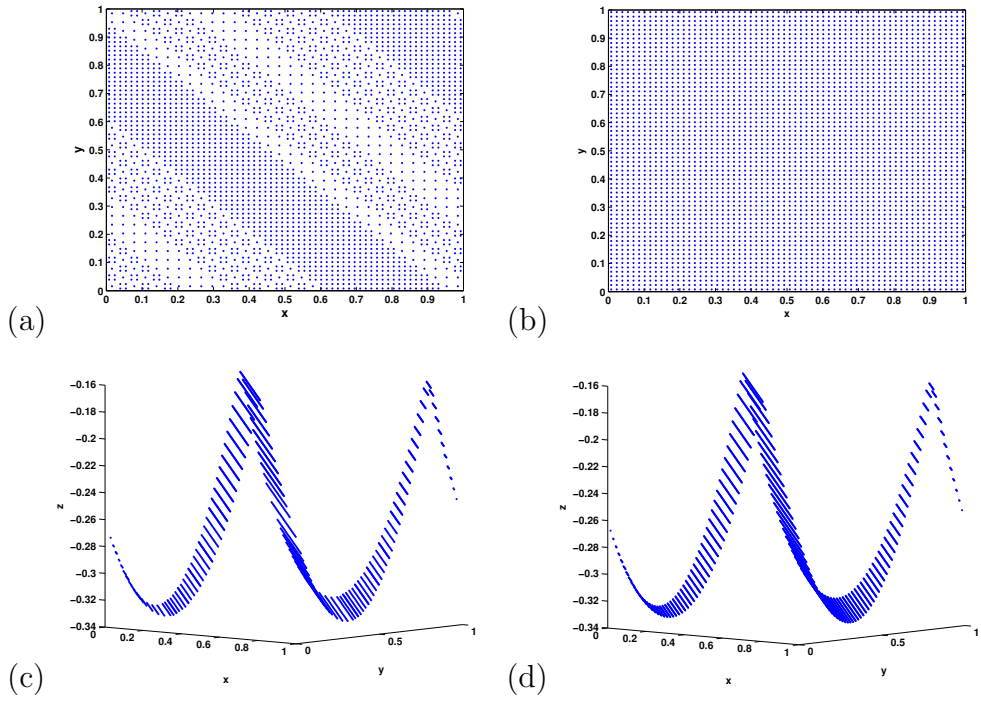


Figure 5: A 2-D Burgers equation. (a) The adaptive computational mesh has 2 levels: dx=1/32 and dx=1/64. (b) The uniform computational mesh has dx=1/64. (c) The solution by WENO-5 with RK-3 based on the adaptive mesh illustrated in (a). (d) The solution by WENO-5 with RK-3 based on the uniform mesh illustrated in (b).

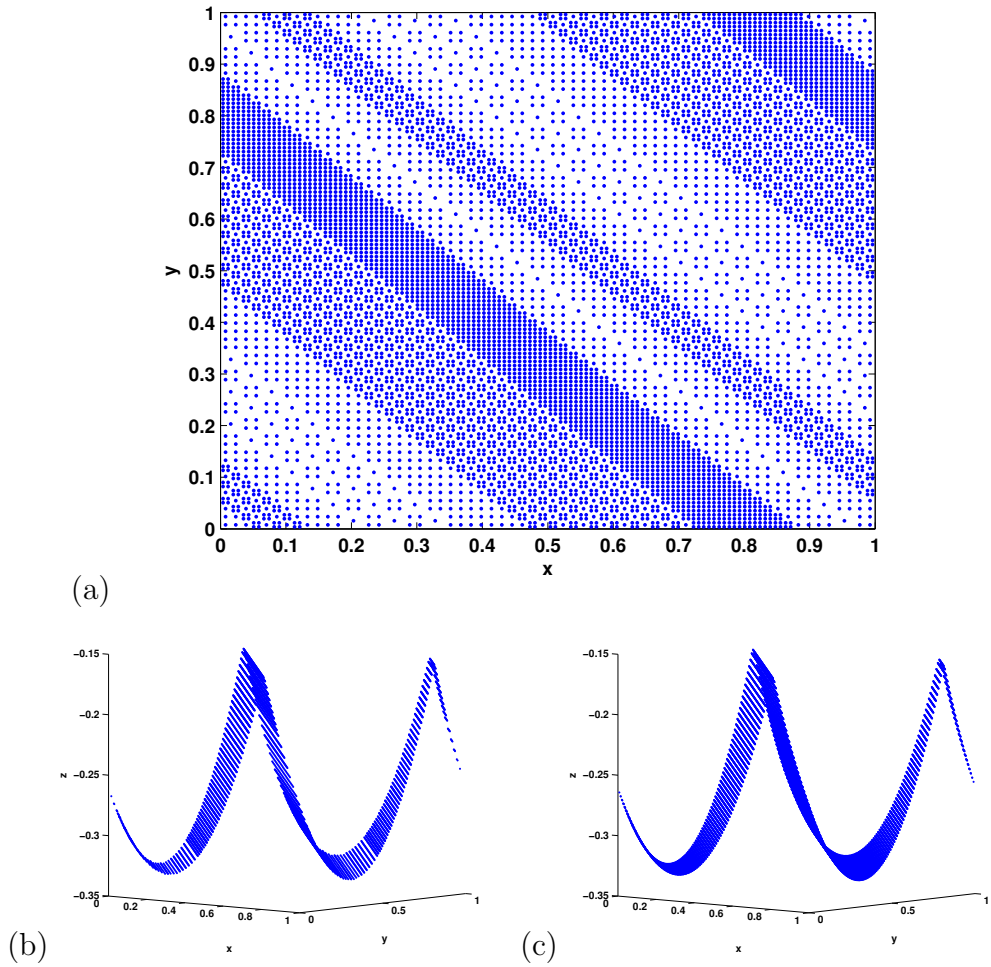


Figure 6: The 2-D Burgers equation. (a) The adaptive mesh has three levels: dx=1/32, dx=1/64, and dx=1/128. b) The solution by WENO-5 with RK-3 based on the adaptive mesh illustrated in (a). (c) The solution by WENO-5 with RK-3 based on the uniform mesh of dx=1/128.

We solve this non-convex equation by using both adaptive meshes and uniform meshes up to the final time T=0.3125. The Hamilton-Jacobi solver is based on Lax-Friedrichs monotone numerical Hamiltonians [20]. We test the adaptive tree method on both two-level and three-level adaptive meshes, with the WENO fifth order in space and the Runge-Kutta third order in time as the driving scheme and with the two-point upwinding and forward Euler time stepping as the prediction scheme. To calibrate our adaptive computation, we use the results computed on uniform meshes.

Figure 7 shows the computational results on two-level adaptive meshes of $dx = 1/32$ and $dx = 1/64$, respectively; the results shown are at the final time T=0.3125 after 41 time steps. As illustrated by the computational mesh, the adaptive mesh method uses more mesh points around the region where the solution changes rapidly and fewer mesh points around the region where the solution changes slowly.

Figure 8 shows the computational results on three-level adaptive meshes of $dx = 1/32$, $dx = 1/64$ and $dx = 1/128$, respectively; the results shown are at the final time T=0.3125 after 41 time steps. As illustrated by the computational mesh, the adaptive mesh method uses more mesh points around the region where the solution changes rapidly and fewer mesh points around the region where the solution changes slowly.

8.3 Hyperbolic conservation laws

Example 4. Sod's shock tube problem We consider the Riemann problem for the Euler equations of gas dynamics for a polytropic gas [29],

$$\mathbf{u}_t + f(\mathbf{u})_x = 0, \tag{11}$$

$$\mathbf{u}(x, 0) = \mathbf{u}^0(x), \tag{12}$$

where $\mathbf{u} = (\rho, \rho q, E)^T$, $f(\mathbf{u}) = q\mathbf{u} + (0, P, qP)^T$, $P = (\gamma - 1)(E - \frac{1}{2}\rho q^2)$, and $\gamma = 1.4$. Here ρ is density, E is the total energy, P is the pressure, and q is the velocity.

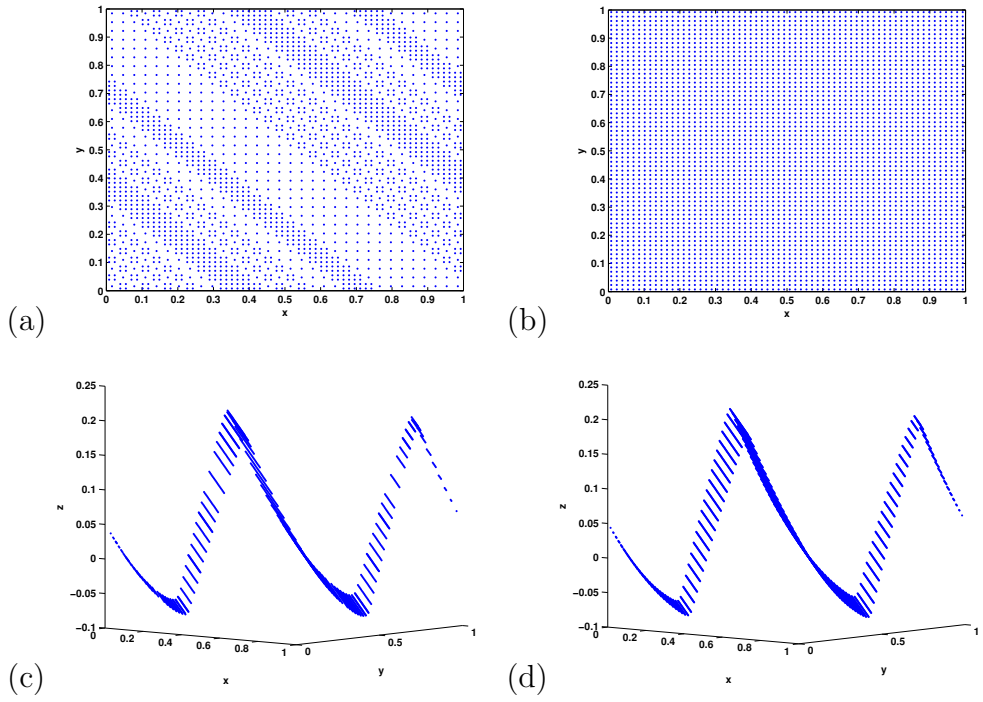
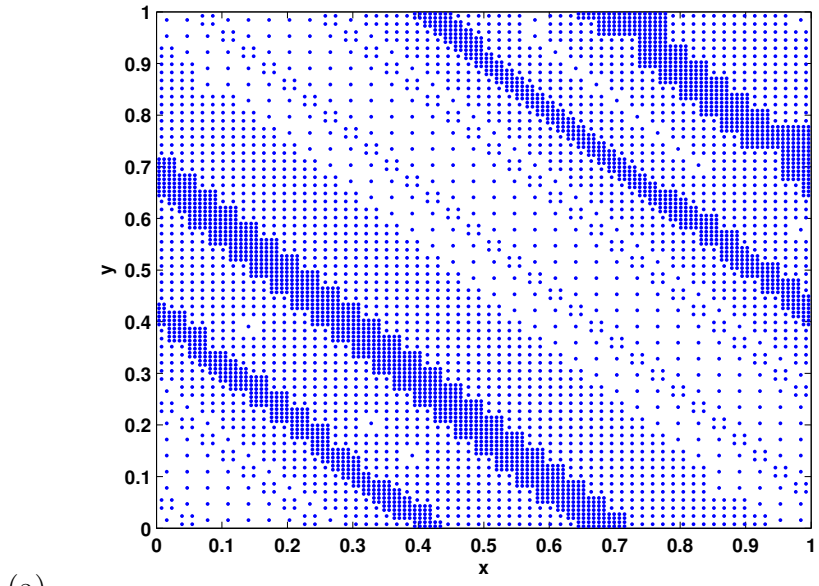
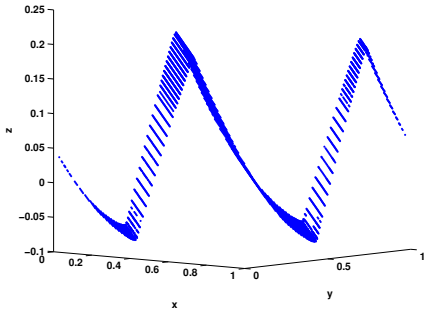


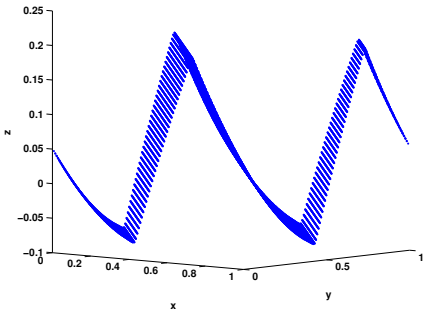
Figure 7: The 2-D non-convex equation. (a) The adaptive computational mesh has 2 levels: dx=1/32 and dx=1/64. (b) The uniform computational mesh has dx=1/64. (c) The solution by WENO-5 with RK-3 based on the adaptive mesh illustrated in (a). (d) The solution by WENO-5 with RK-3 based on the uniform mesh illustrated in (b).



(a)



(b)



(c)

Figure 8: The 2-D non-convex equation. (a) The adaptive mesh has three levels: dx=1/32, dx=1/64, and dx=1/128. b) The solution by WENO-5 with RK-3 based on the adaptive mesh illustrated in (a). (c) The solution by WENO-5 with RK-3 based on the uniform mesh of dx=1/128.

The initial condition is the Riemann data,

$$\mathbf{u}(x,0) = \begin{cases} \mathbf{u}_L, & x < 0.0, \\ \mathbf{u}_R, & \text{otherwise.} \end{cases} \tag{13}$$

Sod's initial condition is given as [29],

$$(\rho_L, q_L, P_L) = (1, 0, 1); \quad (\rho_R, q_R, P_R) = (0.125, 0, 0.10). \tag{14}$$

For expected solution behavior of the shock tube problem, we refer to [16].

We use the Roe-entropy fix numerical flux in the computation [29]. To do the adaptive computation we use the first-order upwind difference and the forward Euler scheme to get a lower order solution and compare with WENO fifth-order spatial discretization and Runge-Kutta third-order time stepping to see where the differences are the largest, and we refine those regions accordingly [22].

The results are shown in Figure 9, where we show the density plot only. We can see that the adaptive mesh method with 252 points has been able to resolve the details around the density discontinues as well as does the finest uniform mesh with 2048 points.

Example 5. Rayleigh-Taylor instability We consider the two dimensional Euler equations of compressible gas dynamics [26],

$$\mathbf{u}_t + f(\mathbf{u})_x + g(\mathbf{u})_y = 0, \tag{15}$$

$$\mathbf{u}(x,y,0) = \mathbf{u}^0(x,y), \tag{16}$$

where

$$\mathbf{u} = (\rho, \rho\bar{u}, \rho\bar{v}, E)^T,$$

$$f(\mathbf{u}) = (\rho\bar{u}, \rho\bar{u}^2 + P, \rho\bar{u}\bar{v}, \bar{u}(E+P))^T,$$

$$g(\mathbf{u}) = (\rho\bar{v}, \rho\bar{u}\bar{v}, \rho\bar{v}^2 + P, \bar{v}(E+P))^T.$$

Here ρ is density, $(\bar{u}, \bar{v})$ is the velocity, E is the total energy, P is the pressure, related to the total energy by $P = (\gamma - 1)(E - \frac{1}{2}\rho(\bar{u}^2 + \bar{v}^2))$, and γ is the ratio of specific heats.

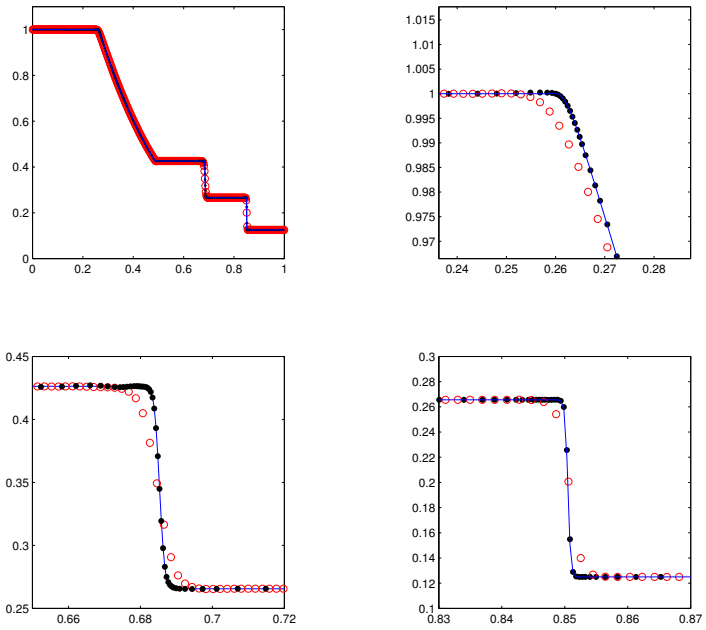


Figure 9: Sod shock tube problem density plot. Blue line: 2048 point uniform grid solution. Red circles: 512 point uniform grid solution. Black dots: 252 point adaptive grid solution with largest $dx = 1/128$, finest $dx = 1/2048$. Top left: Shock tube density at $t = 0.2$. Top right: zoom in near left end of rarefaction. Bottom left: zoom in near contact discontinuity. Bottom right: zoom in near shock.

Rayleigh-Taylor instability happens on an interface between fluids with different densities when an acceleration is directed from the heavy fluid to the light fluid. The instability has a fingering nature, with bubbles of light fluid rising into the ambient heavy fluid and spikes of heavy fluid falling into the light fluid; see [9]. See [23] for an adaptive method based on discontinuous Galerkin formulation.

The problem is set up as follows [26]. The computational domain is $[0, \frac{1}{4}] \times [0, 1]$; initially the interface is at $y = \frac{1}{2}$, the heavy fluid with density $\rho = 2$ is below the interface, and the light fluid with density $\rho = 1$ is above the interface with the acceleration in the positive $y-$direction; the pressure p is continuous across the interface; a small perturbation is given to the $y-$direction fluid speed; thus for $0 \leq y < \frac{1}{2}$, $\rho = 2$, $\bar{u} = 0$, $P = 2y + 1$, $\bar{v} = -0.025c \cdot \cos(8\pi x)$, and for $\frac{1}{2} \leq y < 1$, $\rho = 1$, $\bar{u} = 0$, $P = y + \frac{3}{2}$, $\bar{v} = -0.025c \cdot \cos(8\pi x)$, where c is the sound speed, $c = \sqrt{\frac{\gamma P}{\rho}}$, and the ratio of specific heats $\gamma = 1.4$; reflective boundary conditions are imposed for the left and right boundaries; at the top boundary the flow values are set as $\rho = 1$, $P = 2.5$, $\bar{u} = \bar{v} = 0$, and at the bottom boundary they are $\rho = 2$, $P = 1$, $\bar{u} = \bar{v} = 0$; the source term ρ is added to the right hand side of third equation and $\rho\bar{v}$ is added to the fourth equation of Euler equations. The final simulation time is $t = 1.95$.

We use the Roe-entropy fix numerical flux in the computation [29]. To do the adaptive computation we use the first-order upwind difference and the forward Euler scheme to get a lower order solution and compare with WENO fifth-order spatial discretization and Runge-Kutta third-order time stepping to see where the differences are the largest, and we refine those regions accordingly [22].

The computational results are shown in Figure 10, where we have compared the adaptive solution with the uniform solution. As we can see, by using only the half of the number of mesh points as required on a uniform mesh, the adaptive solution has similar resolution as the one on a uniform mesh.

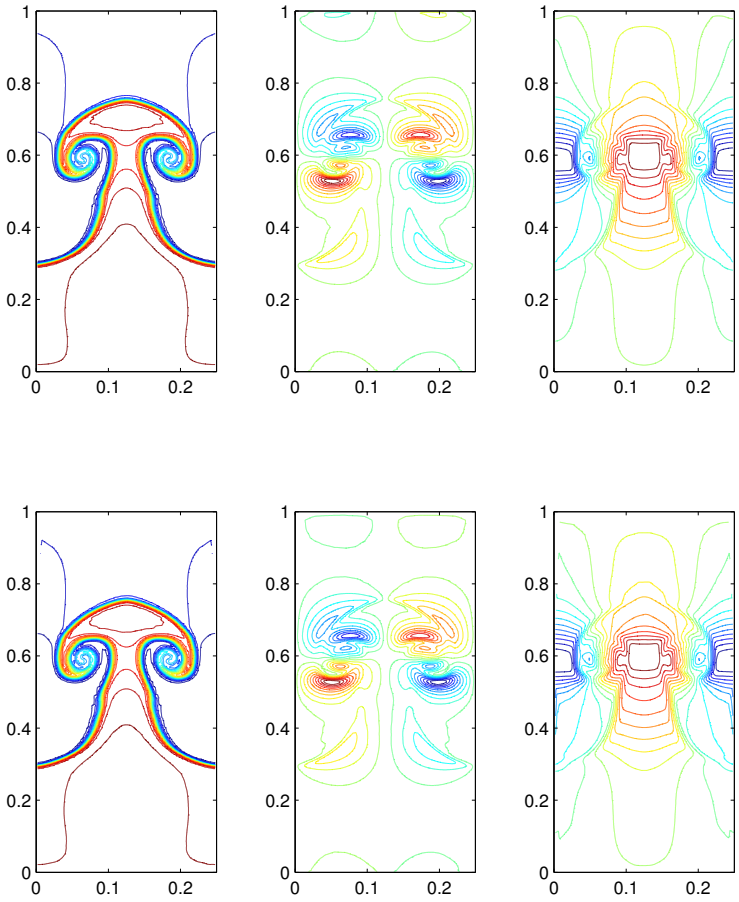


Figure 10: Contour plots for Rayleigh-Taylor instability problem for 2-D Euler equations. Left to right: density, $\bar{u}$, and $\bar{v}$. Top row: uniform grid, $dx = \frac{1}{256}$, 16,384 points. Bottom row: adaptive grid, largest $dx = \frac{1}{64}$, finest $dx = \frac{1}{256}$, 8,788 points.

9 Conclusion

We developed high order ENO schemes on non-uniform meshes based on generalized binary trees. The idea is to adopt an appropriate data structure which allows to communicate information easily between unstructured data structure and virtual uniform meshes. While the generalized binary trees as an unstructured data structure can store solution information efficiently if combined with a good adaptive strategy, virtual uniform meshes allow us to take advantage of many well-developed ENO numerical methods based on uniform meshes. Therefore, the ENO adaptive tree methods proposed here can leverage the merits from both tree structures and uniform meshes. Numerical examples demonstrate that the new method is efficient and accurate.

Acknowledgments

The authors were partially supported by an ONR MURI grant N00014-02-1-0720. The third author was partially supported by NSF DMS-0542174.

References

[1] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation. *J. Comput. Phys.*, 114:45–58, 1994.

[2] S. Albert, B. Cockburn, D. French, and T. Peterson. A posteriori error estimates for general numerical methods for Hamilton-Jacobi equations. part I: The steady state case. *Math. Comp.*, 71:49–76, 2002.

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

[4] M.J. Berger and R.J. LeVeque. Adaptive mesh refinement using wave propagation algorithms for hyperbolic systems. *SIAM J. Numer. Analy.*, 35:2298–2316, 1998.

[5] J. Casper and H. L. Atkins. A finite-volume high-order eno scheme for two-dimensional hyperbolic systems. *J. Comput. Phys.*, 106:62–76, 1993.

[6] Thomas Cecil, Stanley Osher, and Jianliang Qian. Simplex free adaptive tree fast sweeping and evolution methods for solving level set equations in arbitrary dimension. *J. Comput. Phys.*, 213(2):458–473, 2006.

[7] B. Cockburn and B. Yenikaya. An adaptive method with rigorous error control for the Hamilton-Jacobi equations. part I: the one dimensional steady state case. *Appl. Numer. Math.*, 52:175–195, 2005.

[8] Sarah F. Frisken and Ronald N. Perry. Simple and efficient traversal methods for quadtrees and octrees. *Journal of Graphics Tools*, 7(3):1–11, 2002.

[9] J. Glimm, J. Grove, X. Li, W. Oh, and D.C. Tan. The dynamics of bubble growth for Rayleigh-Taylor unstable interfaces. *Physics of Fluids*, 31:174–201, 1988.

[10] L. Gosse and C. Makridakis. Two a posteriori error estimates for one-dimensional scalar conservation laws. *SIAM J. Numer. Analy.*, 38:964–988, 2000.

[11] A. Harten, B. Engquist, S. J. Osher, and S. Chakravarthy. Uniformly high order essentially non-oscillatory schemes, III. *J. Comput. Phys.*, 71:231–303, 1987.

[12] G. S. Jiang and D. Peng. Weighted ENO schemes for Hamilton-Jacobi equations. *SIAM J. Sci. Comput.*, 21:2126–2143, 2000.

[13] G. S. Jiang and C. W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126:202–228, 1996.

[14] Donald E. Knuth. *The art of computer programming, volume 1 (3rd ed.): fundamental algorithms*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1997.

[15] D. Kroner and M. Ohlberger. A posteriori error estimates for upwind finite volume schemes for nonlinear conservation laws in multidimensions. *Math. Comp.*, 69:25–39, 2000.

[16] R. J. LeVeque. *Numerical methods for conservation laws*. Birkhauser-Verlag, 1990.

[17] B. Merriman. Understanding the Shu-Osher conservative finite difference form. *J. Sci. Comput.*, 19:309–322, 2003.

[18] Chohong Min. Local level set method in high dimension and codimension. *J. Comput. Phys.*, 200(1):368–382, 2004.

[19] Doug Moore. The cost of balancing generalized quadtrees. In *SMA '95: Proceedings of the third ACM symposium on Solid modeling and applications*, pages 305–312, New York, NY, USA, 1995. ACM Press.

[20] S. J. Osher and C. W. Shu. High-order Essentially NonOscillatory schemes for Hamilton-Jacobi equations. *SIAM J. Numer. Analy.*, 28:907–922, 1991.

[21] Stanley Osher and Ronald P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Oxford Univ. Press, 2002.

[22] J. Qian and W. W. Symes. Adaptive finite difference method for traveltime and amplitude. *Geophysics*, 67:167–176, 2002.

[23] J.-F. Remacle, J.E. Flaherty, and M. S. Shephard. An adaptive discontinuous Galerkin technique with an orthogonal basis applied to compressible flow problems. *SIAM Review*, 45:53–72, 2003.

[24] Hanan Samet. *Applications of spatial data structures: Computer graphics, image processing, and GIS*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[25] S. Serna and J. Qian. Fifth order weighted power-eno schemes for Hamilton-Jacobi equations. *J. Sci. Comp.*, 29:57–81, 2006.

[26] J. Shi, Y.-T. Zhang, and C.-W. Shu. Resolution of high order WENO schemes for complicated flow structures. *J. Comput. Phys.*, 186:690–696, 2003.

[27] C. W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In B. Cockburn, C. Johnson, C.W. Shu, and E. Tadmor, editors, *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, volume 1697, pages 325–432. Springer, 1998. Lecture Notes in Mathematics.

[28] C. W. Shu and S. J. Osher. Efficient implementation of essentially non-oscillatory shock capturing schemes. *J. Comput. Phys.*, 77:439–471, 1988.

[29] C. W. Shu and S. J. Osher. Efficient implementation of essentially non-oscillatory shock capturing schemes II. *J. Comput. Phys.*, 83:32–78, 1989.

[30] John Strain. Tree methods for moving interfaces. *J. Comput. Phys.*, 151(2):616–648, 1999.