Journal of Scientific Computing, Vol. 19, Nos. 1–3, December 2003 (© 2003)

Multidomain WENO Finite Difference Method with Interpolation at Subdomain Interfaces

Kurt Sebastian[1, 2] **and Chi-Wang Shu**[1, 3]

Received July 31, 2002; accepted (in revised form) November 2, 2002

High order finite difference WENO methods have the advantage of simpler coding and smaller computational cost for multi-dimensional problems, compared with finite volume WENO methods of the same order of accuracy. However a main restriction is that conservative finite difference methods of third and higher order of accuracy can only be used on uniform rectangular or smooth curvilinear meshes. In order to overcome this difficulty, in this paper we develop a multidomain high order WENO finite difference method which uses an interpolation procedure at the subdomain interfaces. A simple Lagrange interpolation procedure is implemented and compared to a WENO interpolation procedure. Extensive numerical examples are shown to indicate the effectiveness of each procedure, including the measurement of conservation errors, orders of accuracy, essentially non-oscillatory properties at the domain interfaces, and robustness for problems containing strong shocks and complex geometry. Our numerical experiments have shown that the simple and efficient Lagrange interpolation suffices for the subdomain interface treatment in the multidomain WENO finite difference method, to retain essential conservation, full high order of accuracy, essentially non-oscillatory properties at the domain interfaces even for strong shocks, and robustness for problems containing strong shocks and complex geometry. The method developed in this paper can be used to solve problems in relatively complex geometry at a much smaller CPU cost than the finite volume version of the same method for the same accuracy. The method can also be used for high order finite difference ENO schemes and an example is given to demonstrate a similar result as that for the WENO schemes.

KEY WORDS: WENO finite difference method; multidomain; subdomain interfaces; Lagrange interpolation; WENO interpolation.

1. INTRODUCTION

In this paper we are concerned with solving a two dimensional hyperbolic conservation law

$$u_t + f(u)_x + g(u)_y = 0 \tag{1.1}$$

Dedicated to Professor Stanley Osher on the occasion of his 60th birthday.
[1] Division of Applied Mathematics, Brown University, Providence, Rhode Island 02912.
[2] E-mail: kseb@cfm.brown.edu
[3] To whom correspondence should be addressed. E-mail: shu@cfm.brown.edu

using a high order finite difference weighted essentially non-oscillatory (WENO) method, such as the third order version in [17], the fifth order version in [11] and the higher order versions in [1], in a multidomain setting. Most of the presentations and numerical examples will be concentrated on the fifth order version in [11], but the application to versions of other orders of accuracy is straightforward. We only consider one and two spatial dimensions in this paper. Generalizations to three spatial dimensions are conceptually straightforward but will involve more complicated coding, and will be studied in a future work.

WENO schemes are designed based on the successful essentially non-oscillatory (ENO) schemes in [8, 24, 25]. Finite volume WENO schemes have been constructed in [17] for a third order version in one space dimension, in [5] and [9] for second, third and fourth order versions for 2D general triangulations, and in [20] for high order versions containing negative linear weights. Finite difference WENO schemes have been constructed in [11] for the third and fifth order versions in multi space dimensions with a general framework for the design of the smoothness indicators and nonlinear weights, and in [1] for very high order (between 7 and 11) versions.

Both ENO and WENO use the idea of adaptive stencils in the reconstruction procedure based on the local smoothness of the numerical solution to automatically achieve high order accuracy and non-oscillatory property near discontinuities. ENO uses just one (optimal in some sense) out of many candidate stencils when doing the reconstruction; while WENO uses a convex combination of all the candidate stencils, each being assigned a nonlinear weight which depends on the local smoothness of the numerical solution based on that stencil. WENO improves upon ENO in robustness, better smoothness of fluxes, better steady state convergence, better provable convergence properties, and more efficiency. For more details of ENO and WENO schemes, we refer to the lecture notes [21, 22].

WENO schemes have been widely used in applications. Some of the examples include dynamical response of a stellar atmosphere to pressure perturbations [4]; shock vortex interactions and other gas dynamics problems [6, 7]; incompressible flow problems [30]; Hamilton–Jacobi equations [10, 31]; magneto-hydrodynamics [12]; underwater blast-wave focusing [14]; the composite schemes and shallow water equations [15, 16], real gas computations [18], wave propagation using Fey's method of transport [19]; etc.

There are two types of WENO schemes for solving (1.1), namely the finite volume schemes and the finite difference schemes.

The finite volume scheme for (1.1) approximates an integral version of it. We will use the one dimensional case to illustrate the ideas. Thus, the computational domain (interval) is partitioned into a collection of cells $I_i = (x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$, not necessarily uniform or smoothly changing, and the one dimensional version of (1.1) is integrated over I_i to obtain

$$\frac{d\bar{u}(x_i, t)}{dt} = -\frac{1}{\Delta x_i}\left(f(u(x_{i+\frac{1}{2}}, t) - f(u(x_{i-\frac{1}{2}}, t))\right), \tag{1.2}$$

where

$$\bar{u}(x_i, t) \equiv \frac{1}{\Delta x_i} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(\xi, t)\, d\xi \tag{1.3}$$

is the cell average. (1.2) is then approximated by the following conservative scheme

$$\frac{d\bar{u}_i(t)}{dt} = -\frac{1}{\Delta x_i} (\hat{f}_{i+\frac{1}{2}} - \hat{f}_{i-\frac{1}{2}}), \tag{1.4}$$

where $\bar{u}_i(t)$ is the numerical approximation to the cell average $\bar{u}(x_i, t)$, and the numerical flux $\hat{f}_{i+\frac{1}{2}}$ is defined by

$$\hat{f}_{i+\frac{1}{2}} = h(u^-_{i+\frac{1}{2}}, u^+_{i+\frac{1}{2}}) \tag{1.5}$$

with h being an exact or approximate Riemann solver (see, e.g., [13, 21] for details), and the values $u^{\pm}_{i+\frac{1}{2}}$ are obtained by a WENO reconstruction procedure, i.e., accurate point values are reconstructed from the cell averages in a high order and essentially non-oscillatory fashion, see [21] for details.

On the other hand, the finite difference schemes approximate directly (1.1). We will still use the one dimensional case to illustrate the ideas. Thus, the computational domain (interval) is partitioned into a collection of grid points x_i, which *must* be either uniform or smoothly varying, for third and higher order approximations. The scheme approximates the derivative in a conservative fashion:

$$\frac{du_i(t)}{dt} = -\frac{1}{\Delta x} (\hat{f}_{i+\frac{1}{2}} - \hat{f}_{i-\frac{1}{2}}) \tag{1.6}$$

where $u_i(t)$ is the numerical approximation to the point value $u(x_i, t)$, and the numerical flux

$$\hat{f}_{i+\frac{1}{2}} = \hat{f}(u_{i-r},..., u_{i+s})$$

has to satisfy several conditions, such as Lipschitz continuity, consistency with the physical flux f, etc. To get these numerical fluxes from the point values $f(u_i)$, we would need to use the same WENO reconstruction procedure as in the finite volume schemes. For details, see [25, 11, 21].

In one spatial dimension, the finite volume schemes and the finite difference schemes described above are equivalent, both in numerical resolution and accuracy and in complexity of coding and CPU timing. However, for multi-spatial dimensions they are no longer equivalent. While finite difference schemes are still very simple to code and fast to compute (essentially only one or more outside "do loops" are needed to change a one dimensional finite difference code to multi-dimensions), the finite volume code becomes much more complicated and costly. For the same orders (higher than two) of accuracy and same resolution for discontinuities, the finite volume WENO scheme is about four times more expensive than the finite difference WENO scheme in 2D, and is about nine times more expensive in 3D, see, e.g., [3, 23]. However, finite volume schemes have the flexibility in handling almost arbitrary meshes, while conservative finite difference schemes can only handle uniform or smooth curvilinear meshes.

A compromise to use the simple and efficient finite difference methods yet still achieving some flexibility in geometry and meshes, is to adopt a multidomain framework, where in each domain the finite difference scheme is used on a uniform or smooth curvilinear mesh, and at domain interfaces a high order interpolation procedure is performed. This approach has been used before, mainly for lower order schemes, see, e.g., [2, 26, 27]. In this paper, we explore this approach for the high order finite difference WENO schemes. We use the fifth order WENO scheme in [11] as the base algorithm, although higher order versions in [1] could also be used in the same framework. We use a simple Lagrange interpolation procedure at the subdomain interfaces and compare its effectiveness to a WENO interpolation procedure based upon the WENO finite difference method.

The multidomain framework investigated in this paper can serve multiple purposes. It is often necessary to divide a larger domain into several smaller subdomains to allow for the computation of complex geometries, e.g., realistic aircraft configurations. It is also necessary, at times to divide a larger domain into subdomains because of the limitations of computer speed and storage. For example, part of a large domain might require the computation of the unsteady 3D Navier Stokes equations; whereas in other regions the Euler equations might suffice. Lastly, different numerical schemes may be required or desired for different flow structures in different subdomains.

Interpolation in any manner is nonconservative if it is not based on cell averages. Thus, although the numerical schemes in the subdomains are conservative, the global schemes used in this paper are not conservative since the interfaces are not treated conservatively. A conservative interface treatment would require the usage of cell averages and hence finite volume base schemes, thus defeating the purpose of trying to save computational and storage costs via using high order finite difference schemes. We will show in this paper, through extensive numerical examples that using either a "linear" type interpolation such as Lagrange or a "nonlinear" WENO type interpolation does not produce $O(1)$ conservation error, even in the difficult situations such as strong shocks passing through multiple domain interfaces or slow moving shocks passing through the interfaces [26]. We show, in fact, that the conservation error for either method is second order for smooth flows and first order for shocked flows, and the full high order of accuracy and essentially non-oscillatory results are achieved for the final multidomain scheme.

We remark that Tang and Zhou [28] investigated such questions as how large the conservation error due to a nonconservative interface algorithm may be, whether a convergent numerical approximation obtained with it is a weak solution to the PDE, and if such a treatment can provide correct jumps and locations of the discontinuities. They subsequently proved that indeed a nonconservative interface algorithm could answer all of these questions in the affirmative under suitable assumptions. Although the situation treated in [28] is lower order in accuracy, it does provide some theoretical assurance that such nonconservative interpolation based interface treatment yields convergent results for weak solutions. We make a brief remark in Sec. 2 to indicate that a similar theoretical result of conservation errors going to zero with mesh refinement can be obtained under similar assumptions as in [28]. Unlike the cases in [28, 26], our approach in this paper allows for accuracy to any high order desired. When used in conjunction with the WENO finite difference formulation, both Lagrange interpolation and WENO interpolation provide equally suitable results in almost all of the standard test cases.

In this paper, when there is only slight or no improvement with the WENO interpolation algorithm over the Lagrange Interpolation algorithm, we will show only the Lagrange interpolation results. When the WENO interpolation algorithm provides significantly better results, we will exhibit both. The reader should find it very interesting that in almost all cases, the Lagrange interpolation provides comparable results to the WENO interpolation. This is of course strongly related to the good base algorithm used, namely the characteristic based fifth order finite difference WENO in [11]. In fact, we have looked at the ghost point values generated by the Lagrange interpolation and they are indeed oscillatory occasionally, but the base WENO algorithm generates non-oscillatory results out of them. This is very good news to the users that the simple, cost effective Lagrange Interpolation can be used in conjunction with high order finite difference WENO schemes in a multidomain setting to obtain satisfactory results.

In some applications, the finite difference ENO schemes might be preferred to the WENO schemes. The method we develop in this paper can also be used for high order finite difference ENO schemes and an example is given to demonstrate a similar result as that for the WENO schemes in Sec. 3.

The rest of the paper is organized as follows. Section 2 deals with a complete description of the WENO and Lagrange interpolation algorithms. The building block for the WENO interpolation algorithm is the Lagrange interpolation algorithm itself. We will also discuss in this section how we have chosen subdomains, including issues concerning subdomain overlap and non-rectangular domains. In Sec. 3, we will exhibit our computational results. Here we will show results in both 1D and 2D, including scalar PDEs and systems of PDEs.

2. THE WENO AND LAGRANGE INTERPOLATION ALGORITHMS

Elementary numerical analysis tells us that, given the point values of a smooth function at $k+1$ points, there is a unique polynomial of degree at most k which passes through all these data points. Thus one can interpolate a point value of the function at any point to $(k+1)$th order accuracy. One way to perform this interpolation is the Lagrange interpolation algorithm. This Lagrange interpolation also serves as the underlying building block for a WENO interpolation.

There is no need to describe the Lagrange interpolation as it is standard. We will now briefly describe the WENO interpolation. With Lagrange interpolation as its underlying building block, the WENO interpolation algorithm attempts to obtain results comparable in the order of accuracy as the Lagrange interpolation in the same stencil but is essentially non-oscillatory when the function being interpolated is not smooth in the stencil. This is achieved by writing the interpolation as a convex sum of several lower order interpolations, with the coefficients in this sum adjusted by the local smoothness of the function inside the smaller stencils. Starting

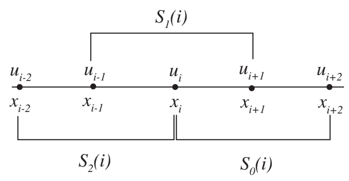

Fig. 1. Starting with a stencil of 5 points, there are 3 candidate substencils.

with a stencil of $(2k-1)$ points, we have k candidate substencils given by (see also Fig. 1):

$$S_r(i) = \{x_{i-r}, \ldots, x_{i-r+k-1}\}; \qquad r = 0, \ldots, k-1. \tag{2.1}$$

Once the k candidate substencils are known, we can easily develop the following Lagrange interpolation polynomial on each substencil r:

$$u_L^{(r)}(x) = \sum_{j=0}^{k-1} u_{i-r+j} c_{rj}(x). \tag{2.2}$$

The $c_{rj}(x)$ are constants dependent upon the location x where the polynomial is to be read. They are given as:

$$c_{rj}(x) = \prod_{\substack{l=0 \\ l \neq j}}^{k-1} \frac{x - x_{i-r+l}}{x_{i-r+j} - x_{i-r+l}}. \tag{2.3}$$

The $(2k-1)$ point Lagrange interpolation polynomial can then be found in terms of the Lagrange interpolation polynomials from each substencil:

$$u_L(x) = \sum_{r=0}^{k-1} d_r(x)\, u_L^{(r)}(x) \tag{2.4}$$

where $d_r(x)$ are constants, also called linear weights, which depend on the location x. Because of consistency we always have

$$\sum_{r=0}^{k-1} d_r(x) = 1. \tag{2.5}$$

Up to now, we have only developed a Lagrange interpolation polynomial, albeit in a unique way which facilitates the development of the WENO interpolation polynomial. The results shown in Sec. 3 of this paper use this $(2k-1)$ point Lagrange interpolation polynomial to compare against the WENO interpolation algorithm. As we will see later, in almost all the test cases, the Lagrange interpolation algorithm performs as well as the WENO interpolation algorithm.

We now look for a WENO interpolation polynomial in the form:

$$u_w(x) = \sum_{r=0}^{k-1} \omega_r(x)\, u_L^{(r)}(x) \tag{2.6}$$

where $\omega_r(x)$ are the nonlinear weights depending on both the location x and the values of u being interpolated. Again for consistency

$$\sum_{r=0}^{k-1} \omega_r(x) = 1. \tag{2.7}$$

In the case that the actual function $u(x)$ is smooth in all of the candidate substencils, we would like to have

$$\omega_r(x) = d_r(x) + O(\Delta x^{k-1}); \qquad r = 0,..., k-1. \tag{2.8}$$

Using (2.6) and (2.8), it is easy to see that this would imply $(2k-1)$th order accuracy, i.e.,

$$u_w(x) = \sum_{r=0}^{k-1} \omega_r(x)\, u_L^{(r)}(x) = u(x) + O(\Delta x^{2k-1}). \tag{2.9}$$

Following the idea in [11], the nonlinear weights $\omega_r(x)$ are chosen to be

$$\omega_r(x) = \frac{\tilde{\omega}_r(x)}{\sum_{s=0}^{k-1} \tilde{\omega}_s(x)}, \qquad \tilde{\omega}_r(x) = \frac{d_r(x)}{(\epsilon + \beta_r(x))^2} \tag{2.10}$$

where ϵ is chosen to be a small constant greater than zero in order to keep the denominator from becoming zero. In this paper $\epsilon = 10^{-6}$ is used for all the test cases. The smoothness indicator $\beta_r(x)$ is determined by

$$\beta_r(x) = \sum_{l=1}^{k-1} \int_a^b \Delta x^{2l-1} \left(\frac{d^l}{dx^l} u_L^{(r)}(x) \right)^2 dx \tag{2.11}$$

where the limits of integration are chosen based upon the location of x in relation to the interpolation points x_j; $j = i-(k-1),..., i+(k-1)$.

For example, if $k = 3$, see Fig. 2 for the five point stencil. If the location of x were as shown in the figure, the smoothness indicator would be

$$\beta_r(x) = \sum_{l=1}^{k-1} \int_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} \Delta x^{2l-1} \left(\frac{d^l}{dx^l} u_L^{(r)}(x) \right)^2 dx. \tag{2.12}$$

This would produce the following smoothness indicators:

$$\begin{aligned}
\beta_0 &= (\ 4u_i^2 - 13u_i u_{i+1} + 13u_{i+1}^2 + \ 5u_i u_{i+2} - 13u_{i+1}u_{i+2} + \ 4u_{i+2}^2)/3 \\
\beta_1 &= (\ 4u_{i-1}^2 - 19u_{i-1}u_i + 25u_i^2 + 11u_{i-1}u_{i+1} - 31u_i u_{i+1} + 10u_{i+1}^2)/3 \\
\beta_2 &= (10u_{i-2}^2 - 49u_{i-2}u_{i-1} + 61u_{i-1}^2 + 29u_{i-2}u_i - 73u_{i-1}u_i + 22u_i^2\ )/3.
\end{aligned} \tag{2.13}$$

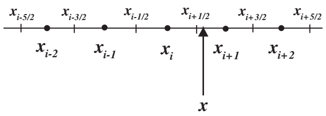

Fig. 2. Example of a $(2k-1)$ point stencil with $k = 3$. The indicated location of x is shown as a possible interpolation location.

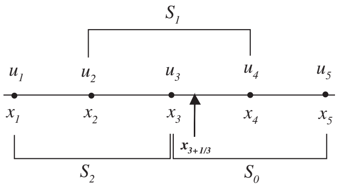

Fig. 3. The 5 point stencil required for a 5th order interpolation.

For further clarification and to exhibit that the smoothness indicators change appreciably according to the location of the interpolation, if $k=3$ and x is located in $(x_{i-1/2}, x_{i+1/2})$, i.e., a centered interpolation, then the following smoothness indicators are produced:

$$\begin{aligned}
\beta_0 &= (10u_i^2 - 31u_iu_{i+1} + 25u_{i+1}^2 + 11u_iu_{i+2} - 19u_{i+1}u_{i+2} + 4u_{i+2}^2)/3 \\
\beta_1 &= (4u_{i-1}^2 - 13u_{i-1}u_i + 13u_i^2 + 5u_{i-1}u_{i+1} - 13u_iu_{i+1} + 4u_{i+1}^2)/3 \\
\beta_2 &= (4u_{i-2}^2 - 19u_{i-2}u_{i-1} + 25u_{i-1}^2 + 11u_{i-2}u_i - 31u_{i-1}u_i + 10u_i^2)/3.
\end{aligned} \tag{2.14}$$

The WENO interpolation algorithm may appear at first glance to be somewhat complicated to implement; however, consider the five point stencil with corresponding substencils shown in Fig. 3. Each stencil yields the following three point Lagrange interpolations:

$$\begin{aligned}
u_L^{(0)}(x_{3+1/3}) &= \frac{5}{9}u_3 + \frac{5}{9}u_4 - \frac{1}{9}u_5 \\
u_L^{(1)}(x_{3+1/3}) &= -\frac{1}{9}u_2 + \frac{8}{9}u_3 + \frac{2}{9}u_4 \\
u_L^{(2)}(x_{3+1/3}) &= \frac{2}{9}u_1 - \frac{7}{9}u_2 + \frac{14}{9}u_3.
\end{aligned}$$

The location $x_{3+1/3}$ yields the following smoothness indicators:

$$\begin{aligned}
\beta_0 &= (10u_3^2 - 31u_3u_4 + 25u_4^2 + 11u_3u_5 - 19u_4u_5 + 4u_5^2)/3 \\
\beta_1 &= (4u_2^2 - 13u_2u_3 + 13u_3^2 + 5u_2u_4 - 13u_3u_4 + 4u_4^2)/3 \\
\beta_2 &= (4u_1^2 - 19u_1u_2 + 25u_2^2 + 11u_1u_3 - 31u_2u_3 + 10u_3^2)/3.
\end{aligned}$$

Thus the weights needed for a WENO interpolation can easily be computed:

$$d_0 = \frac{7}{27}, \qquad d_1 = \frac{35}{54}, \qquad d_2 = \frac{5}{54}$$

$$\tilde{\omega}_0 = \frac{d_0}{(\epsilon+\beta_0)^2}, \qquad \tilde{\omega}_1 = \frac{d_1}{(\epsilon+\beta_1)^2}, \qquad \tilde{\omega}_2 = \frac{d_2}{(\epsilon+\beta_2)^2}$$

$$\omega_0 = \frac{\tilde{\omega}_0}{\tilde{\omega}_1+\tilde{\omega}_2+\tilde{\omega}_3}, \qquad \omega_1 = \frac{\tilde{\omega}_1}{\tilde{\omega}_1+\tilde{\omega}_2+\tilde{\omega}_3}, \qquad \omega_2 = \frac{\tilde{\omega}_2}{\tilde{\omega}_1+\tilde{\omega}_2+\tilde{\omega}_3}.$$

And lastly, the WENO interpolation is then:

$$\begin{aligned} u_w(x_{3+1/3}) = \quad & \omega_0\left(\frac{5}{9}u_3 + \frac{5}{9}u_4 - \frac{1}{9}u_5 \right) \\ & + \omega_1\left(-\frac{1}{9}u_2 + \frac{8}{9}u_3 + \frac{2}{9}u_4 \right) \\ & + \omega_2\left(\frac{2}{9}u_1 - \frac{7}{9}u_2 + \frac{14}{9}u_3 \right). \end{aligned}$$

Clearly the procedure and coding are pretty straightforward.

There are several important issues to account for in the implementation of either interpolation algorithm. When the geometry and boundary conditions of a problem allow, a centered interpolation is intuitively desirable for stability purposes. When a centered interpolation is not possible due to geometry and/or boundary condition restrictions, we have numerically demonstrated that non-centered interpolations also produce satisfactory results.

To achieve a centered interpolation, it is clear that subdomain overlap is necessary. In fact, for a $(2k-1)$ point interpolation, the subdomain overlap must be $(k-1)$ points. For a clearer understanding of this requirement, see Fig. 4 for a graphic description of a 1D subdomain overlap. In a 2D problem, the requirements for overlap are the same with the obvious distinction that the requirements must be met in both directions, as our interpolation is performed first in one direction and then in the other direction. See Fig. 5 for a graphic description of a 2D subdomain overlap.

To summarize, our multidomain finite difference fifth order WENO scheme has the following main features:

1. The domain interface has a 2 point overlapping in each subdomain per direction, whenever this is allowed by the geometry and boundary conditions. For example, in the 1D case shown in Fig. 4, if we denote by x_j^1 the grid points in Subdomain 1 with x_N^1 being the interface node, and by x_j^2 the grid points in Subdomain 2 with x_1^2 being the interface node, then we will evolve in time Subdomain 1 grids up to x_{N+2}^1 and evolve in time Subdomain 2 grids from x_{-1}^2.

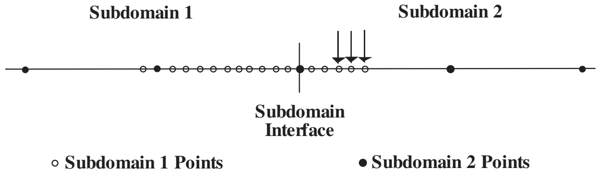

Fig. 4. To demonstrate the requirements of subdomain overlap, consider the mesh points indicated with the arrows. These are points necessary for Subdomain 1 which require interpolation from Subdomain 2. For a 5 point stencil, it is clear then that Subdomain 2 must overlap onto Subdomain 1 by 2 mesh points to ensure a centered interpolation.

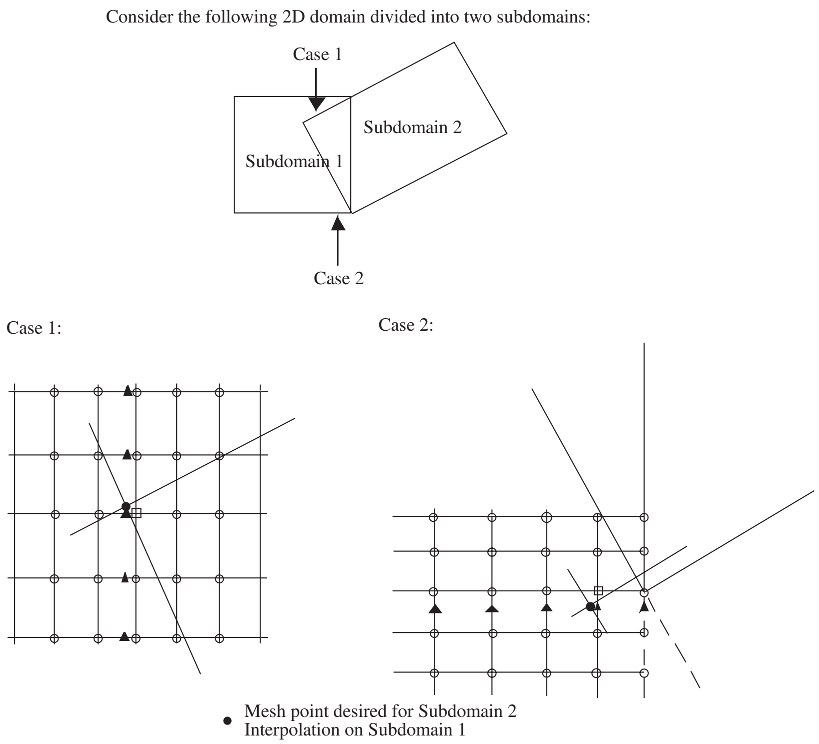
Consider the following 2D domain divided into two subdomains:
Case 1
Subdomain 2
Subdomain 1
Case 2
Case 1:
Case 2:
● Mesh point desired for Subdomain 2 Interpolation on Subdomain 1

Fig. 5. To demonstrate the requirements of 2D subdomain overlap, consider the mesh points indicated with the solid circles. These are points necessary for Subdomain 2 which require interpolation from Subdomain 1. The 5 point stencils shown indicate the center of the stencil with an open square. The open circles indicate the surrounding mesh points required for the 2D interpolation. The solid triangles indicate locations of resulting interpolations in one of the dimensions. These interpolations then must be interpolated one final time in the second dimension to obtain a final interpolated value. In Case 1, a centered interpolation is possible in both dimensions. The graphic indicates 5 interpolations in the x-direction followed by 1 interpolation in the y-direction. In Case 2, we are demonstrating a situation where due to boundary conditions a centered interpolation is not possible. In this case, it is possible to complete five centered interpolations in the y-direction first followed by one non-centered interpolation in the x-direction.

2. The time evolution of grid points in each subdomain uses point values only in that domain. For the points needed but not evolved, e.g., the three points shown with the arrows in Fig. 4, interpolation (either Lagrange or WENO) is performed from point values in the other subdomain, e.g., from the five points in Subdomain 2 shown by solid circles in Fig. 4.
3. The interpolation in 2D is done by first performing five interpolations in one of the directions, e.g., in the x-direction in Fig. 5 case 1 (lower left graph) to obtain the 5 points denoted by solid triangles, followed by one interpolation in the other direction, e.g., in the y-direction in Fig. 5 case 1 (lower left graph) to obtain the interpolated value at the solid circle point.

A significant concern with either interpolation method is that conservation error is introduced. In general, a conservative finite difference scheme for a one dimensional conservation law has the following form:

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x_j} (\hat{f}^n_{j+1/2} - \hat{f}^n_{j-1/2}). \tag{2.15}$$

Thus if the scheme is truly conservative and periodic boundary conditions are imposed:

$$\sum_j \Delta x_j \, u_j^{n+1} = \sum_j \Delta x_j \, u_j^n \tag{2.16}$$

which by induction leads to

$$\sum_j \Delta x_j \, u_j^n = \sum_j \Delta x_j \, u_j^0. \tag{2.17}$$

Then the conservation error is given by:

$$CE = \left| \sum_j \Delta x_j \, u_j^n - \sum_j \Delta x_j \, u_j^0 \right|. \tag{2.18}$$

A similar (but more complex) expression can be defined for the 2D case.

The numerical examples to follow exhibit that conservation error is introduced in multidomain problems with interpolation at subdomain interfaces. The magnitude of this error is small, and with smooth solutions the order of conservation error is consistently two. With non-smooth solutions, the order of conservation error reduces to one. In no example, however, do we observe a conservation error of $O(1)$. We always observe diminishing conservation error with grid refinement.

We remark that a result similar to that in [28], namely the conservation error goes to zero with mesh refinement, can be obtained for our method under similar assumptions as that in [28]. To be precise, the assumptions we need to make about the numerical solutions are:

1. The numerical solution u_j^n is uniformly bounded with respect to the discretization mesh sizes Δx_j and Δt.
2. The numerical solution u_j^n has a bounded total variation.
3. $$\sum_n \left| \hat{f}^{(1),n}_{J-\frac{1}{2}} - \hat{f}^{(2),n}_{J-\frac{1}{2}} \right| \Delta t \to 0, \tag{2.19}$$

   when $\Delta x_j, \Delta t \to 0$. Here J is the mesh index of an interface of two subdomains, $\hat{f}^{(1),n}_{J-\frac{1}{2}}$ and $\hat{f}^{(2),n}_{J-\frac{1}{2}}$ are the two numerical fluxes at $J-\frac{1}{2}$ for the two subdomains. These two fluxes are not equal to each other and they are not located at the same physical location because the two mesh sizes in the two subdomains are different.

Under these assumptions we can prove the Lax–Wendroff theorem, namely any converged solution will be a weak solution of the PDE. The proof is similar to that of the traditional Lax–Wendroff theorem, see, e.g., [13]. One extra (easy) proof is to show that the Lagrange or WENO interpolation based on a bounded numerical solution stays bounded. The third assumption (2.17) is reasonable because the difference between the two numerical fluxes from the two subdomains is $O(\Delta x)$ when the solution is smooth *and* when a stationary shock is at the interface due to the Rankine–Hugoniot jump condition. Thus this difference is significant only when a moving shock is at the interface, implying the sum in (2.17) should go to zero with mesh refinement. A similar assumption is also made in [28]. This explains why the conservation error is larger for a slow moving shock, as demonstrated numerically in the following section.

3. COMPUTATIONAL RESULTS

Although either the WENO or Lagrange interpolation algorithm at subdomain interfaces can be computed to any order of accuracy desired, for the purposes of this paper we display all computational results for both algorithms with 5th order accuracy in combination with the 5th order WENO finite difference method in the subdomain interiors using the Lax–Friedrichs flux splitting method [11].

3.1. Demonstration of WENO vs. Lagrange Interpolation

The usefulness of the WENO interpolation algorithm and its ability to more accurately capture function discontinuities without producing oscillations is easily demonstrated. Figure 6 shows that the WENO interpolation algorithm is much more suited to reconstruct a discontinuous function than the Lagrange interpolation algorithm. In both of the examples shown, the WENO interpolation algorithm

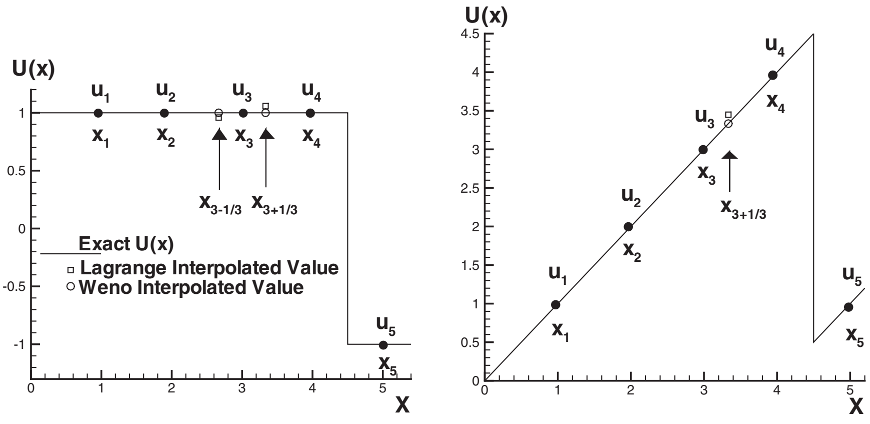

Fig. 6. Numerical examples demonstrating the theoretical effectiveness of the WENO interpolation algorithm over the Lagrange interpolation algorithm.

successfully capitalizes on its main idea which is to place much heavier weights on the candidate substencils through which a discontinuity is unlikely. The Lagrange interpolation algorithm, however, which uses a fixed stencil and disregards likely discontinuities is prone to over and undershoots.

3.2. Scalar Conservation Laws in One Dimension

Example 3.1 (Burgers' Equation). Here we solve the inviscid Burgers' equation with various initial conditions and final computation times:

$$u_t + \left(\frac{u^2}{2}\right)_x = 0$$
$$u(x, 0) = u_0(x).$$

Case 1. The first test case is that of a very slow moving shock. The initial condition for this slowly traveling shock is given as:

$$u(x, 0) = \begin{cases} 1.02, & \text{if} \quad x < 0.03; \\ -1.0, & \text{if} \quad x \geqslant 0.03. \end{cases}$$

This problem was used in [26] by Pärt–Enander and Sjögreen to show that a slowly traveling shock wave has difficulties in passing through the interface between two overlapping subdomains unless the interpolation between the subdomains is made conservative. Since both of our interpolation algorithms are nonconservative, we would like to test if difficulties appear in either method. It is clearly seen, however, in Fig. 7, top, that although both the Lagrange and WENO interpolation algorithms at subdomain interfaces cause the numerical solution to pace ahead of the exact solution, which is due to a conservation error from the interpolation, the WENO interpolation algorithm seems to result in a smaller conservation error for this case. We note that there are no other artifacts such as oscillations appearing in this case. We have also tested the two dimensional test case involving slow moving shocks in [26], again without noticing any significant artifacts. See Example 3.8 for details.

It is interesting and perhaps surprising to observe the lack of oscillations produced with the Lagrange interpolation method. See again Fig. 7, top. It is our belief that the strong stability feature of the base WENO finite difference method smoothes any oscillations produced by the Lagrange interpolation algorithm.

We also demonstrate through this example that, when the base WENO scheme is replaced by an ENO scheme, similar results can be obtained. We thus solve this problem under exactly the same conditions, except that we replace the fifth order WENO scheme by a fifth order ENO scheme [24, 25]. In Fig. 7, bottom, we show the result of using this ENO scheme. Comparing with Fig. 7, top, we can see that qualitatively the results are similar between these two schemes. For this example ENO seems to yield better results than WENO in terms of conservation errors.

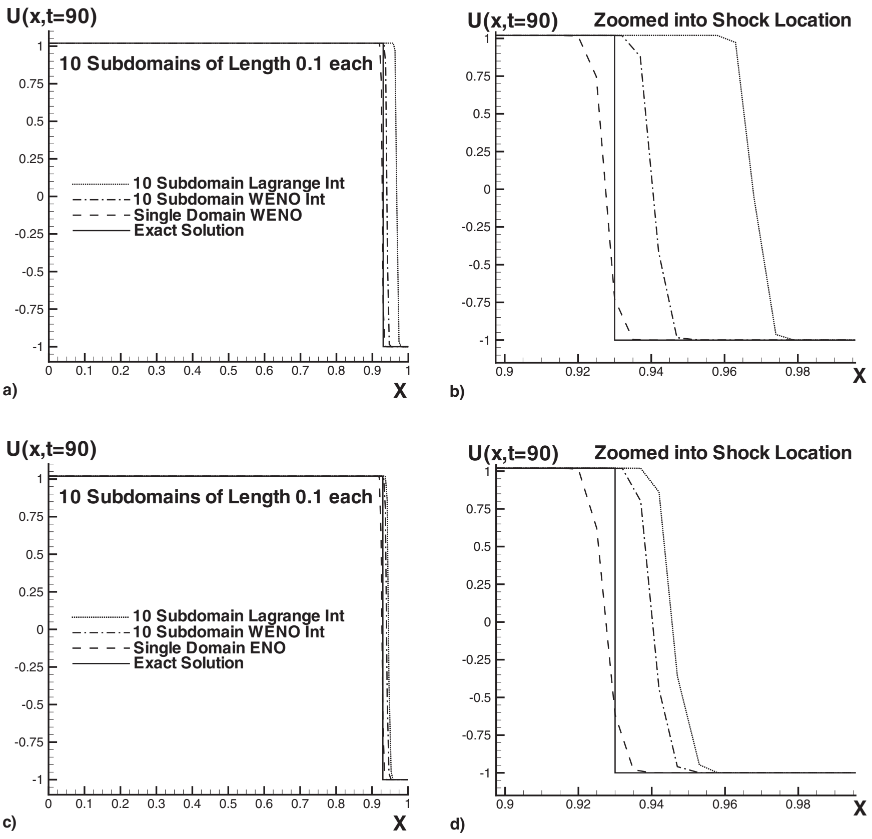

Fig. 7. Burgers' Equation with a slow traveling shock. Numerical solutions shown after more than 650000 time steps, having passed through nine subdomain interfaces. Subdomain $1 = x \in [0, 0.1]$, Subdomain $2 = x \in [0.1, 0.2]$,..., Subdomain $10 = x \in [0.9, 1.0]$. Mesh points in Subdomains 1 through 10 were chosen so that interpolation was required at every subdomain interface. They were, for each respective subdomain: 17, 19, 21, 20, 17, 19, 21, 20, 17, and 19 mesh points. Top: base WENO scheme; bottom: base ENO scheme. It is clear here that the WENO interpolation algorithm holds the shock much better than the Lagrange interpolation algorithm although the nonconservative nature of both algorithms is evident in that they both pace ahead of the shock.

Case 2. This next initial condition for Burgers equation was introduced to test the overall accuracy of the WENO finite difference method with both interpolation algorithms as well as the conservation error of both algorithms.

$$u(x, 0) = 0.3 + 0.1 \sin(\pi x), \quad -1 \leqslant x \leqslant 1 \qquad \text{and periodic.}$$

In Fig. 8, we display the numerical results of the Lagrange interpolation algorithm at subdomain interfaces and the WENO finite difference method in the subdomain interiors. The subdomain interfaces occur at $x = -1, 0, 1$ as the problem is periodic. Again, no oscillations are produced by the Lagrange interpolation

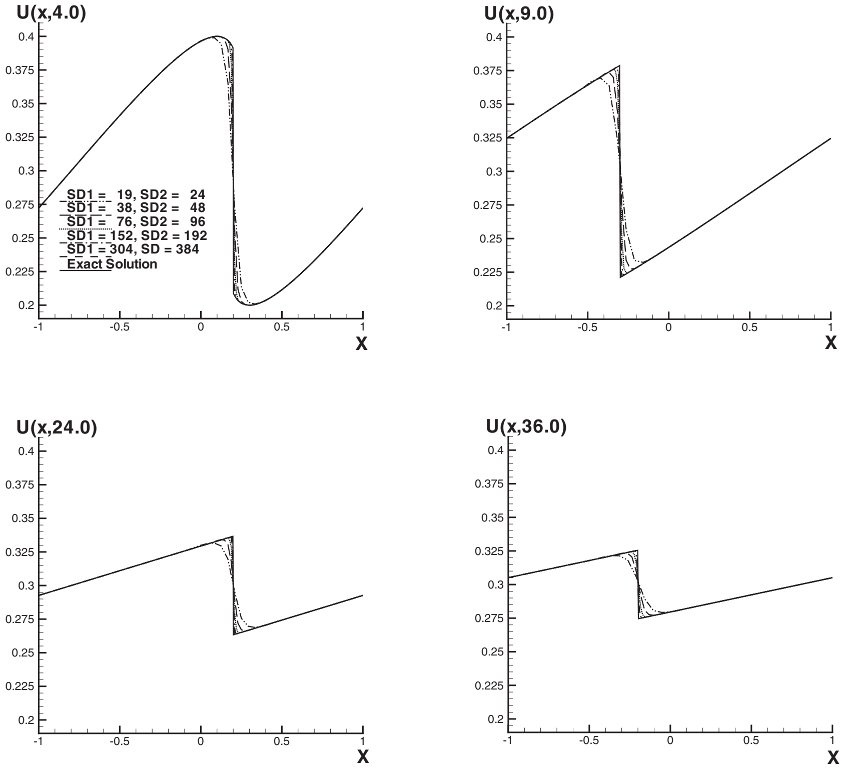

Fig. 8. Burgers' Equation with a smooth initial condition $u(x, 0) = 0.3 + 0.1 \sin(\pi x)$. Numerical solutions shown are at $t = 4$, 9, 24 and 36 and are typical of all final computation times. Subdomain $1 = x \in [-1, 0]$, Subdomain $2 = x \in [0, 1]$. Mesh points in Subdomains 1 and 2 were chosen so that interpolation was required at every subdomain interface. The periodicity of the problem allowed for many passes through subdomain interfaces. With this weak shock, it is hard to visually detect the conservation error in the plotted results. The WENO interpolation algorithm provides no visible improvement in the computations, thus the Lagrange interpolation results are shown only.

algorithm, at least none that are not smoothed out by the WENO finite difference method. Here, we begin to get an understanding that the WENO interpolation algorithm may not be a significant improvement over the Lagrange interpolation algorithm. Also see Table I for accuracy and conservation error results. Shock development in this problem occurs at $t = \frac{1}{0.1\pi} \approx 3.18$, thus accuracy results are given for computational times prior to shock development. Conservation error results are given for up to $t = 40$.

Case 3. This next initial condition for Burgers equation was introduced to test the ability of the WENO finite difference method with both interpolation algorithms to capture a much stronger shock. The initial condition is now chosen as

$$u(x, 0) = 17.4 + 13.3 \sin(\pi x), \qquad -1 \leqslant x \leqslant 1 \qquad \text{and periodic.}$$

Table I. Burgers' Equation with $u(x, 0) = 0.3 + 0.1 \sin(\pi x)$. Periodic Boundary Conditions. Two Subdomains: $SD1 = [-1, 0]$, $SD2 = [0, 1]$. L^1 and L^∞ Errors Given Prior to Shock Development at $t \approx 3.18$. Conservation Error Given Up to $t = 40$. Uniform Meshes on Each Subdomain with N_1 and N_2 Mesh Points. WENO-LF-5 Finite Difference Method on Subdomain Interiors with Lagrange Interpolation at Subdomain Interfaces. WENO Interpolation Results Are Similar and Hence Are Not Shown

		$N_{1,2} = 19, 24$	$N_{1,2} = 38, 48$		$N_{1,2} = 76, 96$		$N_{1,2} = 152, 192$	
t		Error	Error	Order	Error	Order	Error	Order
1	L^1	9.53E-06	2.92E-07	5.03	7.55E-09	5.27	2.02E-10	5.23
	L^∞	2.66E-05	8.57E-07	4.96	2.56E-08	5.07	7.78E-10	5.04
2	L^1	2.20E-04	1.29E-05	4.09	5.13E-07	4.65	1.61E-08	4.99
	L^∞	1.11E-03	1.25E-04	3.14	5.79E-06	4.44	1.92E-07	4.92
3	L^1	1.43E-05	7.44E-07	4.27	9.02E-09	6.37	1.60E-10	5.82
	L^∞	1.10E-04	7.24E-06	3.92	1.10E-07	6.04	4.05E-09	4.76
1	*CE*	2.77E-05	7.00E-06	1.98	1.75E-06	2.00	4.38E-07	2.00
2	*CE*	5.98E-05	1.52E-05	1.98	3.79E-06	2.00	9.48E-07	2.00
3	*CE*	1.30E-04	2.10E-05	2.63	5.27E-06	2.00	1.31E-06	2.01
5	*CE*	5.30E-04	2.13E-04	1.32	9.46E-05	1.17	4.39E-05	1.11
10	*CE*	4.32E-04	1.67E-04	1.37	7.15E-05	1.22	3.23E-05	1.15
20	*CE*	4.09E-04	2.51E-04	0.71	1.38E-04	0.87	7.27E-05	0.92
40	*CE*	4.74E-04	3.00E-04	0.66	1.63E-04	0.88	8.61E-05	0.92

In this case, it becomes evident upon viewing the numerical results graphically that both algorithms present noticeable conservation errors. Clearly, they both cause the resulting numerical shock location to lag behind the exact shock location. It is also interesting to see that the Lagrange interpolation algorithm appears to capture the exact shock location better than the WENO interpolation algorithm for this case. It is important, however, to notice that such conservation errors decay linearly with the mesh refinement. In Fig. 9, we display the numerical results graphically for both the Lagrange and WENO interpolation algorithms. In Table II, we show the calculated conservation errors for both algorithms.

Example 3.2 (Non-Convex Problem). Here we test the Lagrange and WENO interpolation algorithms on the Buckley–Leverett problem which is

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

where

$$f(u) = \frac{4u^2}{4u^2 + (1-u)^2}$$

with initial data $u(x, 0) = 1$ for $x \in [-\frac{1}{2}, 0]$ and $u = 0$ elsewhere. The exact solution is a shock-rarefaction-contact discontinuity mixture. The result obtained with Lagrange interpolation at the subdomain interface is shown in Fig. 10. It is clear

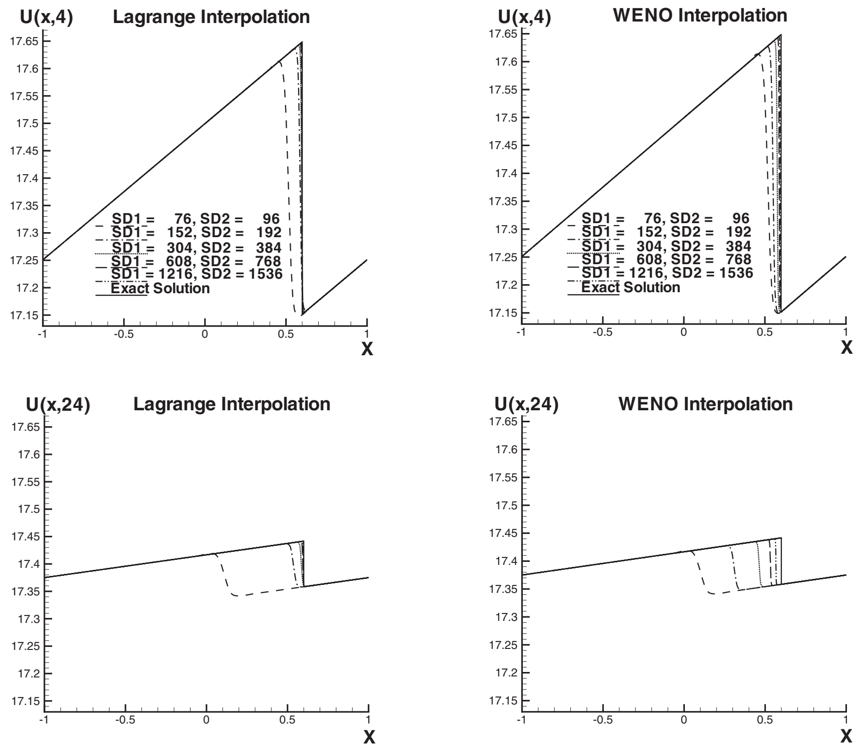

Fig. 9. Burgers' Equation with a smooth initial condition $u(x, 0) = 17.4 + 13.3 \sin(\pi x)$. This results in a much stronger and faster shock than that shown in Fig. 8. Numerical solutions shown are at $t = 4$ and 24 and are typical of all final computation times. Subdomain $1 = x \in [-1, 0]$, Subdomain $2 = x \in [0, 1]$. Mesh points in Subdomains 1 and 2 were chosen so that interpolation was required at every subdomain interface. The periodicity of the problem allowed for many passes through subdomain interfaces. With this strong shock, it is easy to visually detect the conservation error in the plotted results. The Lagrange interpolation algorithm provides some improvement over WENO interpolation in the conservation error observed in the computations, thus results for both types of interpolation are shown.

that even without a WENO interpolation, the scheme converges to the correct entropy solution and gives sharp, non-oscillatory shock profile.

3.3. Euler System in One Dimension

Example 3.3 (1D Riemann Problems). In considering the Euler system of equations in one dimension, we first examine two well-known problems which have the following Riemann type initial conditions:

$$\mathbf{u}(\mathbf{x}, \mathbf{0}) = \begin{cases} \mathbf{u_L} & \text{if} \quad x < 0 \\ \mathbf{u_R} & \text{if} \quad x > 0. \end{cases}$$

Table II. Burgers' Equation with $u(x, 0) = 17.4 + 13.3 \sin(\pi x)$. Periodic Boundary Conditions. Two Subdomains: SD1 = [−1, 0], SD2 = [0, 1]. Conservation Error Given Up to $t = 40$. Uniform Meshes on Each Subdomain with N_1 and N_2 Mesh Points. WENO-LF-5 Finite Difference Method on Subdomain Interiors with Lagrange (LI) and WENO (WI) Interpolation at Subdomain Interfaces

	Cons	$N_{1,2} = 76, 96$	$N_{1,2} = 152, 192$		$N_{1,2} = 304, 384$		$N_{1,2} = 608, 768$	
t	error	error	error	order	error	order	error	order
4	*LI*	4.23E-02	6.65E-03	2.67	2.14E-03	1.64	9.84E-04	1.12
	WI	4.09E-02	2.47E-02	0.73	1.20E-02	1.05	5.98E-03	1.00
20	*LI*	4.11E-02	5.38E-03	2.94	1.28E-03	2.07	5.05E-04	1.35
	WI	4.17E-02	2.43E-02	0.78	1.14E-02	1.09	5.64E-03	1.02
40	*LI*	4.10E-02	5.28E-03	2.96	1.21E-03	2.13	4.52E-04	1.42
	WI	4.19E-02	2.43E-02	0.78	1.14E-02	1.09	5.61E-03	1.03

Case 1. The first of these is Sod's problem. The initial data are:

$$(\rho_L, q_L, P_L) = (1, 0, 1), \qquad (\rho_R, q_R, P_R) = (0.125, 0, 0.1).$$

Case 2. The next of these is the problem proposed by Lax:

$$(\rho_L, q_L, P_L) = (0.445, 0.698, 3.528), \qquad (\rho_R, q_R, P_R) = (0.5, 0, 0.571).$$

The numerical results for these are presented in Fig. 11. Tests were run using the Lagrange interpolation algorithm and then compared to tests using the WENO interpolation algorithm. We remark that there are small oscillations (especially in

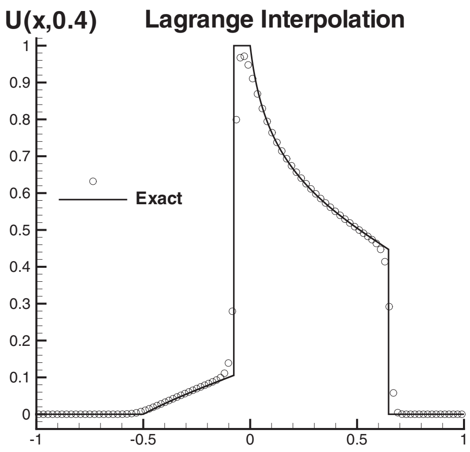

Fig. 10. Buckley–Leverett problem on four subdomains: Subdomain $1 = x \in [-1, -0.5]$; Subdomain $2 = x \in [-0.5, 0]$; Subdomain $3 = x \in [0, 0.5]$; and Subdomain $4 = x \in [0.5, 1]$. Number of mesh points are for each subdomain respectively 23, 27, 22, and 25.

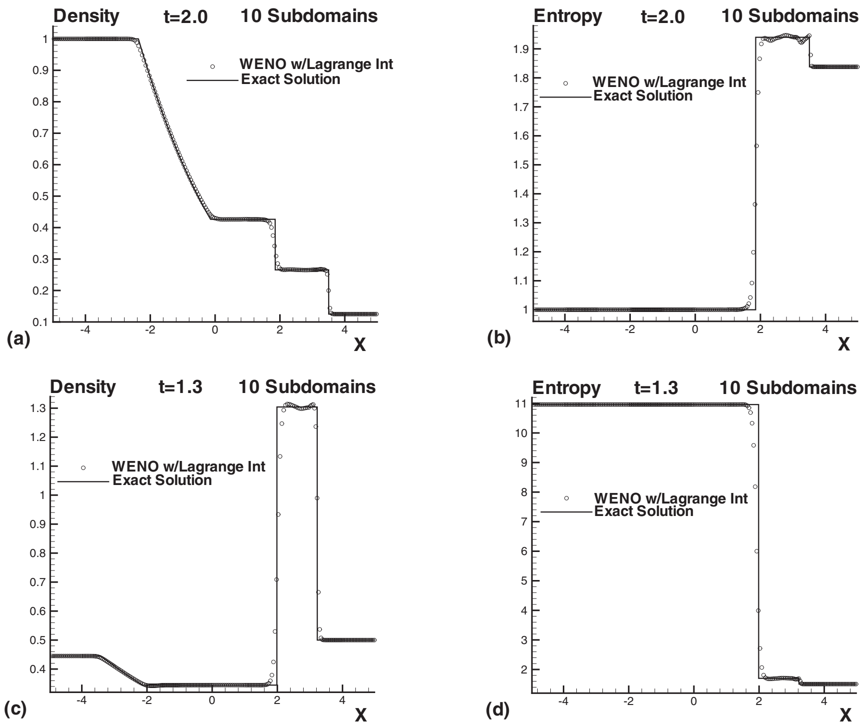

Fig. 11. Sod's problem (a), (b); Lax's Problem (c), (d). Showing only Lagrange interpolation algorithm results as both component WENO and characteristic WENO interpolation provided insignificantly better results. All computations computed with 10 subdomains; thus 9 subdomain interfaces required interpolation.

the graphs for entropy) which come from the inter-domain interpolation, as they are not there for a single domain calculation. However, a WENO interpolation does not seem to be able to remove these oscillations. Further investigation is needed here. However, these oscillations are minor and they do not seem to appear in other test problems we have tried.

Example 3.4 (Two Interacting Blast Waves). Next we consider the interaction of two blast waves. The initial data are

$$\mathbf{u}(\mathbf{x},\mathbf{0}) = \begin{cases} \mathbf{u_L} & \text{if} \quad 0 < x < 0.1 \\ \mathbf{u_M} & \text{if} \quad 0.1 < x < 0.9 \\ \mathbf{u_R} & \text{if} \quad 0.9 < x < 1, \end{cases}$$

where

$$\rho_L = \rho_M = \rho_R = 1, \qquad u_L = u_M = u_R = 0, \qquad P_L = 10^3, \qquad P_M = 10^{-2}, \qquad P_R = 10^2.$$

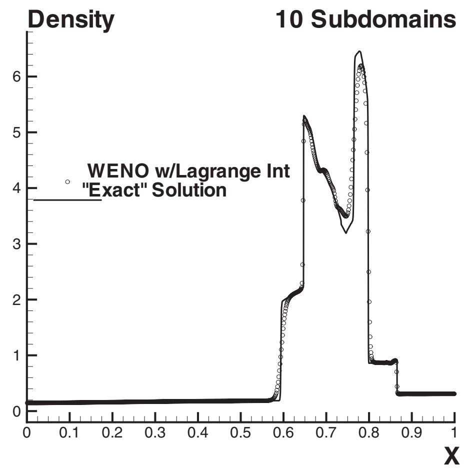

Fig. 12. Two interacting blast waves computed on 10 subdomains with interpolation required on 9 subdomain interfaces. Both interpolation algorithms provided comparable results, thus only the Lagrange interpolation algorithm results are shown.

The reflective boundary condition is applied at both $x = 0$ and $x = 1$. See [29] for a detailed discussion of this problem. Again we divide the domain into ten subdomains requiring interpolation at nine subdomain interfaces. The subdomains are all of equal length and the number of mesh points on each grid is approximately 80 with the actual number chosen so that interpolation at every subdomain interface is again required. The numerical solution is plotted at $t = 0.038$. The "exact" solution is computed by single domain WENO with 10000 mesh points. In Fig. 12, we only show the Lagrange interpolation algorithm results once again as there is no significant improvement made with the WENO interpolation algorithm.

Example 3.5 (Periodic 1D Riemann Problems). In this last example for the Euler System in one dimension, we set up the two Riemann problems described previous in a periodic manner to test once again for the conservation properties of both interpolation algorithms. That is, we now have the following initial conditions:

$$\mathbf{u}(\mathbf{x}, \mathbf{0}) = \begin{cases} \mathbf{u_L} & \text{if} \quad x < x_{LM} \\ \mathbf{u_R} & \text{if} \quad x_{LM} < x < x_{RM} \\ \mathbf{u_L} & \text{if} \quad x > x_{RM} \end{cases}$$

where $\mathbf{u_L}$ and $\mathbf{u_R}$ are as given before for the problems proposed by Sod and Lax respectively. And instead of the membrane for these "shock tube" type problems being placed at one location, namely $x = 0$, these periodic "shock tube" problems required two membranes, one at $x = x_{LM}$ and the other at $x = x_{RM}$.

In Fig. 13, we present the numerical results for these periodic Riemann problems. The periodic Sod's problem begins with a left membrane at $x = -7$ and a

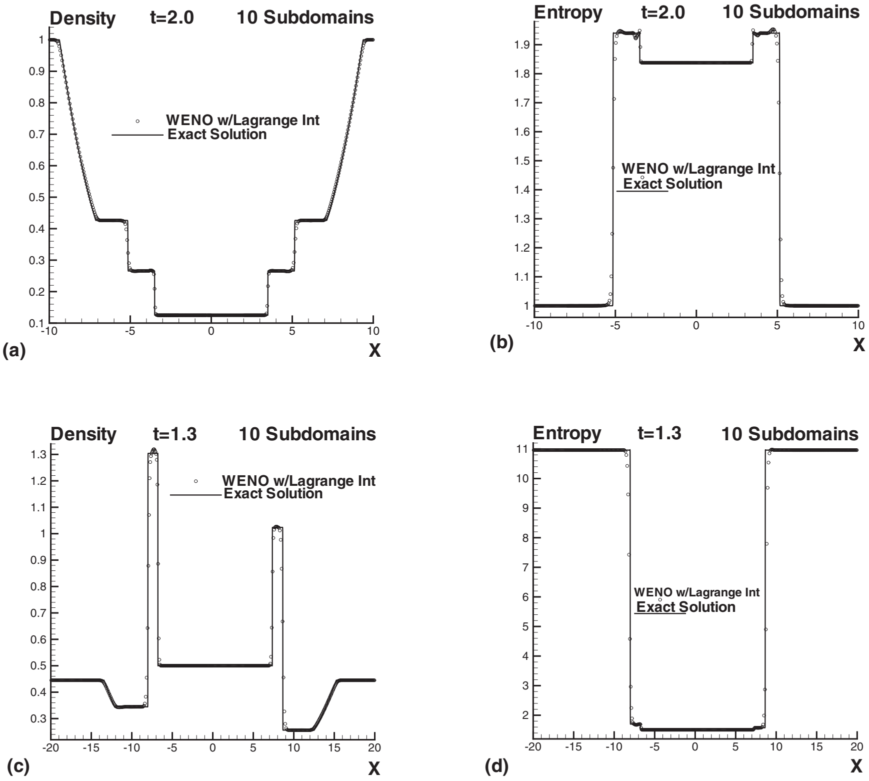

Fig. 13. Periodic Sod's problem (a), (b); Periodic Lax's Problem (c), (d). Showing only Lagrange interpolation algorithm results as both component WENO and characteristic WENO interpolation provided insignificantly better results. All computations computed with 10 subdomains.

right membrane at $x = 7$. The entire domain is taken as $x \in [-10, 10]$ and periodic. The domain is divided into 10 subdomains of length 2 each. The periodic Lax's problem begins with a left membrane at $x = -10$ and a right membrane at $x = 10$. The entire domain is taken as $x \in [-20, 20]$ and periodic. The domain is again divided into 10 subdomains. Because of the larger domain, they are of length 4 each in Lax's problem. The "exact" solution in each problem is presented as single domain WENO results with 10000 mesh points for Sod's problem and 20000 mesh points for Lax's problem. The WENO interpolation algorithm again does not provide significantly less oscillatory results, thus we again display only the Lagrange interpolation algorithm results. In Table III, we show the conservation error results for both the Lagrange interpolation algorithm and the WENO interpolation algorithm. It is clear that the conservation errors go to zero linearly with mesh refinements.

Table III. Euler System in One Dimension. The Riemann Problems Proposed by Sod and Lax Were Adapted for Periodicity in Order to Check the Conservation Error Introduced by the Lagrange and WENO Interpolation Algorithms. The Domain for Each Problem Was Divided Into Ten Subdomains so that Interpolation at Each Subdomain Interface Was Required. Mesh 1 Was such that Each Subdomain Contained Approximately 35 Mesh Points. Mesh 2 Was 2 Times Mesh 1, etc.

		Density		Momentum		Energy	
Mesh	Int Type	Error	order	Error	order	Error	order
		Sod's Double Shock Tube Problem					
1	Lagrange	2.02E-03		4.79E-04		1.46E-03	
	WENO	3.19E-03		7.95E-04		1.80E-02	
2	Lagrange	1.17E-03	0.78	4.50E-04	0.09	2.22E-03	−0.61
	WENO	1.27E-03	1.33	6.74E-04	0.24	7.69E-03	1.23
3	Lagrange	5.37E-04	1.13	2.15E-04	1.06	1.23E-03	0.85
	WENO	6.35E-04	1.00	3.16E-04	1.09	3.71E-03	1.05
4	Lagrange	2.44E-04	1.14	9.92E-05	1.12	6.38E-04	0.95
	WENO	3.29E-04	0.95	1.50E-04	1.08	1.83E-03	1.02
5	Lagrange	1.10E-04	1.15	4.30E-05	1.21	3.14E-04	1.02
	WENO	1.73E-04	0.93	6.65E-05	1.17	9.20E-04	1.00
		Lax's Double Shock Tube Problem					
1	Lagrange	2.19E-02		1.24E-02		1.94E-01	
	WENO	3.88E-02		2.67E-03		3.24E-01	
2	Lagrange	1.29E-02	0.76	5.07E-03	1.29	1.05E-01	0.89
	WENO	2.13E-02	0.87	2.06E-04	3.69	1.70E-01	0.93
3	Lagrange	6.78E-03	0.93	2.30E-03	1.14	5.32E-02	0.97
	WENO	1.10E-02	0.96	3.25E-04	−0.66	8.62E-02	0.98
4	Lagrange	3.54E-03	0.94	9.90E-04	1.21	2.70E-02	0.98
	WENO	5.75E-03	0.93	4.84E-04	−0.57	4.36E-02	0.98
5	Lagrange	1.74E-03	1.03	5.78E-04	0.78	1.35E-02	1.00
	WENO	2.87E-03	1.00	1.98E-04	1.29	2.19E-02	0.99

3.4. Scalar Conservation Laws in Two Dimensions

Example 3.6 (Burgers' Equation in 2D). Here we solve the inviscid Burgers' equation in 2D on two different types of domains. The equation with initial condition is given as:

$$u_t + \left(\frac{u^2}{2}\right)_x + \left(\frac{u^2}{2}\right)_y = 0$$
$$u(x, y, 0) = u_0(x, y).$$

Case 1. We first choose a rectangular domain and divide it into four subdomains as shown in Fig. 14. As we have been able to accomplish in all test cases to this point, we ensure that each subdomain overlaps adjacent subdomains by two

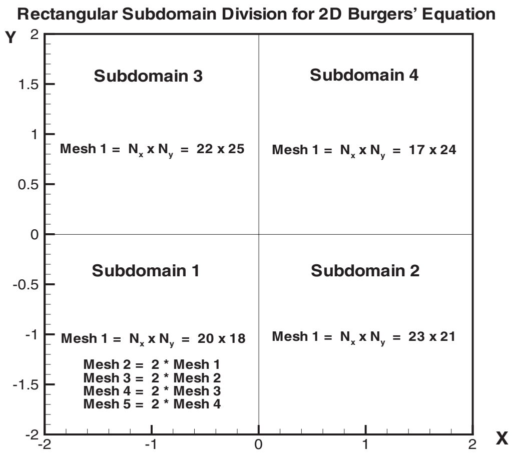
Rectangular Subdomain Division for 2D Burgers' Equation
Y
X
Subdomain 3
Subdomain 4
Subdomain 1
Subdomain 2
Mesh 1 = N_x x N_y = 22 x 25
Mesh 1 = N_x x N_y = 17 x 24
Mesh 1 = N_x x N_y = 20 x 18
Mesh 1 = N_x x N_y = 23 x 21
Mesh 2 = 2 * Mesh 1
Mesh 3 = 2 * Mesh 2
Mesh 4 = 2 * Mesh 3
Mesh 5 = 2 * Mesh 4

Fig. 14. Demonstrating the subdomain division of the rectangular domain upon which the 2D Burgers' equation was solved.

mesh points. This allows for the intuitively more stable centered interpolation procedure for all points requiring interpolation. We set the initial condition to be

$$u(x, y, 0) = 0.3 + 0.1 \sin\left(\frac{\pi}{2}(x+y)\right) \qquad \text{and periodic}$$

for which we have the exact solution. A shock wave develops at $t = \frac{1}{0.1\pi} \approx 3.18$. Table IV illustrates the L^1 and L^∞ errors obtained as well as the conservation error obtained when using both a simple Lagrange interpolation and the more complicated WENO interpolation. Figure 15 demonstrates the numerical results graphically for the Lagrange interpolation algorithm. We see once again that the Lagrange interpolation algorithm does not introduce oscillations in the results.

Case 2. We still solve the 2D Burgers' equation; however, this time we choose the domain to be partially rectangular and partially rotated rectangular. This is the first test case in which we can begin to simulate a more difficult domain that might require for practical purposes subdomain division. One might think of this domain, as illustrated in Fig. 16, as a model for a bend in a 2D duct.

The initial condition remains

$$u(x, y, 0) = 0.3 + 0.1 \sin\left(\frac{\pi}{2}(x+y)\right) \qquad \text{and periodic in } x \text{ and } y.$$

However, we now must solve the following equations:

$$u_t + \left(\frac{u^2}{2}\right)_x + \left(\frac{u^2}{2}\right)_y = 0 \qquad \text{on Subdomains 1 and 3}$$

$$u_t + \frac{\sqrt{3}+1}{2}\left(\frac{u^2}{2}\right)_\xi + \frac{\sqrt{3}-1}{2}\left(\frac{u^2}{2}\right)_\eta = 0 \qquad \text{on Subdomains 2 and 4.}$$

Table IV. 2D Burgers' Equation with Initial Condition $u(x, 0) = 0.3 + 0.1 \sin(\frac{\pi}{2}(x + y))$. Periodic Boundary Conditions. Four Subdomains as Pictured in Fig. 14. Shock Development Occurs at $t \approx 3.18$. Conservation Error Given at All Times Computed. Uniform Meshes on Each Subdomain with the Number of Mesh Points Given in Fig. 14. Illustrating 5th Order WENO Finite Difference Method on Subdomain Interiors with Both Lagrange and WENO Interpolation Algorithms Employed at the Subdomain Interfaces. (W) Indicates WENO Interpolation and (L) Indicates Lagrange Interpolation

t	(W) (L)	1st Mesh error	2nd Mesh error	2nd Mesh order	3rd Mesh error	3rd Mesh order	4th Mesh error	4th Mesh order
1	$W-L^1$	2.62E-05	8.59E-07	4.93	1.73E-08	5.63	3.56E-10	5.60
	$L-L^1$	2.56E-05	8.38E-07	4.93	1.70E-08	5.63	3.53E-10	5.58
	$W-L^\infty$	1.51E-05	8.03E-07	4.23	1.77E-08	5.50	1.96E-10	6.50
	$L-L^\infty$	1.51E-05	8.03E-07	4.23	1.77E-08	5.50	1.96E-10	6.50
2	$W-L^1$	6.18E-05	2.24E-06	4.78	4.15E-08	5.75	6.27E-10	6.05
	$L-L^1$	6.00E-05	2.17E-06	4.79	4.06E-08	5.74	6.18E-10	6.04
	$W-L^\infty$	6.41E-05	4.11E-06	3.96	1.04E-07	5.31	1.12E-09	6.53
	$L-L^\infty$	6.40E-05	4.11E-06	3.96	1.03E-07	5.31	1.13E-09	6.51
3	$W-L^1$	5.87E-05	1.27E-06	5.53	2.62E-08	5.60	5.02E-10	5.70
	$L-L^1$	5.61E-05	1.21E-06	5.54	2.56E-08	5.56	4.96E-10	5.69
	$W-L^\infty$	4.92E-05	2.14E-06	4.52	3.08E-08	6.12	1.89E-10	7.35
	$L-L^\infty$	4.92E-05	2.12E-06	4.53	3.05E-08	6.12	2.14E-10	7.15
1	$W-Cons$	1.33E-03	3.32E-04	2.00	8.30E-05	2.00	2.08E-05	2.00
	$L-Cons$	3.23E-04	7.97E-05	2.02	1.99E-05	2.00	4.98E-06	2.00
2	$W-Cons$	5.02E-04	1.33E-04	1.92	3.36E-05	1.99	8.42E-06	2.00
	$L-Cons$	3.67E-04	8.44E-05	2.13	2.08E-05	2.02	5.18E-06	2.00
3	$W-Cons$	2.17E-03	5.79E-04	1.90	1.51E-04	1.94	3.91E-05	1.95
	$L-Cons$	5.93E-04	1.17E-04	2.34	2.31E-05	2.34	4.49E-06	2.36
6	$W-Cons$	8.52E-03	3.54E-03	1.27	1.61E-03	1.14	7.68E-04	1.06
	$L-Cons$	1.27E-03	5.63E-04	1.17	2.59E-04	1.12	1.21E-04	1.09
10	$W-Cons$	2.39E-03	9.58E-04	1.32	5.00E-04	0.94	2.50E-04	1.00
	$L-Cons$	4.48E-03	2.33E-03	0.95	1.20E-03	0.96	6.08E-04	0.98

On Subdomains 2 and 4, we have the usual rotation transformation from the variables in the (x, y)-coordinate system to the variables in the (ξ, η)-coordinate system. This of course changes the flux in both the ξ-variable and the η-variable as shown in the previous equation. Again, a shock wave develops at $t \approx 3.18$, and we illustrate in Table V the L^1, L^∞, and conservation errors obtained when using either the Lagrange or WENO interpolation algorithms at the subdomain interfaces.

When examining Table V, note the inaccuracy in demonstrating 2nd order accuracy in the conservation error prior to shock development. The only explanation for this is the inherent difficulty in calculating conservation error in this problem. Recall conservation error is defined only in periodic problems. This problem is periodic in the (x, y)-coordinate system; however, we have subdomains which extend beyond the periodicity. In addition, the subdomains have much

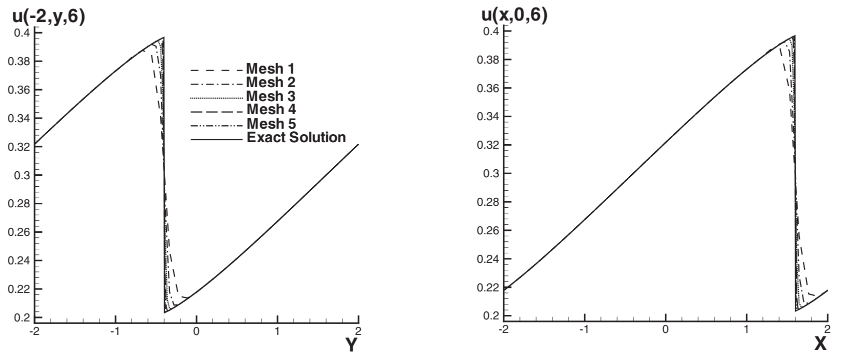

Fig. 15. Cuts of the solution to the 2D Burgers' equation along the lines $x = -2$ and $y = 0$. Both presented at time $t = 6$. These results are once again for the Lagrange interpolation algorithm. It is clear that the algorithm does not give oscillatory solutions.

greater overlap than before, and to check conservation error we must compute the following in 2D:

$$CE = \left| \sum_{i,j} \Delta x_i \, \Delta y_j \, u^n_{i,j} - \sum_{i,j} \Delta x_i \, \Delta y_j \, u^0_{i,j} \right|.$$

This is clear and easy to calculate for the rectangular domain. However, to ensure that no mesh points are counted twice and the periodicity is accounted for in the calculation with the rotated rectangular domain, the equation to determine the conservation error must be modified to read:

$$CE = \left| \sum_{i,j} \text{Area}_{i,j} \, u^n_{i,j} - \sum_{i,j} \text{Area}_{i,j} \, u^0_{i,j} \right|.$$

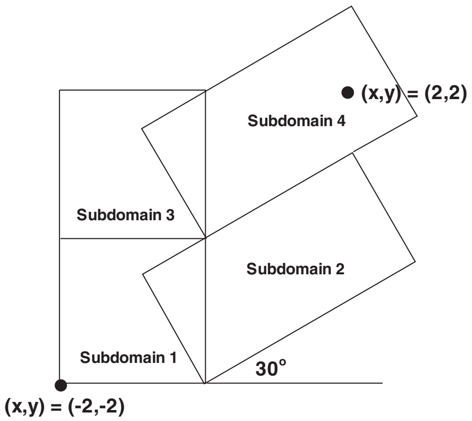

Fig. 16. Illustration of the subdomain division for the problem in which we are solving the 2D Burgers' equation on a rectangular rotated domain. The number of mesh points in each subdomain is identical to those denoted for the completely rectangular domain illustrated in Fig. 14.

Table V. 2D Burgers' Equation with Initial Condition $u(x,0)=0.3+0.1\sin(\frac{\pi}{2}(x+y))$. Periodic Boundary Conditions. Four Subdomains as Pictured in Fig. 16. Shock Development Occurs at $t\approx 3.18$. Conservation Error Given at All Times Computed. Uniform Meshes on Each Subdomain with the Number of Mesh Points Given in Fig. 16. Illustrating 5th Order WENO Finite Difference Method on Subdomain Interiors with Both Lagrange and WENO Interpolation Algorithms Employed at the Subdomain Interfaces. (W) Indicates WENO Interpolation and (L) Indicates Lagrange Interpolation

t	(W) (L)	1st Mesh error	2nd Mesh error	2nd Mesh order	3rd Mesh error	3rd Mesh order	4th Mesh error	4th Mesh order
1	$W-L^1$	2.43E-03	1.10E-04	4.46	3.88E-06	4.83	1.10E-07	5.15
	$L-L^1$	2.43E-03	1.10E-04	4.47	3.87E-06	4.83	1.09E-07	5.14
	$W-L^\infty$	1.42E-03	8.18E-05	4.11	2.70E-06	4.92	9.19E-08	4.88
	$L-L^\infty$	1.42E-03	8.18E-05	4.11	2.70E-06	4.92	9.19E-08	4.88
2	$W-L^1$	7.57E-04	3.49E-05	4.44	1.20E-06	4.86	2.54E-08	5.56
	$L-L^1$	7.56E-04	3.48E-05	4.44	1.19E-06	4.86	2.54E-08	5.55
	$W-L^\infty$	7.29E-04	3.16E-05	4.53	3.11E-06	3.34	5.02E-08	5.95
	$L-L^\infty$	7.29E-04	3.17E-05	4.52	3.11E-06	3.35	5.03E-08	5.95
3	$W-L^1$	9.48E-04	3.04E-05	4.96	5.36E-07	5.82	1.25E-08	5.42
	$L-L^1$	9.48E-04	3.03E-05	4.97	5.33E-07	5.83	1.25E-08	5.41
	$W-L^\infty$	7.46E-04	5.12E-05	3.86	6.60E-07	6.29	2.06E-08	5.00
	$L-L^\infty$	7.45E-04	5.12E-05	3.86	6.60E-07	6.28	2.06E-08	5.00
1	$W-Cons$	3.96E-04	5.37E-05	2.88	5.00E-06	3.43	5.98E-07	3.06
	$L-Cons$	3.96E-04	5.38E-05	2.88	5.00E-06	3.43	5.98E-07	3.06
2	$W-Cons$	1.80E-04	6.33E-06	4.83	1.18E-05	−0.90	2.48E-06	2.25
	$L-Cons$	1.83E-04	6.24E-06	4.87	1.18E-05	−0.92	2.48E-06	2.25
3	$W-Cons$	1.03E-04	2.32E-04	−1.18	2.32E-05	3.32	2.48E-06	3.22
	$L-Cons$	6.50E-05	2.30E-04	−1.82	2.34E-05	3.30	2.38E-06	3.29
6	$W-Cons$	7.23E-03	4.61E-03	0.65	2.59E-03	0.83	1.49E-03	0.80
	$L-Cons$	6.40E-03	4.27E-03	0.59	2.52E-03	0.76	1.38E-03	0.87
10	$W-Cons$	1.37E-02	6.24E-03	1.14	2.97E-03	1.07	1.61E-03	0.89
	$L-Cons$	1.24E-02	5.62E-03	1.15	2.83E-03	0.99	1.44E-03	0.98

where Area$_{i,j}$ is the area of the cell surrounding the meshpoint located at (x_i, y_j). See Fig. 17 for one example of the many possible cases that must be accounted for when determining conservation error with this 2D rectangular rotated domain.

As this problem is still periodic in the (x, y)-coordinate system, we are still able to overlap each subdomain by at least two mesh points in order to compute the WENO finite difference method in each subdomain interior with both Lagrange and WENO interpolation at subdomain interfaces. Once again, we see no improvement with the more complicated WENO interpolation algorithm. However, this problem is our first demonstration of the ability either interpolation algorithm possesses to perform the interpolations on adjacent subdomains of a different geometry. This will allow us in the future to work with the finite difference method in more complicated domains. As stated in the introduction to this paper, our goal here was to develop a method for being able to work with finite difference methods

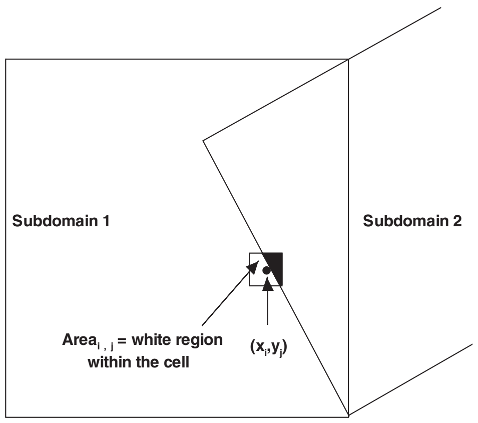

Fig. 17. Illustration of just one of the many difficult cases in calculating the conservation error with the rotated rectangular subdomain division presented in Fig. 16. The white area of the cell surrounding the mesh point (x_i, y_j) must be calculated and the value of the solution then multiplied by this area. There are many other cases to consider and all area must be accounted for or an inaccurate calculation of conservation error will occur.

in more complicated domains. This first look at the ability to interpolate between adjacent subdomains of different geometry offers significant advancement towards this goal.

In Fig. 18, we exhibit numerical results of the 2D solution to this equation over the entire domain using Lagrange interpolation at the subdomain interfaces. Careful examination against the exact solution leads us to conclude that Lagrange interpolation is once again sufficient. Cuts along different lines shown in Fig. 19 demonstrate even more clearly that once again we do not observe any oscillations in the results. Recall here that to obtain a two dimensional interpolation with 5th order accuracy, we must first compute 5 one dimensional interpolations in one of the dimensions followed by a final interpolation in the other dimension.

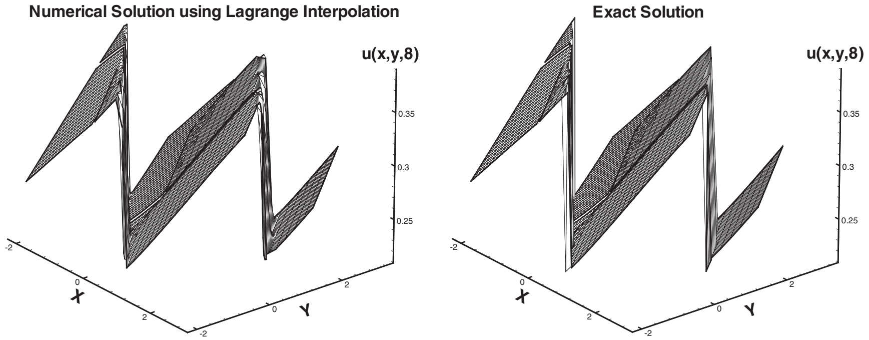

Fig. 18. (a) Numerical Solution for the 2D Burgers' equation for Mesh 3 as outlined in Fig. 16. (b) Exact Solution for the 2D Burgers' equation. Both presented at time $t = 8$. For clarity, both figures are given showing every fourth mesh point in both directions.

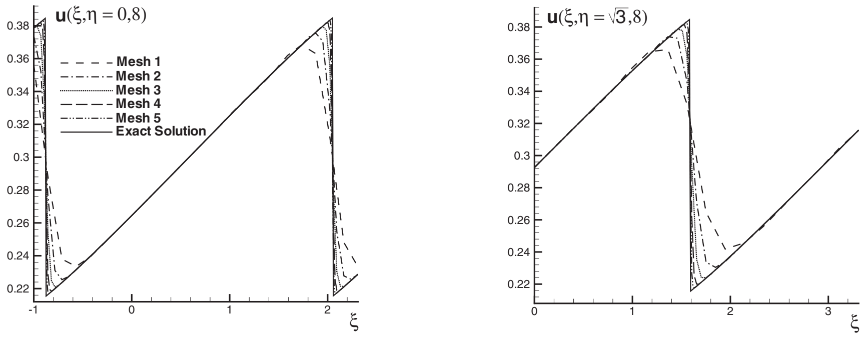

Fig. 19. Cuts of the solution to the 2D Burgers' equation along the lines $\eta = 0$ and $\eta = \sqrt{3}$. Both presented at time $t = 8$. These results are once again for the Lagrange interpolation algorithm. It remains clear that the algorithm does not give oscillatory solutions for Burgers' equation.

3.5. Euler System in Two Dimensions

Example 3.7. Euler System with smooth initial conditions generating a smooth exact solution, that is

$$\begin{aligned} \rho(x, y, 0) &= 1 + 0.1 \sin(\pi(x+y)) \\ u(x, y, 0) &= 1 \\ v(x, y, 0) &= -0.7 \\ p(x, y, 0) &= 1. \end{aligned}$$

These provide the exact solution $\rho(x, y, 0) = 1 + 0.2 \sin(\pi(x + y - (u+v)\,t))$. With this we can check the accuracy and conservation error of the WENO finite difference method in a multidomain problem with either Lagrange or WENO interpolation at the subdomain interfaces. We have computed the results for the two types of 2D domains we introduced previously. As the results are comparable, we only show here in Table VI those over the more complicated combination rectangular and rotated rectangular domain demonstrated earlier in Fig. 16. The number of mesh points for each refinement is also the same as before. We can clearly observe fifth order accuracy and second order in conservation errors.

Example 3.8. Euler system of equations in 2D with a slow moving shock in the positive y-direction. The problem was introduced in [26]. The initial data

$$\mathbf{u(x, y, 0)} = \begin{cases} (1.4 \quad 0 \quad 4.2 \quad 8.8)^T & \text{if} \quad y < 0.67; \\ (5.38064 \quad 0 \quad 4.25971 \quad 27.2576)^T & \text{if} \quad y \geqslant 0.67 \end{cases}$$

were chosen such that the exact solution is a shock moving with speed $s = 0.015$ in the positive y-direction. Computations were done using a two subdomain setup. An "outer" rectangular subdomain upon which a "hole" is cut for an "inner"

Table VI. L^1 and L^∞ Errors and Conservation Errors for the Euler System in 2D with Smooth Exact Solution. Showing Both the WENO Interpolation (W) and the Lagrange Interpolation (L) Algorithms

t	(W) (L)	1st Mesh error	2nd Mesh error	2nd Mesh order	3rd Mesh error	3rd Mesh order	4th Mesh error	4th Mesh order
5	$W-L^1$	1.25E-02	3.98E-04	4.97	1.20E-05	5.05	3.66E-07	5.04
	$L-L^1$	1.20E-02	3.89E-04	4.95	1.19E-05	5.03	3.63E-07	5.03
	$W-L^\infty$	6.41E-03	2.21E-04	4.86	6.73E-06	5.03	1.98E-07	5.08
	$L-L^\infty$	5.72E-03	2.10E-04	4.77	6.68E-06	4.97	1.96E-07	5.09
10	$W-L^1$	2.33E-02	7.98E-04	4.87	2.41E-05	5.05	7.34E-07	5.04
	$L-L^1$	2.25E-02	7.83E-04	4.85	2.39E-05	5.03	7.31E-07	5.03
	$W-L^\infty$	1.04E-02	3.92E-04	4.73	1.26E-05	4.96	3.81E-07	5.04
	$L-L^\infty$	9.72E-03	3.84E-04	4.66	1.25E-05	4.94	3.80E-07	5.04
5	$W-Cons$	1.67E-03	3.45E-04	2.27	8.49E-05	2.02	1.81E-05	2.23
	$L-Cons$	1.73E-03	3.46E-04	2.32	8.49E-05	2.03	1.81E-05	2.23
10	$W-Cons$	2.15E-03	4.53E-04	2.24	1.08E-04	2.07	2.20E-05	2.29
	$L-Cons$	2.15E-03	4.55E-04	2.24	1.08E-05	2.07	2.20E-05	2.29

subdomain complete the domain setup. Periodic boundary condition is applied in the x-direction and exact boundary condition is given in the y-direction. Outlines of these two subdomains along with density contours are shown in each part of Fig. 20. We remark here that the results for this test case are comparable to the results given earlier for the 1D slow moving shock using Burgers' Equation. Both the Lagrange and WENO interpolation schemes cause the numerical shock to pace ahead of the exact shock location. Also similar to the 1D slow moving shock case is the observation that the WENO interpolation scheme holds the correct shock location noticeably better than the Lagrange interpolation scheme. However, we observe graphically a first order convergence to the correct shock location with refinement for both interpolation schemes.

Example 3.9. Euler system of equations in 2D with the combination rectangular, rotated rectangular type domain exhibited previously. Note that on subdomains one and three, we can solve the usual 2D Euler system in the (x, y)-plane given as:

$$\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix}_x + \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}_y = 0.$$

Whereas on subdomains two and four which are the rotated rectangular subdomains, we now must solve the rotated 2D Euler system of equations on the (ξ, η)-plane which are given as:

$$\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix}_t+\begin{pmatrix}\rho(\cos\theta u+\sin\theta v)\\ \rho u(\cos\theta u+\sin\theta v)+\cos\theta p\\ \rho v(\cos\theta u+\sin\theta v)+\sin\theta p\\ (E+p)(\cos\theta u+\sin\theta v)\end{pmatrix}_\xi+\begin{pmatrix}\rho(-\sin\theta u+\cos\theta v)\\ \rho u(-\sin\theta u+\cos\theta v)-\sin\theta p\\ \rho v(-\sin\theta u+\cos\theta v)+\cos\theta p\\ (E+p)(-\sin\theta u+\cos\theta v)\end{pmatrix}_\eta=0$$

where θ is the angle through which the axes are rotated counterclockwise.

We have tested this setup using Sod's initial conditions extended to two dimensions but will omit the details here. We are mainly interested to solve the

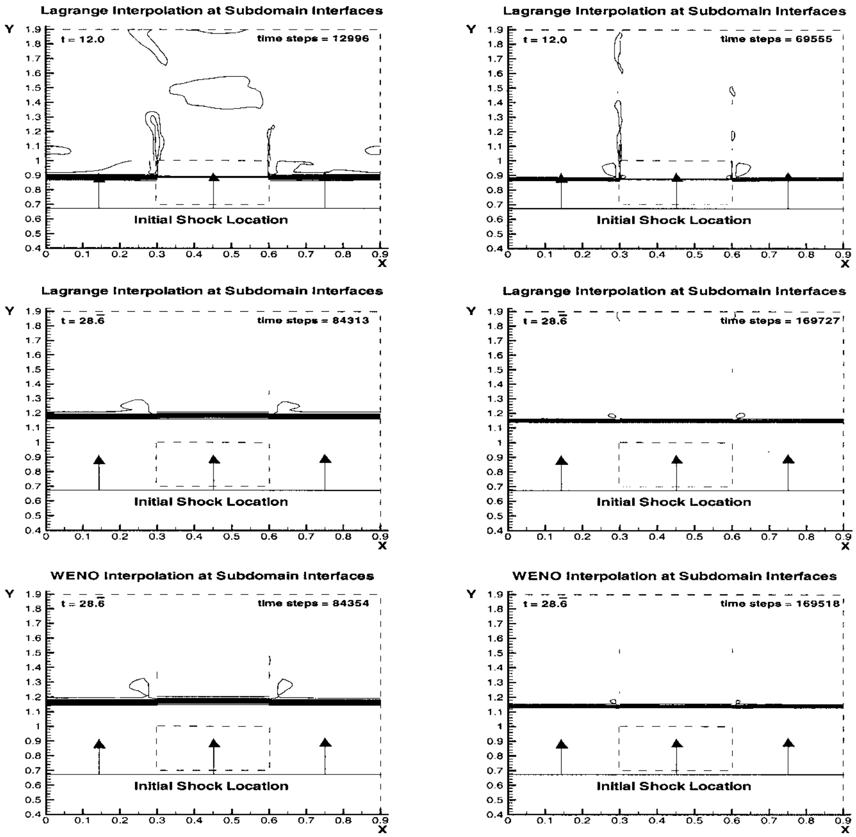

Fig. 20. Slow Moving Shock in the positive y-direction. 30 Density contours from $\rho = 1.3$ to $\rho = 6.0$. Two subdomains are used as indicated by the dotted lines. The plots on the left give the results for a coarse mesh: "Outer" subdomain: $N_x \times N_y = 75 \times 115$. "Inner" subdomain: $N_x \times N_y = 82 \times 114$. The plots on the right give the results for a refined mesh: "Outer" subdomain: $N_x \times N_y = 150 \times 230$. "Inner" subdomain: $N_x \times N_y = 164 \times 228$. The upper plots show the results using the Lagrange interpolation scheme for $t = 12.0$. The exact shock location for this time is $y = 0.85$. The middle and lower plots show the results using both the Lagrange and WENO interpolation schemes for $t = 28.\overline{6}$. The exact shock location for this time is $y = 1.1$.

double Mach reflection problem, see [29]. The physical domain for this problem is not rectangular but a union of two rectangles with an angle in between, see Fig. 21. Most of the calculations in the literature emulate this physical situation in a rectangle with an artificial boundary condition for a lower left portion of the boundary, see [29]. There are also attempts to solve it directly in the physical domain but with a curvilinear mesh which has a singularity at the corner, hence lowering the order of the scheme there. With the use of our interpolation algorithms, however, we can now easily solve the problem in the original physical domain without lowering the order of accuracy anywhere. The computational domain is that shown in Fig. 21 depicting the numerical results. The domain is divided into two subdomains

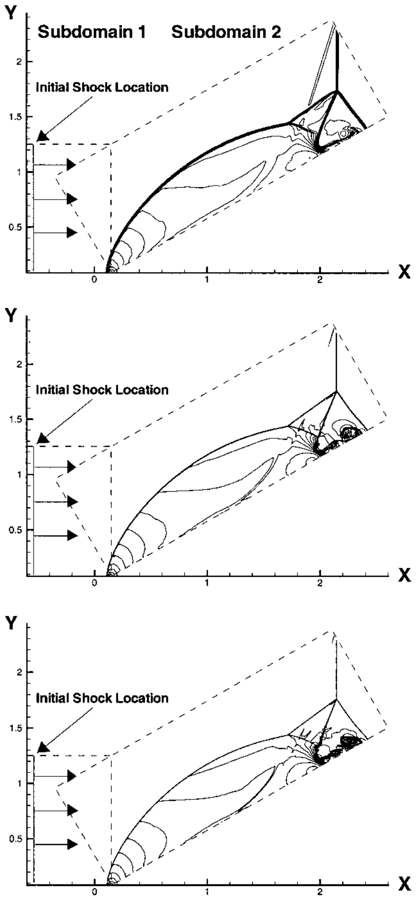

Fig. 21. Double Mach reflection. 30 Density contours from $\rho = 1.731$ to $\rho = 20.92$. Top: Grid 1 with 89×138 on Subdomain 1 with 460×119 on Subdomain 2; middle: Grid 2 with 356×552 on Subdomain 1 with 1840×476 on Subdomain 2; bottom: Grid 3 with 712×1104 on Subdomain 1 with 3680×952 on Subdomain 2.

with the bottom of each subdomain being a reflecting wall. The subdomain boundaries are represented by dashed lines. We note that, since the bottom of each subdomain is a reflecting wall, this test case requires non-centered interpolations between the subdomains at a few points. This, however, did not appear to introduce any instability into the numerical problem. Initially a Mach 10 shock is positioned at $x = -0.55$ making an angle of 90° with the x-axis. For the bottom boundaries, a reflective boundary condition is used. At the top boundaries of the domain, the flow values are set to describe the exact motion of the Mach 10 shock.

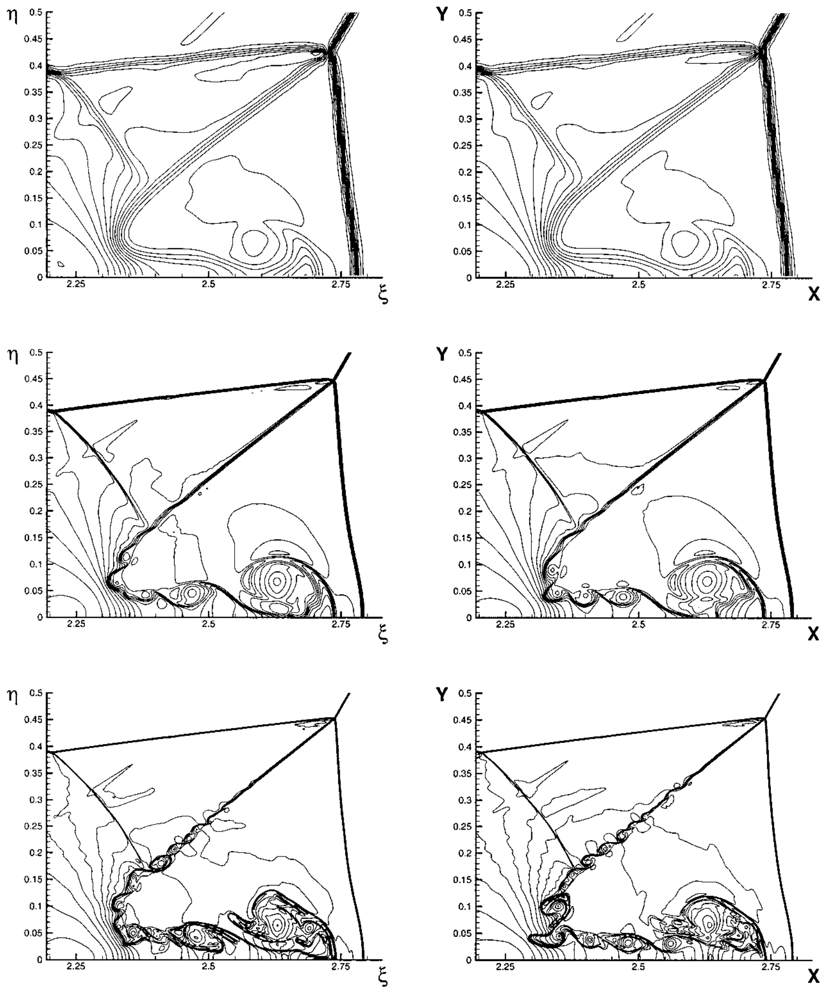

Fig. 22. Double Mach reflection. 30 Density contours from $\rho = 1.731$ to $\rho = 20.92$. Zoomed-in region. Left: two-domain calculation with domain sizes described in Fig. 21; Right: one-domain calculation with comparable mesh sizes.

We have used three grids in our tests: 89×138 on Subdomain 1 with 460×119 on Subdomain 2; 356×552 on Subdomain 1 with 1840×476 on Subdomain 2; and 712×1104 on Subdomain 1 with 3680×952 on Subdomain 2. The results are shown in Fig. 21 for the complete domain and in Fig. 22, left, for the zoomed-in region near the double Mach reflection. For comparison, single domain simulation results using the same finite difference fifth order WENO scheme with the same grid resolution are shown in Fig. 22, right, for the zoomed-in region. We can clearly see that the multidomain calculation is very similar to the single domain calculation, indicating that the Lagrange interpolation between the two subdomains has not introduced any artifact and has not affected the resolution of the base scheme for this problem. We would like to emphasize that the purpose of using a high order method on this problem is to show the smallness of the numerical viscosity for such schemes, demonstrated by the appearence of the small scale features (roll-ups) near the double Mach stem. The details of these small scale features may not be physical, as the numerical viscosity is not of the same form as the physical one. In order to obtain physically correct small scale features we would need to use a Navier–Stokes equation solver.

4. CONCLUDING REMARKS

We have developed multidomain finite difference WENO schemes in 1D and 2D to solve problems with complex domain geometry and/or different mesh sizes. A Lagrange or WENO interpolation is performed at the subdomain interfaces to pass information, and extensive numerical experiments have been performed to verify the accuracy, stability, and conservation error reduction with mesh refinements. For all the test cases the simple Lagrange interpolation can produce comparably good results as the more costly WENO interpolation, hence we would advocate using the Lagrange interpolation coupled with high order WENO finite difference schemes to solve such problems. Extensions to 3D problems constitute current work.

ACKNOWLEDGMENTS

Research supported by ARO Grant DAAD19-00-1-0405, NSF Grants DMS-9804985, DMS-0207451 and ECS-9906606, NASA Langley Grant NCC1-01035 and AFOSR Grant F49620-99-1-0077.

REFERENCES

1. Balsara, D., and Shu, C.-W. (2000). Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy. *J. Comput. Phys.* **160**, 405–452.
2. Berger, M., and Colella, P. (1989). Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.* **82**, 64–84.
3. Casper, J., Shu, C.-W., and Atkins, H. L. (1994). Comparison of two formulations for high-order accurate essentially nonoscillatory schemes. *AIAA J.* **32**, 1970–1977.
4. Del Zanna, L., Velli, M., and Londrillo, P. (1998). Dynamical response of a stellar atmosphere to pressure perturbations: Numerical simulations. *Astron. Astrophys.* **330**, L13–L16.
5. Friedrichs, O. (1998). Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. *J. Comput. Phys.* **144**, 194–212.

6. Grasso, F., and Pirozzoli, S. (2000). Shock-wave-vortex interactions: Shock and vortex deformations, and sound production. *Theor. Comp. Fluid Dyn.* **13**, 421–456.
7. Grasso, F., and Pirozzoli, S. (2000). Shock wave-thermal inhomogeneity interactions: Analysis and numerical simulations of sound generation. *Phys. Fluids* **12**, 205–219.
8. Harten, A., Engquist, B., Osher, S., and Chakravarthy, S. (1987). Uniformly high order essentially non-oscillatory schemes, III. *J. Comput. Phys.* **71**, 231–303.
9. Hu, C., and Shu, C.-W. (1999). Weighted essentially non-oscillatory schemes on triangular meshes. *J. Comput. Phys.* **150**, 97–127.
10. Jiang, G., and Peng, D.-P. (2000). Weighted ENO schemes for Hamilton–Jacobi equations. *SIAM J. Sci. Comput.* **21**, 2126–2143.
11. Jiang, G., and Shu, C.-W. (1996). Efficient implementation of weighted ENO schemes. *J. Comput. Phys.* **126**, 202–228.
12. Jiang, G., and Wu, C.-C. (1999). A high order WENO finite difference scheme for the equations of ideal magnetohydrodynamics. *J. Comput. Phys.* **150**, 561–594.
13. Le Veque, R. J. (1990). *Numerical Solutions for Conservation Laws*, Birkhäuser Verlag.
14. Liang, S., and Chen, H. (1999). Numerical simulation of underwater blast-wave focusing using a high-order scheme. *AIAA J.* **37**, 1010–1013.
15. Liska, R., and Wendroff, B. (1998). Composite schemes for conservation laws. *SIAM J. Numer. Anal.* **35**, 2250–2271.
16. Liska, R., and Wendroff, B. (1999). Two-dimensional shallow water equations by composite schemes. *Int. J. Numer. Meth. Fluids* **30**, 461–479.
17. Liu, X.-D., Osher, S., and Chan, T. (1994). Weighted essentially non-oscillatory schemes. *J. Comput. Phys.* **115**, 200–212.
18. Montarnal, P., and Shu, C.-W. (1999). Real gas computation using an energy relaxation method and high order WENO schemes. *J. Comput. Phys.* **148**, 59–80.
19. Noelle, S. (2000). The MoT-ICE: a new high-resolution wave-propagation algorithm for multidimensional systems of conservation laws based on Fey's method of transport. *J. Comput. Phys.* **164**, 283–334.
20. Shi, J., Hu, C., and Shu, C.-W. (2002). A technique of treating negative weights in WENO schemes. *J. Comput. Phys.* **175**, 108–127.
21. Shu, C.-W. (1998). Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In Cockburn, B., Johnson, C., Shu, C.-W., and Tadmor, E. (Quarteroni, A. (ed)), *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, Lecture Notes in Mathematics, Vol. 1697, Springer, pp. 325–432.
22. Shu, C.-W. (1999). High order ENO and WENO schemes for computational fluid dynamics. In Barth, T. J., and Deconinck, H. (eds.), *High-Order Methods for Computational Physics*, Lecture Notes in Computational Science and Engineering, Vol. 9, Springer, pp. 439–582.
23. Shu, C.-W. High order finite difference and finite volume WENO schemes and discontinuous Galerkin methods for CFD. *Int. J. Comput. Fluid Dyn.*, to appear.
24. Shu, C.-W., and Osher, S. (1988). Efficient implementation of essentially non-oscillatory shock capturing schemes. *J. Comput. Phys.* **77**, 439–471.
25. Shu, C.-W., and Osher, S. (1989). Efficient implementation of essentially non-oscillatory shock capturing schemes, II. *J. Comput. Phys.* **83**, 32–78.
26. Part-Enander, E., and Sjogreen, B. (1994). Conservative and non-conservative interpolation between overlapping grids for finite volume solutions of hyperbolic problems. *Computers Fluids* **23**, 551–574.
27. Quemere, P., Sagaut, P., and Couailler, V. (2001). A new multidomain/multiresolution method for large-eddy simulation. *Int. J. Numer. Meth. Fluids* **36**, 391–416.
28. Tang, H.-S., and Zhou, T. (1999). On nonconservative algorithms for grid interfaces. *SIAM J. Numer. Anal.* **37**, 173–193.
29. Woodward, P., and Colella, P. (1984). The numerical simulation of two-dimensional fluid flow with strong shocks. *J. Comput. Phys.* **54**, 115–173.
30. Yang, J., Yang, S., Chen, Y., and Hsu, C. (1998). Implicit weighted ENO schemes for the three-dimensional incompressible Navier–Stokes equations. *J. Comput. Phys.* **146**, 464–487.
31. Zhang, Y.-T. and Shu, C.-W. High order WENO schemes for Hamilton–Jacobi equations on triangular meshes. *SIAM J. Sci. Comput.*, to appear.