Journal of Computational Physics **167,** 413–438 (2001)
doi:10.1006/jcph.2000.6686, available online at http://www.idealibrary.com on IDEAL®

# A Level-Set Algorithm for Tracking Discontinuities in Hyperbolic Conservation Laws

## I. Scalar Equations

Tariq D. Aslam

*Los Alamos National Laboratory, Los Alamos, New Mexico 87545*
E-mail: aslam@lanl.gov

Received December 27, 1999; revised November 21, 2000

A level-set algorithm for tracking discontinuities in hyperbolic conservation laws is presented. The algorithm uses a simple finite difference approach, analogous to the method of lines scheme presented in C.-W. Shu and S. Osher (1988, *J. Comput. Phys.* **77**, 439). The zero of a level-set function is used to specify the location of the discontinuity. Since a level-set function is used to describe the front location, no extra data structures are needed to keep track of the location of the discontinuity. Also, two solution states are used at all computational nodes, one corresponding to the "real" state, and one corresponding to a "ghost node" state, analogous to the "Ghost Fluid Method" of R. P. Fedkiw *et al.* (1999, *J. Comput. phys.* **154**, 459). High-order pointwise convergence is demonstrated for linear and nonlinear conservation laws, even at discontinuities and in multiple dimensions. The solutions are compared to standard high-order shock-capturing schemes. This paper focuses on scalar conservation laws. An example is given for shock tracking in the one-dimensional Euler equations. Level-set tracking for systems of conservation laws in multidimensions will be presented in future work.

*Key Words:* shock tracking; ENO; WENO.

## 1. INTRODUCTION

While high-order shock-capturing schemes have proven to be invaluable tools in solving hyperbolic conservation laws, they generally do not converge at a high-order in the presence of discontinuities. Typically, there will be a few "intermediate" points within a numerical shock profile. The state at these "intermediate" points are in error by an $O(1)$ amount, causing $L_1$ convergence to be at best first-order accurate, and $L_\infty$ convergence to be zeroth-order accurate. These errors are even worse in the case of a linear discontinuity (usually less than first order in the $L_1$ norm), since the the number of intermediate points typically grows with time (an exception would be a scheme that employs artificial compression, for example [29]).

For scalar equations, high-order rates of convergence can be achieved if one measures convergence at a finite distance away from the location of discontinuities; for examples see [16, 26]. This is due to the fact that the characteristics of a scalar hyperbolic equation will either run in parallel to the discontinuity (linear) or travel into the discontinuity (nonlinear). So, it is expected that $O(1)$ errors near a discontinuity will not effect the solution a finite distance away from the discontinuity under mesh refinement.

For nonlinear systems of conservation laws, there will be information that passes through a shock wave, and typically there will be a loss of high-order convergence, even *away* from discontinuities [7]. These errors also show up as "noise" in the non-shock characteristic fields. This is observed in slowly moving shock waves [1, 22, 23]. This "noise" will typically limit the rate of convergence, even a finite distance away from shock waves. As pointed out in [1], there may be no way of eliminating this noise without using subcell resolution or tracking [3, 9, 14].

This paper is devoted to presenting a level-set technique [20] for tracking linear and nonlinear discontinuities for scalar conservation laws. The algorithm is based on a method of lines, finite difference framework. Since a level-set formulation is used to denote the location of the discontinuity, there is no extra front logic required (i.e., no points to track in 1-D, or curves in 2-D). All variables are located numerically on a uniform Cartesian grid and are updated with standard method of lines, essentially nonoscillatory (ENO) schemes.

The $O(1)$ errors introduced by most schemes can be attributed to interpolating across a discontinuity. It will be shown that the present algorithm is similar to Harten's subcell resolution technique [9] and Mao's treatment of discontinuities [17–19], in that this algorithm does not interpolate across a discontinuity, and the difference algorithm always "sees" a continuous solution, even near discontinuities. Harten achieves this by replacing a standard interpolation scheme, in cells that contain a discontinuity, with one that extrapolates in a conservative manner from smooth regions of the solution. Mao's method is similar but not exactly conservative. Although Harten's subcell method works quite well in one dimension, there seems to be no easy way to extend it to multidimensional problems, or to problems involving nonlinear discontinuities (shocks) [27]. Mao's work has been extended to two dimensions, but at the cost of complexity in dealing with "critical intervals" (numerical cells that contain a discontinuity). The main features of the present work are as follows. 1. Linear and nonlinear discontinuities can be treated. 2. Multidimensional problems are relatively straightforward to implement (the high-order 2-D Burgers' examples in Section 4 require only a few hundred lines of code). 3. Conservation is achieved under mesh refinement. The scheme has conservation errors of the same order as the truncation errors but still converges to the proper weak solution. 4. The zero of a level-set function is used to specify the location of the discontinuity, and thus one must know the initial location of any discontinuities to initialize the level-set function; no discontinuity detectors are used, and any shocks that form later in time, away from the zero of the level-set function, will be captured. 5. Nonconvex flux functions are not treated, since a single jump (even in a scalar equation) can result in multiple discontinuities [13].

It should be noted that the use of multiple solution states (say one "real" and one "ghost" state) is similar to most conventional front-tracking algorithms near a discontinuity.

Typically front tracking requires a cell which contains a discontinuity to have two constant states, with some geometry information to achieve subcell resolution. A notable example is presented in Charrier and Tessieras [4], where the authors formulate a 1-D scalar in terms of two scalar equations separated by a single shock, whose evolution is governed by the shock jump condition. Again, here the geometry of the front is represented by the zero of a level-set function.

The outline of the paper is as follows. The mathematical formulation is presented in Section 2. Section 3 describes the fifth-order numerical implementation. In Section 4, examples are presented to demonstrate the high rates of convergence that are achieved by the level-set tracking method, even near discontinuities. And finally, Section 5 discusses issues of efficiency.

## 2. MATHEMATICAL FORMULATION

### 2.1. *Shock Jump Condition*

Here, we present the mathematical formulation of the level-set tracking method for scalar conservation laws. We wish to solve the scalar conservation law

$$u_t + f(u)_x = 0. \tag{1}$$

Equation (1) is hyperbolic and admits discontinuous solutions. Depending on the form of $f(u)$, these discontinuities may be linear or nonlinear. Of particular interest is the formulation of a numerical method that treats the propagation of these discontinuities accurately. Shock waves, where $u$ is discontinuous, are of greatest importance. Other discontinuities, such as derivatives of $u$ in the case of rarefaction corners, are also important, but shock-capturing schemes already typically converge at second order in the $L_1$ norm and first order in the $L_\infty$ norm in these cases (an exception would be the creation of a self-similar rarefaction wave; see [5, 25]). Here, a method for dealing with linear and nonlinear discontinuities in $u$ is presented.

It is well known that a shock will travel at a speed that depends, in general, on the value of $u$ on both sides of the discontinuity as well as the flux function, $f(u)$. For a scalar conservation law, the shock speed is given by

$$s = \frac{[f(u)]}{[u]} = \frac{f(u_r) - f(u_l)}{u_r - u_l}, \tag{2}$$

where $u_r$ is the value of $u$ just to the right of the shock wave, and $u_l$ is the value of $u$ just to the left of the shock. While Eq. (2) guarantees conservation, it doesn't necessarily guarantee that the weak solution will be the proper viscosity-limiting solution. For the solution to be a shock wave, the following entropy condition must also be satisfied:

$$f'(u_l) \geq s \geq f'(u_r). \tag{3}$$

Note, the standard entropy condition, $f'(u_l) > s > f'(u_r)$, does not admit discontinuities in linear equations, whereas Eq. (3) does (see [13]). If Eq. (3) is not satisfied, the solution will be a rarefaction wave. Again, the focus of this paper is on dealing with shock waves, since they introduce the largest numerical errors. Nonconvex flux functions have a slightly more complicated entropy condition, since both shocks and rarefactions can originate from a single discontinuity (see [13] and the references therein). Issues relating to nonconvex flux functions are not addressed here.

## 2.2. *Representation*

The key idea to avoiding discretizing across discontinuities is to have two solution states, $u_1$ and $u_2$, at all locations, each of which is continuous across the discontinuity. Note that this approach is similar to the "Ghost Fluid Method" of [8], where a level-set function, $\psi$, is used to determine the location of the discontinuity. The real solution state at a point is selected to be $u_1$ or $u_2$, depending on the sign of the level-set function at that location.

Denoting the real solution state as $u$, we have

$$u = \begin{cases} u_1, & \text{if } \psi > 0 \\ u_2, & \text{if } \psi \leq 0, \end{cases} \tag{4}$$

and $\psi$ is the continuous level-set function, whose zero is located at the discontinuity. The ghost state, $u_g$, is given by

$$u_g = \begin{cases} u_1, & \text{if } \psi \leq 0 \\ u_2, & \text{if } \psi > 0. \end{cases} \tag{5}$$

Since this algorithm works with the variables $u_1$, $u_2$, and $\psi$, and never uses $u$ directly, all variables are typically continuous. To make this clear, consider representing the following discontinuous function:

$$u = \begin{cases} \cos(x), & \text{if } x \leq 0 \\ \sin(x), & \text{if } x > 0. \end{cases} \tag{6}$$

This can be represented by the following $u_1$, $u_2$, and $\psi$:

$$u_1 = \sin(x) \tag{7}$$

$$u_2 = \cos(x) \tag{8}$$

$$\psi = x. \tag{9}$$

Importantly, even though $u$ is discontinuous, $u_1$, $u_2$, and $\psi$ are all continuous. Clearly, discretizing continuous functions will yield higher order convergence than using the discontinuous function directly.

## 2.3. *Solution*

Here, we describe how to solve Eq. (1) using $u_1$, $u_2$, and $\psi$. For initial conditions, define $u_1$, $u_2$, and $\psi$ such that Eq. (4) is satisfied at the initial time. Since $u_1$ and $u_2$ can take on any value for $\psi \leq 0$ and $\psi > 0$ respectively, and still satisfy Eq. (4), this representation is not unique. It will be advantageous to extend $u_1$ and $u_2$ smoothly into their respective "ghost node regions." The nonuniqueness also extends to the level-set function. Any level-set function, $\psi$, whose zero corresponds to the discontinuity is adequate. Here, the signed distance function is usually used to initialize $\psi(t = 0)$.

Clearly, when $\psi > 0$, we need to solve

$$(u_1)_t + f(u_1)_x = 0, \tag{10}$$

since $u = u_1$ when $\psi > 0$. Likewise, when $\psi \leq 0$, we need to solve

$$(u_2)_t + f(u_2)_x = 0. \tag{11}$$

For smoothness, it is desirable to solve Eqs. (10) and (11) everywhere. But, since $u_2$ ($u_1$) will be considered a shocked state of $u_1$ ($u_2$) when $\psi > 0$ ($\psi \leq 0$), we have to check to see if $u_2$ ($u_1$) satisfies the shock entropy condition Eq. (3). Appropriate left and right states in the shock entropy condition can be cast in terms of $u_1$, $u_2$, and $\psi$ by

$$u_l = \begin{cases} u_1, & \text{if } \psi_x \leq 0 \\ u_2, & \text{if } \psi_x > 0 \end{cases} \tag{12}$$

$$u_r = \begin{cases} u_1, & \text{if } \psi_x > 0 \\ u_2, & \text{if } \psi_x \leq 0 \end{cases} \tag{13}$$

And the shock entropy condition can be recast (for a convex flux function) as

$$f'(u_l) \geq f'(u_r). \tag{14}$$

If $\psi > 0$ ($\psi \leq 0$) and the state $u_2$ ($u_1$) does not satisfy the shock entropy condition Eq. (14), we set $u_2 = u_1$ ($u_1 = u_2$). This will only affect the smoothness of $u_1$ or $u_2$ in their ghost node states and will locally reduce to a shock-capturing scheme. This will only affect the accuracy of solution when an initial discontinuity will form a self-similar rarefaction, and not a shock.

Once entropy-satisfying states $u_1$ and $u_2$ have been given, we define the shock speed by Eq. (2). This shock speed function, $s$, will be determined everywhere, and this speed is used in the level set equation to propagate the level set function, $\psi$,

$$\psi_t + s\psi_x = 0. \tag{15}$$

Importantly, at $\psi = 0$, $s$ will be the proper shock speed for the discontinuity in $u$. Also, if $u_1$ and $u_2$ are both smooth functions near $\psi = 0$, then $s$ will also be smooth. This is important numerically, since accuracy will be lost if $s$ is not smooth near $\psi = 0$.

So, in summary, we initialize $u_1$ and $u_2$ with smooth functions, and set $\psi$ to be the signed distance function from the discontinuity location (or some other smooth function whose zero corresponds to the initial location of the discontinuity). This satisfies Eq. (4) at the initial time. Then, making sure that the ghost node states satisfy the shock entropy condition (for all time), solve Eqs. (10), (11), and (15) everywhere. The real solution, $u$, can be recovered easily by using Eq. (4) anytime the solution state is required.

## 2.4. *Justification*

Clearly, the algorithm should work for a linear problem, where the characteristics are parallel to any discontinuity and are also state independent. It is not obvious that the method will work for nonlinear problems, where the shock speed is determined by both states, $u_1$ and $u_2$. In particular, how does the algorithm prevent seemingly arbitrary initial conditions in the ghost state from polluting the real state? The answer lies in applying the shock entropy condition.

Let's first consider the case of a single discontinuity in a 1-D nonlinear scalar conservation law. The equations being solved are (10), (11), and (15), with the shock-entropy condition (14) constraining the "ghost" solution state.

PROPOSITION 1. *The level set function, $\psi$, if initially monotonic, will remain monotonic.*

*Proof 1.* Taking the partial derivative of (15) w.r.t. the spatial coordinate, $x$, and replacing $\psi_x$ with $v$ yields the PDE for the slope of the level-set function:

$$v_t + s v_x + s_x v = 0. \tag{16}$$

Along a characteristic curve, $\xi = \xi_0 + \int_0^t s\, \mathrm{d}\tau$, the above PDE (16) becomes the ODE

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -s_x v, \tag{17}$$

whose solution is

$$v = v_0 \mathrm{e}^{-\int\limits_0^t s_x \mathrm{d}\tau}, \tag{18}$$

along the characteristic curve. So, if $v_0 > 0\, \forall x$ ($v_0 < 0\, \forall x$), then $v > 0\, \forall x, t$ ($v < 0\, \forall x, t$), since the exponential term on the right-hand side of (18) is always positive. This concludes the monotonicity proof.

Now let us consider the evolution of the two solution states $u_1$ and $u_2$.

PROPOSITION 2. *Information from the ghost region never reaches the real region.*

*Proof 2.*

CASE 1: $\psi_x > 0$. From the previous proof, the entropy condition then becomes $f'(u_2) \geq s \geq f'(u_1)$ at any location in $x, t$. This gives the desired property that $u_2$ characteristics will travel faster than the $\psi$ (shock) characteristics; i.e., real $u_2$ characteristics will travel from $\psi < 0$ through the shock to become characteristics in the ghost region. Likewise $u_1$ characteristics will travel slower than the shock; i.e., real $u_1$ characteristics will travel from $\psi > 0$ through the shock to become characteristics in the ghost region.

CASE 2: $\psi_x < 0$. This case is handled by the same arguments as above. This concludes the proof.

Now let us consider the case when there are multiple discontinuities, and thus a non-monotonic level-set function (i.e., the level-set function has more than one zero). There are several questions to ask when such a case arises. First, let us examine the properties of the level-set equation.

PROPOSITION 3. *The level set function will remain nonoscillatory.*

*Proof 3.*

CASE 1: $s_x$ remains bounded. This results from Eq. (18) as well. As long as the exponent in (18) is finite, one has a one-to-one mapping of the sign of the slope of the level-set function to its initial value at a paricular $\xi$ (i.e., along a characteristic). Since the definition of an oscillation is a change in sign of the slope of a function, it follows that there will be the same number of oscillations in $\psi(t > 0)$ as there were in $\psi(t = 0)$.

CASE 2: $s_x$ does not remain bounded. This case corresponds to a discontinuity in $s$. Here, it is not always possible to have a one-to-one mapping of the sign of the slope of the level-set function. But, importantly, each spatial location later in time will have a unique characteristic which originated at $t = 0$ (not every characteristic at $t = 0$ will exist forever, since its information can be lost in a shock wave). Therefore, at later times, there will be at most the same number of oscillations that existed at $t = 0$. Therefore, the level-set function will remain nonoscillatory (i.e., no new oscllations in $\psi$ will appear for $t > 0$).

PROPOSITION 4. *Near an isolated discontinuity, the information in the ghost region never reaches the real region.*

*Proof 4.* The proof is basically the same as in Proof 2, except that the entropy condition is now only locally in a region of $x, t$ near a discontinuity. Again, either Case 1 or 2 from Proof 2 will hold for all $x, t$.

This concludes the section on justification. Before moving on, the issue of shock collisions will be addressed. Clearly, by tracking multiple discontinuities, there exists the possibility that two discontinuities will intersect. See Section 4.2 for an example. The above propositions hold true, so long as there is some finite distance between tracked discontinuities. For scalar equations, when two discontinuies (shocks) collide, they coalesce into a single discontinuity.

For two discontinuities to be represented with a single level-set function, the level-set function would have two zeroes. Say the first shock corresponds to $x_1(t)$ and the second to $x_2(t)$. For example one might have $\psi > 0$ for $x < x_1$, $\psi < 0$ for $x_1 < x < x_2$, and finally $\psi > 0$ for $x > x_2$. If the initial conditions are such that $x_1(t_i) = x_2(t_i)$ (i.e., the two shocks collide at $t = t_i$), then what does the scheme do after $t_i$? First let us examine what happens to the level-set function. At time $t_i$, the region $\psi < 0$, corresponding to the state between the two shocks, vanishes. After $t_i$, the real $u$ will be equal to $u_1$ for all $x$. After $t_i$, there will be captured shock in $u_1$, and this final shock will not be tracked at high order, but rather captured. So, in conclusion for shock interactions, the algorithm will achieve the correct solution, but after the shock collision time, the solution will be as accurate as the underlying capturing scheme.

## 2.5. *Extension to Two Dimensions*

The above formulation extends easily to multidimensional problems. In particular, the scalar conservation law in two dimensions is

$$u_t + f(u)_x + g(u)_y = 0. \tag{19}$$

Again, Eq. (4) can be used to represent the solution. The states $u_1$ and $u_2$ are evolved according to

$$(u_1)_t + f(u_1)_x + g(u_1)_y = 0 \tag{20}$$

$$(u_2)_t + f(u_2)_x + g(u_2)_y = 0; \tag{21}$$

and the level-set function, $\psi$, according to

$$\psi_t + \mathbf{s} \cdot \nabla\psi = 0, \tag{22}$$

where

$$\mathbf{s} = \frac{[f(u)]}{[u]}\hat{\imath} + \frac{[g(u)]}{[u]}\hat{\jmath}. \tag{23}$$

Again, one needs to make sure that the ghost node state satisfies the shock-entropy condition. In multiple dimensions, for a convex flux function, it is sufficient to check if the characteristics flow into the shock. In two dimensions, the characteristic velocity is given by

$$\mathbf{c} = f'(u)\hat{\imath} + g'(u)\hat{\jmath}. \tag{24}$$

Also, the orientation of a shock will be given by

$$\hat{n} = \frac{\nabla\psi}{|\nabla\psi|}. \tag{25}$$

So, the characteristic speeds for the states $u_1$ and $u_2$ in the normal direction are given by

$$c_1 = \hat{n} \cdot (f'(u_1)\hat{\imath} + g'(u_1)\hat{\jmath}) \tag{26}$$

$$c_2 = \hat{n} \cdot (f'(u_2)\hat{\imath} + g'(u_2)\hat{\jmath}). \tag{27}$$

For the states $u_1$ and $u_2$ to satisfy the shock-entropy condition, the following must be true:

$$c_2 \geq c_1. \tag{28}$$

In one dimension, Eq. (28) is equivalent to Eq. (3).

## 3. DISCRETIZATION

Here, one particular discretization is presented. Numerical results are given in the next section. Recall that Eqs. (10), (11), and (15), along with the shock-entropy condition need be discretized. Notice that Eqs. (10) and (11) are scalar conservation laws, Eq. (15) is a Hamilton–Jacobi-like partial differential equation, and the entropy condition is an algebraic constraint on the ghost state.

### 3.1. *Grid*

A uniform Cartesian grid is used to discretize the domain $x \in (x_{min}, x_{max})$, with $N_x + 1$ equally spaced nodes. The numerical solution of $u_1$ is denoted by $u_1(i, n)$, where $i$ is the spatial node number corresponding to the location $x_i = x_{min} + i\Delta x$, where $\Delta x = (x_{max} - x_{min})/N_x$. And $n$ is the time level corresponding to $t_n = n\Delta t$, where $\Delta t = t_{final}/N_t$. The states $u_2$ and $\psi$ are denoted similarly.

## 3.2. *Time Integration*

An explicit method of lines approach is taken. The results presented here use a third-order TVD Runge–Kutta time integrator [26]. For example, the solution to $u_1(i, n)$ will be advanced from $t = t_n$ to $t = t_{n+1}$ via

$$\begin{aligned} u_1(i,*) &= u_1(i,n) + \Delta t L(u_1(i,n)) \\ u_1(i,**) &= \frac{3}{4}u_1(i,n) + \frac{1}{4}\Delta t L(u_1(i,*)) \\ u_1(i,n+1) &= \frac{1}{3}u_1(i,n) + \frac{2}{3}u_1(i,**) + \frac{2}{3}\Delta t L(u_1(i,**)). \end{aligned} \tag{29}$$

Here, the $*$ and $**$ represent intermediate stages of the Runge–Kutta integration. The updates for $u_2$ and $\psi$ are similar. Note that the $L()$ operator corresponds to the spatial flux differences for $u_1$ and $u_2$, or derivatives of $\psi$. These are described in the next section.

## 3.3. *Spatial Discretization*

*3.3.1. Project ghost nodes into entropy-satisfying state.* As described in Section 2.3, it is necessary to make sure that the ghost node state be an entropy-satisfying state. This is done at the beginning of every Runge–Kutta cycle. As stated in Section 2.3, the ghost node state is only set to the real state if Eq. (14) in one dimension, or (28) in two dimensions, is not satisfied. If the entropy condition is satisfied, then the ghost node state is not modified. The entropy condition is only an algebraic constraint on the ghost node state but requires knowledge of $\psi_x$ (and $\psi_y$ in two dimensions). These spatial derivatives are obtained by averaging Eq. (51) and Eq. (52).

*3.3.2. Conservative discretization for $u_1$ and $u_2$.* For $L(u_1(i, n))$ and $L(u_2(i, n))$ we use the fifth-order weighted ENO (WENO) scheme of [12], with a local Lax–Friedrichs solver. This scheme is a conservative flux difference method, which has been shown to be stable, and yields the proper viscosity-vanishing solution to Eq. (1). The operator $L(u_1(i, n))$ is given by

$$L(u_1(i,n)) = -\left(\hat{f}_{i+1/2} - \hat{f}_{i-1/2}\right)/\Delta x, \tag{30}$$

where $\hat{f}_{i+1/2}$ and $\hat{f}_{i-1/2}$ are numerical approximations to the flux function, $f(u)$. In particular, for the local Lax–Friedrichs scheme, we take

$$\hat{f}_{i+1/2} = \hat{f}_i^+ + \hat{f}_{i+1}^-, \tag{31}$$

where

$$\hat{f}_i^+ = WENO5(f_{i-2}^+, f_{i-1}^+, f_i^+, f_{i+1}^+, f_{i+2}^+) \tag{32}$$

$$\hat{f}_{i+1}^- = WENO5(f_{i+3}^-, f_{i+2}^-, f_{i+1}^-, f_i^-, f_{i-1}^-) \tag{33}$$

and

$$f_i^+ = \frac{1}{2}(f(u_1(i,n)) + \alpha u_1(i,n)) \tag{34}$$

$$f_i^- = \frac{1}{2}(f(u_1(i,n)) - \alpha u_1(i,n)) \tag{35}$$

and

$$\alpha = \max(|f'(u_1(i,n))|, |f'(u_1(i+1,n))|). \tag{36}$$

The function $WENO5(a, b, c, d, e)$ is defined next. First, define three interpolated values

$$q_1 = \frac{a}{3} - \frac{7b}{6} + \frac{11c}{6} \tag{37}$$

$$q_2 = -\frac{b}{6} + \frac{5c}{6} + \frac{d}{3} \tag{38}$$

$$q_3 = \frac{c}{3} + \frac{5d}{6} - \frac{e}{6} \tag{39}$$

and three indicators of smoothness

$$IS_1 = 13(a - 2b + c)^2 + 3(a - 4b + 3c)^2 \tag{40}$$

$$IS_2 = 13(b - 2c + d)^2 + 3(d - b)^2 \tag{41}$$

$$IS_3 = 13(c - 2d + e)^2 + 3(3c - 4d + e)^2 \tag{42}$$

and take

$$\alpha_1 = \frac{1}{(\epsilon + IS_1)^2} \tag{43}$$

$$\alpha_2 = \frac{6}{(\epsilon + IS_2)^2} \tag{44}$$

$$\alpha_3 = \frac{3}{(\epsilon + IS_3)^2} \tag{45}$$

and finally

$$WENO5(a, b, c, d, e) = \frac{\alpha_1 q_1 + \alpha_2 q_2 + \alpha_3 q_3}{\alpha_1 + \alpha_2 + \alpha_3}. \tag{46}$$

In all computations presented here, $\epsilon = 10^{-6}$, as suggested in [11, 12, 16]. The operator for $u_2$ is the same, with $u_2$ replacing $u_1$. Note that, for the linear advection equation, the local Lax–Friedrichs scheme is equivalent to the standard upwind discretization.

*3.3.3. Level-set discretization.* For the level-set Eq. (15), we have the operator

$$L(\psi) = -s\hat{\psi}_x, \tag{47}$$

where $s$ is the shock speed and $\hat{\psi}_x$ is the numerical approximation to $\psi_x$. First, the shock speed at each node location, $s(i, n)$, is determined from $u_1$ and $u_2$ as

$$s(i,n) = \frac{f(u_1(i,n)) - f(u_2(i,n))}{u_1(i,n) - u_2(i,n)}. \tag{48}$$

The $\psi_x$ derivative is approximated using a fifth-order WENO scheme for Hamilton–Jacobi Eq. [11] using the following local Lax–Friedrichs method. Define the first-order difference operators as

$$D_i^- = \frac{\psi(i,n) - \psi(i-1,n)}{\Delta x} \tag{49}$$

$$D_i^+ = \frac{\psi(i+1,n) - \psi(i,n)}{\Delta x}. \tag{50}$$

The numerical approximations to the spatial derivative, $\hat{\psi}_x$ are then given by

$$\hat{\psi}_x^- = WENO5(D_{i-2}^-, D_{i-1}^-, D_i^-, D_{i+1}^-, D_{i+2}^-) \tag{51}$$

$$\hat{\psi}_x^+ = WENO5(D_{i+2}^+, D_{i+1}^+, D_i^+, D_{i-1}^+, D_{i-2}^+). \tag{52}$$

The approximation to $L(\psi)$ at each grid point is given by

$$L(\psi(i,n)) = \frac{1}{2}(s(i,n)(\hat{\psi}_x^- + \hat{\psi}_x^+) - \alpha(\hat{\psi}_x^+ - \hat{\psi}_x^-)), \tag{53}$$

where

$$\alpha = \max(|s(i-1,n)|, |s(i,n)|, |s(i+1,n)|). \tag{54}$$

## 4. EXAMPLES

Here, we present solutions to various scalar hyperbolic conservation laws using both the level-set tracking algorithm and the traditional shock-capturing algorithm. Notice that the discretization presented in Section 3 will have a truncation error of $O(\Delta t^3) + O(\Delta x^5)$ in smooth regions. All computations are performed with $\Delta t \propto \Delta x^{5/3}$, which effectively yields a truncation error of $O(\Delta x^5)$ in smooth regions. For all cases, the number of time steps, $N_t$, and number of spatial nodes, $N_x$, are noted in the tables and figures.

### 4.1. *Linear Advection Equation*

The algorithm presented in Sections 2 and 3 is tested on a standard test problem [9, 10, 27]. The equation to be solved is the linear advection equation

$$u_t + u_x = 0, \tag{55}$$

with periodic boundary conditions at $x = \pm 1$ and subject to the initial conditions

$$u = \begin{cases} 2(x+1) - \frac{1}{6}\sin\left(3\pi\left(x+\frac{3}{2}\right)\right), & -1 < x < -\frac{1}{2} \\ -\left(x-\frac{1}{2}\right)\sin\left(\frac{3}{2}\pi\left(x-\frac{1}{2}\right)^2\right), & -\frac{1}{2} < x < \frac{1}{6} \\ \sin\left(2\pi\left(\frac{1}{2}-x\right)\right), & \frac{1}{6} < x < \frac{1}{2} \\ \sin\left(2\pi\left(x-\frac{1}{2}\right)\right), & \frac{1}{2} < x < \frac{5}{6} \\ 2(x-1) - \frac{1}{6}\sin\left(3\pi\left(x-\frac{1}{2}\right)\right), & \frac{5}{6} < x < 1. \end{cases} \tag{56}$$

Notice that this initial condition has discontinuities at $x = -\frac{1}{2}$, $x = \frac{1}{6}$, and $x = \frac{5}{6}$. Also, there is a discontinuity in derivative at $x = \frac{1}{2}$ (see Fig. 1). This function is represented by Eq. (4) with

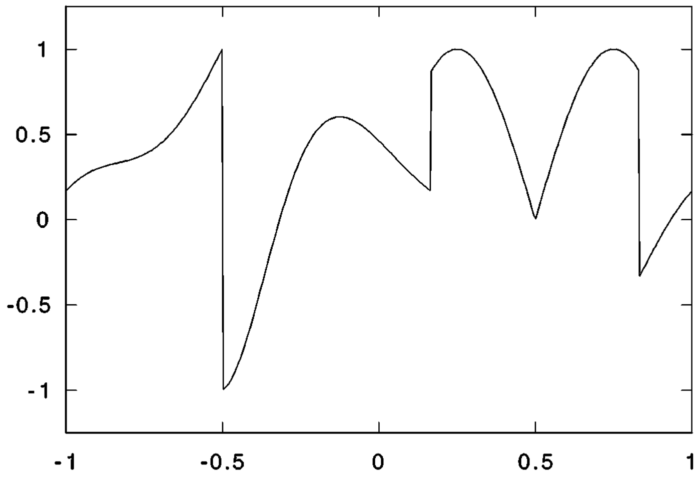

**FIG. 1.** Plot of initial conditions corresponding to Eq. (56).

$$u_1 = \begin{cases} 2(x+1) - \frac{1}{6}\sin\left(3\pi\left(x+\frac{3}{2}\right)\right), & -1 < x < -\frac{1}{6} \\ \sin\left(2\pi\left(\frac{1}{2}-x\right)\right), & -\frac{1}{6} < x < \frac{2}{3} \\ 2(x-1) - \frac{1}{6}\sin\left(3\pi\left(x-\frac{1}{2}\right)\right), & \frac{2}{3} < x < 1 \end{cases} \tag{57}$$

$$u_2 = \begin{cases} \sin\left(2\pi\left(x-\frac{1}{2}\right)\right), & -1 < x < -\frac{5}{6} \\ -\left(x-\frac{1}{2}\right)\sin\left(\frac{3}{2}\pi\left(x-\frac{1}{2}\right)^2\right), & -\frac{5}{6} < x < \frac{1}{3} \\ \sin\left(2\pi\left(x-\frac{1}{2}\right)\right), & \frac{1}{3} < x < 1 \end{cases} \tag{58}$$

$$\psi = \begin{cases} x + \frac{7}{6}, & -1 < x < -\frac{5}{6} \\ -x - \frac{1}{2}, & -\frac{5}{6} < x < -\frac{1}{6} \\ x - \frac{1}{6}, & -\frac{1}{6} < x < \frac{1}{3} \\ -x + \frac{1}{2}, & \frac{1}{3} < x < \frac{2}{3} \\ x - \frac{5}{6}, & \frac{2}{3} < x < 1. \end{cases} \tag{59}$$

Although $u_1$ and $u_2$ are also discontinuous and $\psi$ is discontinuous in derivative, they are all $C_\infty$ at the true discontinuity locations of $u$. And since most shock-capturing schemes will be convergent away from the discontinuities (at least in a scalar problem), it is expected that using Eq. (4) and operating on $u_1$, $u_2$, and $\psi$ will be more accurate than discretizing $u$ directly. This is confirmed numerically. At $t = 2$ with $N_t$ time steps, the error in the numerical solution using the discrete $L_1$ and $L_\infty$ norms is measured. These norms measure the pointwise convergence of the numerical solution to the exact solution (all points in the numerical solution are included, not just the points away from discontinuities). The errors are denoted by $E_1$ and $E_\infty$, and the order at which they converge are denoted by $R_1$ and $R_\infty$, respectively. Also, a subscript $LST$ indicates the level-set tracking algorithm, and $SC$ indicates the WENO5 shock-capturing algorithm (see Table I). Notice that the

**TABLE I**
**Numerical Accuracy for 1D Linear Advection**

| $N_x$ | $N_t$ | $E_{1\text{-}LST}$ | $R_{1\text{-}LST}$ | $E_{\infty\text{-}LST}$ | $R_{\infty\text{-}LST}$ | $E_{1\text{-}SC}$ | $R_{1\text{-}SC}$ |
|---|---|---|---|---|---|---|---|
| 61 | 75 | 7.24e-3 | | 3.46e-2 | | 2.37e-1 | |
| 121 | 235 | 3.32e-4 | 4.52 | 1.64e-3 | 4.40 | 1.18e-1 | 1.01 |
| 241 | 740 | 1.04e-5 | 4.99 | 6.58e-5 | 4.64 | 6.31e-2 | 0.90 |
| 481 | 2340 | 2.91e-7 | 5.16 | 2.90e-6 | 4.51 | 3.47e-2 | 0.86 |
| 961 | 7425 | 9.58e-9 | 4.93 | 1.54e-7 | 4.23 | 1.93e-2 | 0.85 |
| 1921 | 23555 | 2.36e-10 | 5.34 | 3.51e-9 | 5.46 | 1.08e-2 | 0.84 |

level-set tracking algorithm converges at fifth order in both the $L_1$ and $L_\infty$ norms. The standard shock-capturing algorithm converges at roughly $5/6$ order in the $L_1$ norm. This is commensurate with the notion that a captured linear discontinuity will smear at a rate proportional to $N_t^{1/(r+1)}$, where $r$ is the order of the scheme [6]. The $L_\infty$ error for the capturing was roughly constant and equal to $1/2$ the maximum jump in $u$, as expected. Figure 2 shows the solution of the WENO5 shock-capturing algorithm, and Fig. 3 shows the solution using the WENO5 level-set tracking algorithm. Notice that this is the roughly the same resolution used in [9], with comparable results. Importantly, the level-set tracking algorithm maintains a perfect discontinuity in $u$, while achieving high-order pointwise convergence.

## 4.2. *Burgers' Equation*

Here, the level-set tracking algorithm is tested on the nonlinear Burgers' equation

$$u_t + \left(\frac{u^2}{2}\right)_x = 0 \tag{60}$$

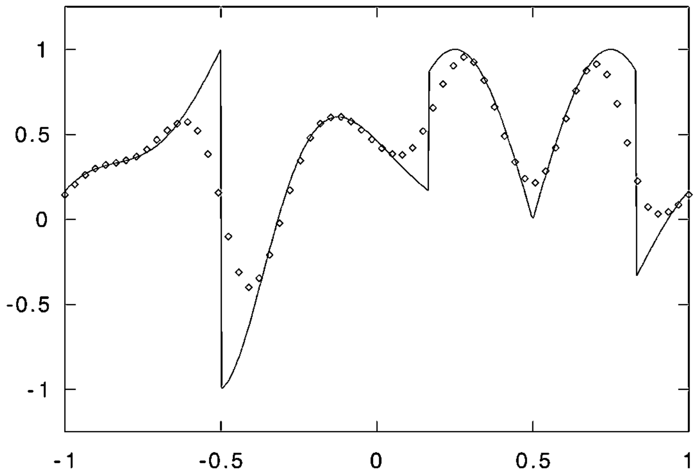

**FIG. 2.** Plot of WENO5 shock-capturing solution at $t = 2$ with $N_x = 61$ ($\diamond$) and exact solution (solid line).

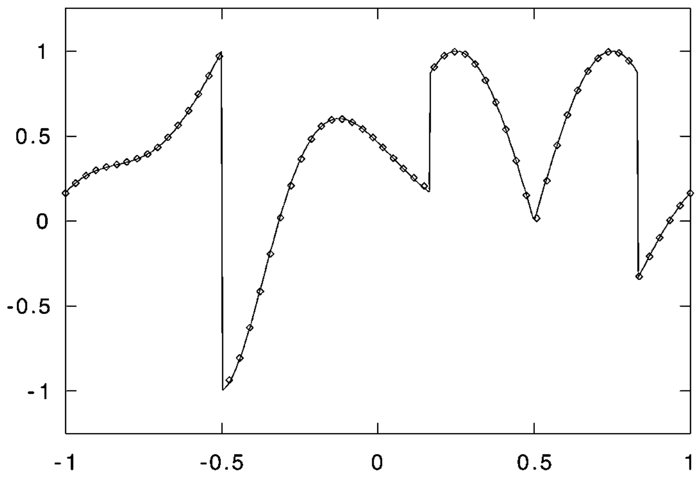

**FIG. 3.** Plot of WENO5 level-set tracking solution at $t = 2$ with $N_x = 61$ ($\diamond$) and exact solution (solid line).

with periodic boundary conditions at $x = 0$ and $x = 1$ subject to the initial conditions

$$u = \begin{cases} \frac{1}{2}(1 + \cos(2\pi x)), & \frac{1}{3} < x < \frac{2}{3} \\ \frac{1}{2} + \sin(2\pi x), & \text{otherwise.} \end{cases} \tag{61}$$

Notice that this initial condition has discontinuities at $x = \frac{1}{3}$ and $x = \frac{2}{3}$ (see Fig. 4). This function is represented by Eq. (4) with

$$u_1 = \frac{1}{2} + \sin(2\pi x) \tag{62}$$

$$u_2 = \frac{1}{2}(1 + \cos(2\pi x)) \tag{63}$$

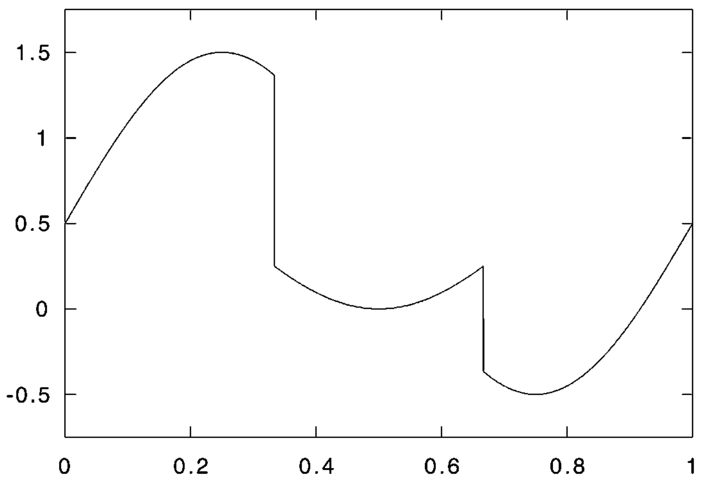

**FIG. 4.** Plot of initial conditions corresponding to Eq. (61).

**TABLE II**
**Numerical Accuracy for Burgers' Equation**

| $N_x$ | $N_t$ | $E_{1\text{-}LST}$ | $R_{1\text{-}LST}$ | $E_{\infty\text{-}LST}$ | $R_{\infty\text{-}LST}$ | $E_{1\text{-}SC}$ | $R_{1\text{-}SC}$ |
|---|---|---|---|---|---|---|---|
| 40 | 15 | 1.84e-4 | | 1.09e-3 | | 1.48e-2 | |
| 80 | 50 | 8.16e-6 | 4.49 | 6.70e-5 | 4.03 | 7.84e-3 | 0.92 |
| 160 | 150 | 1.67e-7 | 5.61 | 9.15e-6 | 2.87 | 3.43e-3 | 1.19 |
| 320 | 480 | 9.40e-9 | 4.15 | 6.53e-7 | 3.81 | 1.47e-3 | 1.22 |
| 640 | 1525 | 2.56e-10 | 5.20 | 2.48e-8 | 4.72 | 7.63e-4 | 0.95 |
| 1280 | 4840 | 4.72e-12 | 5.76 | 2.70e-10 | 6.52 | 2.76e-4 | 1.47 |

$$\psi = \begin{cases} \frac{1}{3} - x, & 0 \le x < \frac{1}{2} \\ x - \frac{2}{3}, & \frac{1}{2} \le x < 1. \end{cases} \tag{64}$$

Again, $u_1$ and $u_2$ are also discontinuous and $\psi$ is discontinuous in derivative, but they are all $C_\infty$ at the true discontinuity locations of $u$. The $L_1$ and $L_\infty$ errors and rates of convergence, $R_1$ and $R_\infty$, are measured at $t = 0.2$. These are measured by comparing with a numerical solution using twice as fine of a grid; thus we measure the pointwise self convergence of the solution. Fifth-order convergence in both the $L_1$ and $L_\infty$ norm is achieved for the level-set tracking algorithm, while first-order convergence is achieved in the $L_1$ norm for the capturing scheme (see Table II). Line plots at $t = 0.2$ are shown for both the capturing scheme (Fig. 5) and the level-set tracking scheme (Fig. 6).

It is interesting to run the problem further in time, since at $t \approx 0.369$ the two shocks collide. At this time, the $\psi < 0$ region disappears, and the algorithm then captures the remaining shock. See Fig. 7 for a contour plot of $\psi(x, t) = 0$. Even in this case, the level-set tracking scheme achieves the correct solution, but after $t \approx 0.369$, it reduces back to a capturing scheme (see Fig. 8). It would be interesting to see if using two level sets and three solution states could accurately track the merging of two tracked shocks into one.

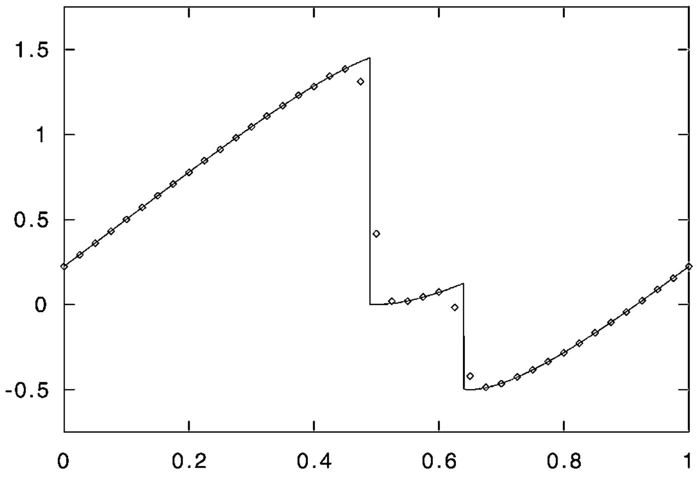

**FIG. 5.** Plot of WENO5 shock-capturing solution at $t = 0.2$ with $N_x = 40$ (◇) and converged level-set tracking solution with $N_x = 2560$ (solid line).

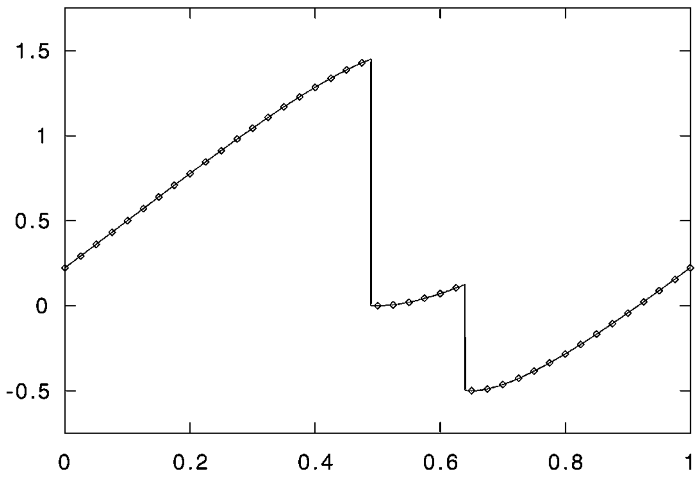

**FIG. 6.** Plot of WENO5 level-set tracking solution at $t = 0.2$ with $N_x = 40$ ($\diamond$) and converged level-set tracking solution with $N_x = 2560$ (solid line).

### 4.3. *Two-Dimensional Burgers' Equation*

Here, a nonlinear two-dimensional test is conducted. The equation to be solved is the two-dimensional Burgers' equation

$$u_t + \left(\frac{u^2}{2}\right)_x + \left(\frac{u^2}{2}\right)_y = 0, \tag{65}$$

with periodic boundary conditions at $x = 0$, $x = 1$, $y = 0$, and $y = 1$, subject to the initial conditions

$$u = \begin{cases} \frac{1}{2}(1 + \cos(2\pi(x + a))), & \text{if } \frac{1}{3} < x + a < \frac{2}{3} \\ \frac{1}{2} + \sin(2\pi(x + a)), & \text{otherwise,} \end{cases} \tag{66}$$

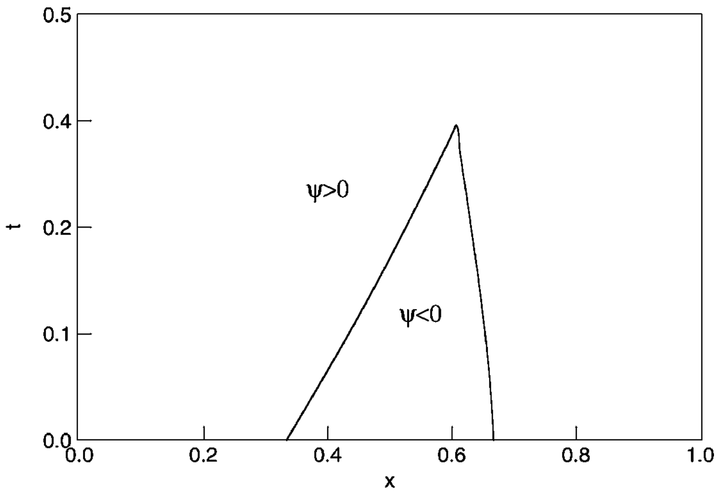

**FIG. 7.** Contour plot of $\psi(x, t) = 0$ from the WENO5 level-set tracking solution with $N_x = 160$.

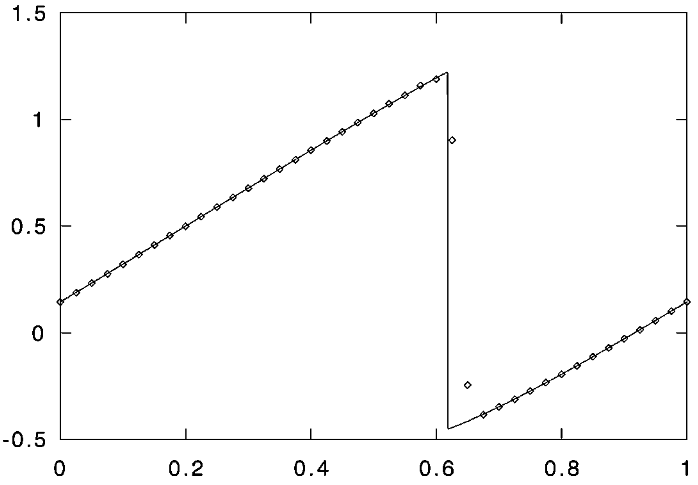

**FIG. 8.** Plot of WENO5 level-set tracking solution at $t = 0.4$ with $N_x = 40$ ($\diamond$) and converged level-set tracking solution with $N_x = 2560$ (solid line).

where

$$a = 0.1\sin(2\pi y). \tag{67}$$

Notice that this initial condition has discontinuities along the curves $x = \frac{1}{3} - 0.1\sin(2\pi y)$ and $x = \frac{2}{3} - 0.1\sin(2\pi y)$. See Fig. 9.

This function is represented by Eq. (4) with

$$u_1 = \frac{1}{2} + \sin(2\pi(x + a)) \tag{68}$$

$$u_2 = \frac{1}{2}(1 + \cos(2\pi(x + a))) \tag{69}$$

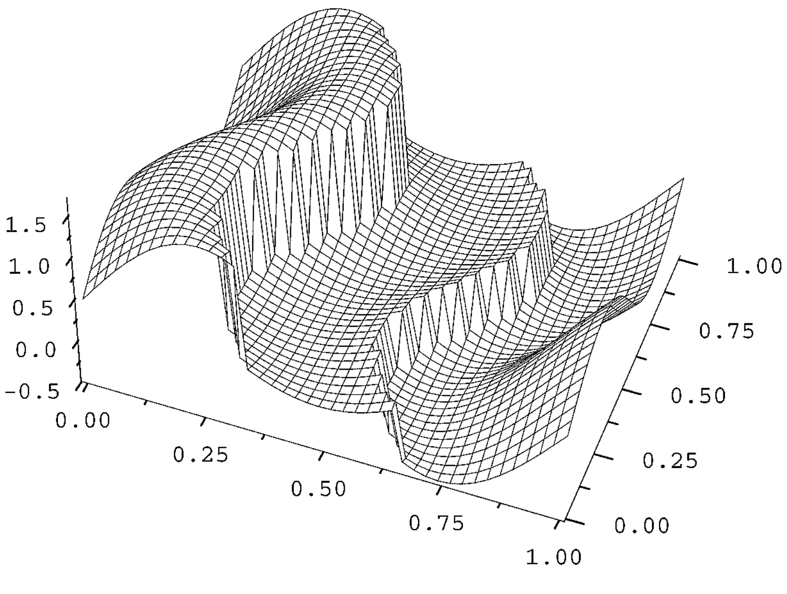

**FIG. 9.** Surface plot of initial conditions corresponding to Eq. (66).

**TABLE III**
**Numerical Accuracy for 2D Burgers' Equation**

| $N_x = N_y$ | $N_t$ | $E_{1\text{-}LST}$ | $R_{1\text{-}LST}$ | $E_{\infty\text{-}LST}$ | $R_{\infty\text{-}LST}$ | $E_{1\text{-}SC}$ | $R_{1\text{-}SC}$ |
|---|---|---|---|---|---|---|---|
| 20 | 5 | 1.83e-3 | | 8.15e-3 | | 2.89e-2 | |
| 40 | 15 | 1.77e-4 | 3.37 | 1.55e-3 | 2.39 | 1.36e-2 | 1.09 |
| 80 | 50 | 9.72e-6 | 4.18 | 1.43e-4 | 3.44 | 6.34e-3 | 1.10 |
| 160 | 150 | 1.73e-7 | 5.81 | 5.41e-6 | 4.72 | 3.08e-3 | 1.04 |

$$\psi = \begin{cases} \frac{1}{3} - (x+a), & \text{if } x + a < \frac{1}{2} \\ x + a - \frac{2}{3}, & \text{otherwise.} \end{cases} \tag{70}$$

At $t = 0.1$ with $N_t$ time steps, the error in the numerical solution using the discrete $L_1$ and $L_\infty$ norms is measured (see Table III). Notice that the level-set tracking algorithm converges at fifth order in both the $L_1$ and $L_\infty$ norms. Again, the standard shock-capturing algorithm converges at first order in the $L_1$ norm. Figure 10 shows a surface plot of the solution of the WENO5 shock-capturing algorithm, and Fig. 11 shows the solution using the WENO5 level-set tracking algorithm. Figure 12 shows a line plot of the shock-capturing solution at $t = 0.1$ at $y = 1/2$. Figure 13 shows a line plot of the level-set tracking solution at $t = 0.1$ at $y = 1/2$. As in Example 4.2, if this problem is solved later in time, the two shocks merge.

## 4.4. *Two-Dimensional Burgers' Equation*

Here, another two-dimensional Burgers' test is conducted, again with periodic boundary conditions at $x = 0$, $x = 1$, $y = 0$, and $y = 1$. The initial conditions are

$$u = \begin{cases} 1, & \text{if } \left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2 < \left(\frac{1}{3}\right)^2 \\ 0, & \text{otherwise.} \end{cases} \tag{71}$$

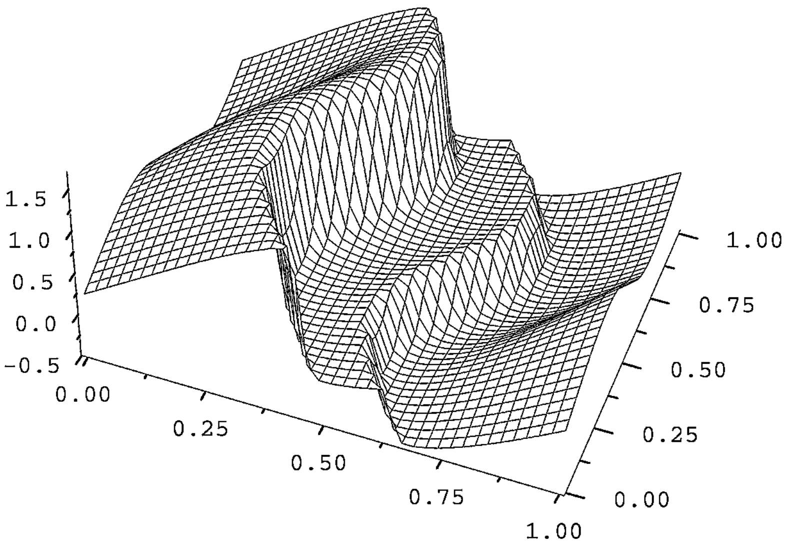

**FIG. 10.** Surface plot of WENO5 shock-capturing solution at $t = 0.1$ with $N_x = N_y = 40$.

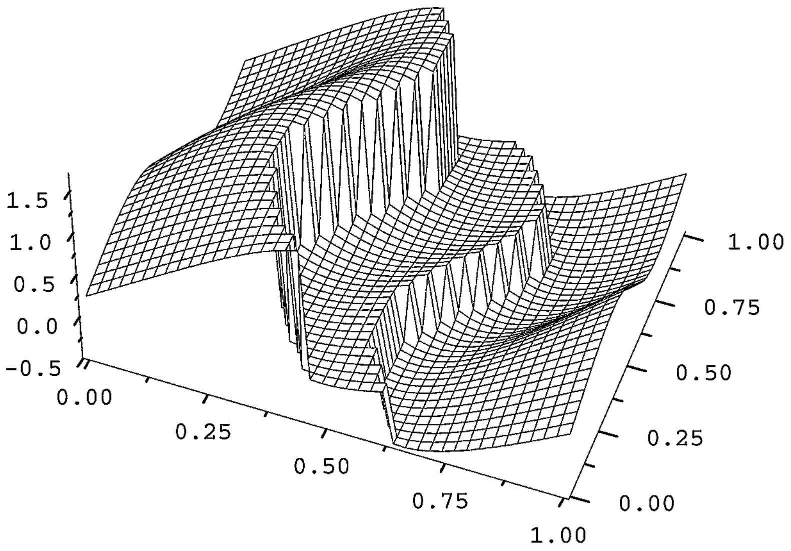

**FIG. 11.** Surface plot of WENO5 level-set tracking solution at $t = 0.1$ with $N_x = N_y = 40$.

Notice that this initial condition is discontinuous along a circle of radius $\frac{1}{3}$ centered at $(\frac{1}{2}, \frac{1}{2})$. See Fig. 14.

This function is represented by Eq. (4) with

$$u_1 = 1 \tag{72}$$

$$u_2 = 0 \tag{73}$$

$$\psi = \frac{1}{3} - \left[\left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2\right]^{\frac{1}{2}}. \tag{74}$$

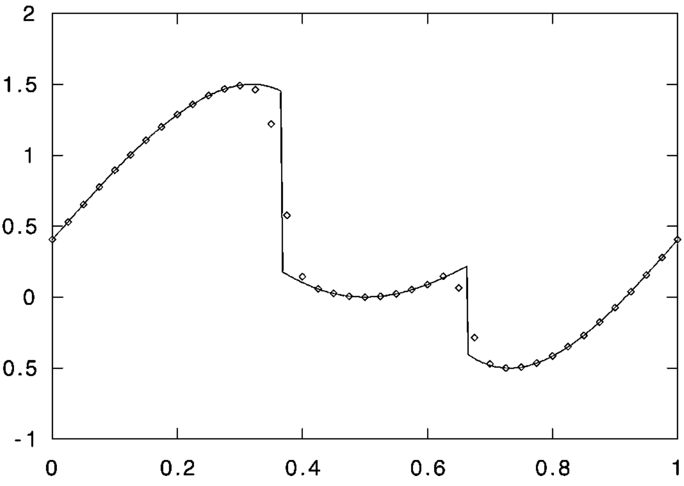

**FIG. 12.** Plot of WENO5 shock-capturing solution at $y = 0.5$, $t = 0.1$ with $N_x = N_y = 40$ (◇) and converged level-set tracking solution with $N_x = N_y = 320$ (solid line).

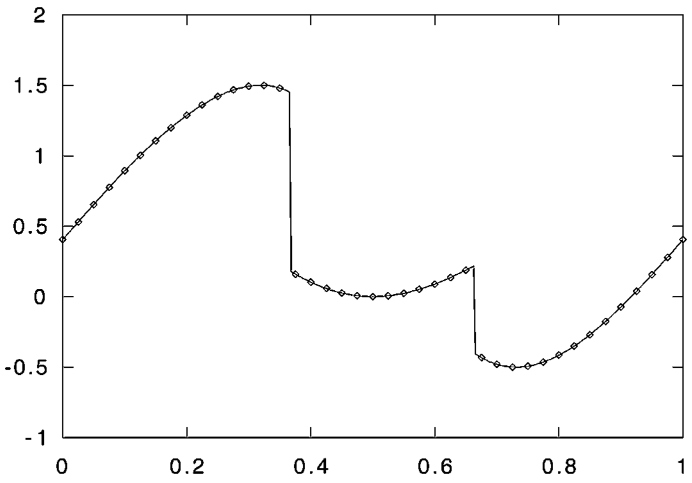

**FIG. 13.** Plot of WENO5 level-set tracking solution at $y = 0.5$, $t = 0.1$ with $N_x = N_y = 40$ ($\diamond$) and converged level-set tracking solution with $N_x = N_y = 320$ (solid line).

Note that the 2-D Burgers' Eq. (65) can be rewritten in terms of the rotated coordinates $\eta = \frac{x+y}{\sqrt{2}}$ and $\xi = \frac{y-x}{\sqrt{2}}$ as

$$u_t + \left(\frac{u^2}{\sqrt{2}}\right)_\eta = 0, \tag{75}$$

which is simply the 1-D Burgers' equation scaled by $\sqrt{2}$. So, the initial conditions (71) will look like "top-hat" functions in terms of $\eta$; see Fig. 15 (for $\xi = \frac{-3}{8\sqrt{2}}$). And, the exact solution will consist of shock wave at the right discontinuity and a self-similar rarefaction on the left discontinuity. The rarefaction eventually catches up to the shock wave and subsequently modifies the shock speed.

The numerical solution is integrated to $t = 0.2$ with $N_t = 30$ and $N_x = N_y = 40$. Surface plots for the WENO5 shock-capturing and level-set tracking algorithms are given in Figs. 16

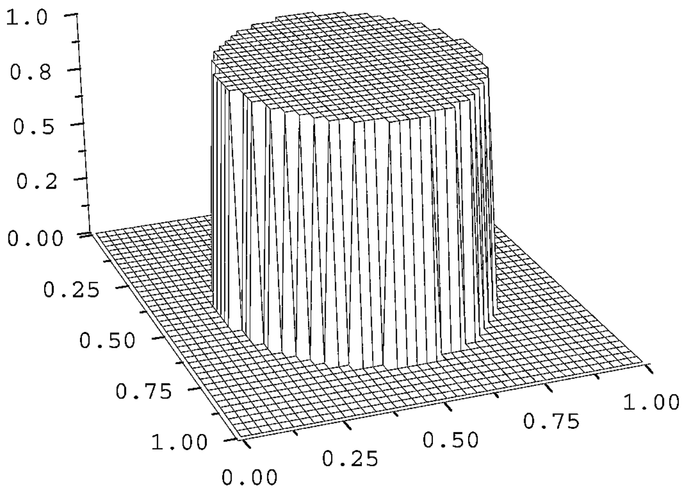

**FIG. 14.** Surface plot of initial conditions corresponding to Eq. (71).

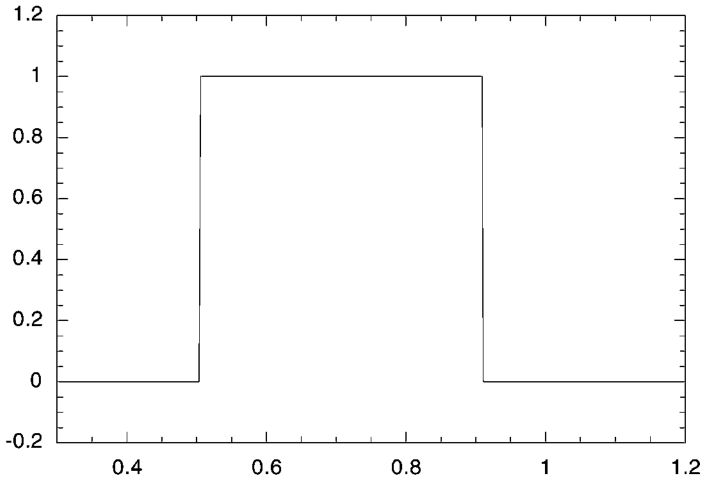

**FIG. 15.** Plot of initial conditions $u(\eta, \xi = \frac{-3}{8\sqrt{2}}, t = 0)$ corresponding to Eq. (71).

and 17, respectively. Notice that the tracking algorithm has a sharp jump in $u$ on the "front" of the circle where a shock forms, while the capturing scheme has a couple of intermediate points along the shock. Also, the rarefaction portion of the solutions are nearly identical between the two algorithms. This is expected, since enforcing the entropy condition at $t = 0$ gives $u_1 = u_2$ for $\eta < \frac{1}{\sqrt{2}}$ in the tracking scheme. So, in this region, the level curve $\psi = 0$ plays little role, since $u_1 = u_2$ (which is also the same as $u$ in the capturing scheme). Line plots of $u(\eta, \xi = \frac{-3}{8\sqrt{2}}, t = 0.2)$ are given in Figs. 18 and 19 for each method, along with the exact solution. Notice that the level-set tracking algorithm is very accurate near the shock front. The level-set function, $\psi(\eta, \xi = \frac{-3}{8\sqrt{2}}, t = 0.2)$, is plotted in Fig. 20. Notice that there are two zeroes, one corresponding to the right-going shock, and one corresponding to the rarefaction fan. But, again, the zero near $\xi = 2/3$ plays little role, since $u_1 = u_2$ in this region.

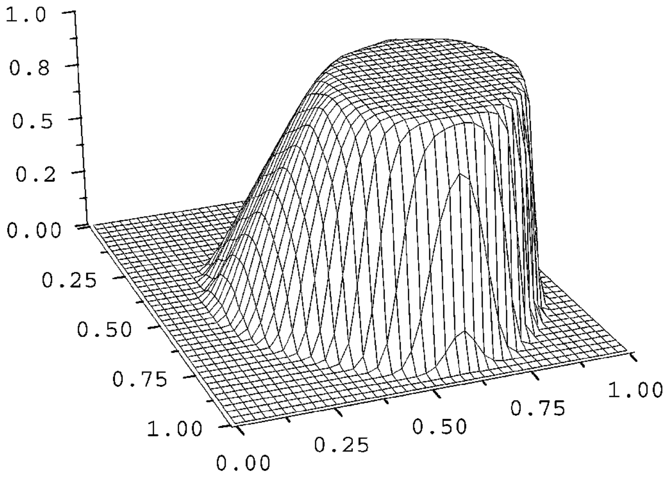

**FIG. 16.** Surface plot of WENO5 shock-capturing solution at $t = 0.2$ with $N_x = N_y = 40$.

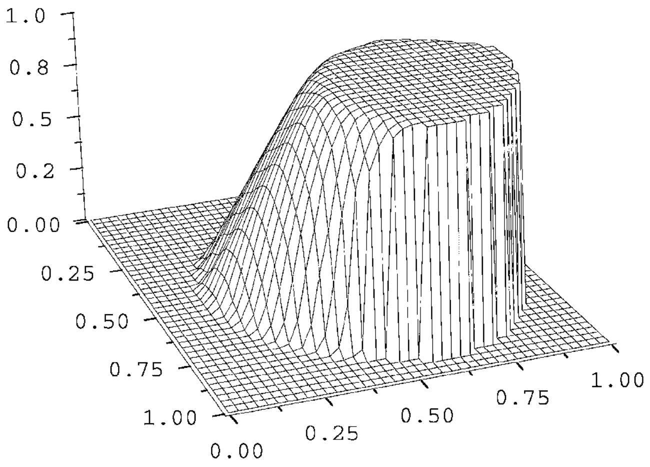

**FIG. 17.** Surface plot of WENO5 level-set tracking solution at $t = 0.2$ with $N_x = N_y = 40$.

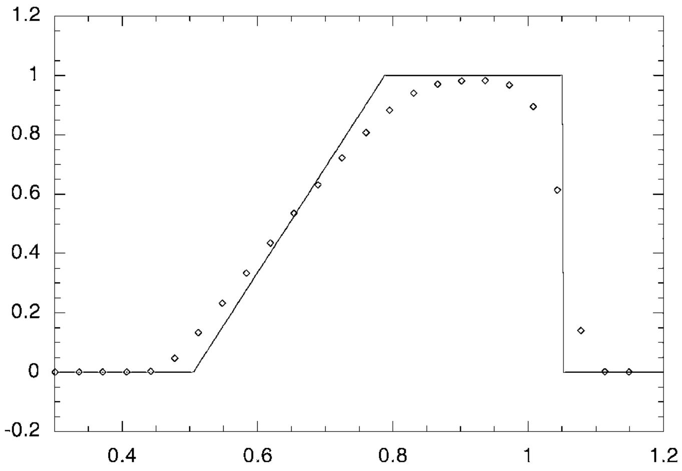

**FIG. 18.** Plot of WENO5 shock-capturing solution at $\xi = \frac{-3}{8\sqrt{2}}$ and $t = 0.2$ with $N_x = N_y = 40$ ($\diamond$) and exact solution (solid line).

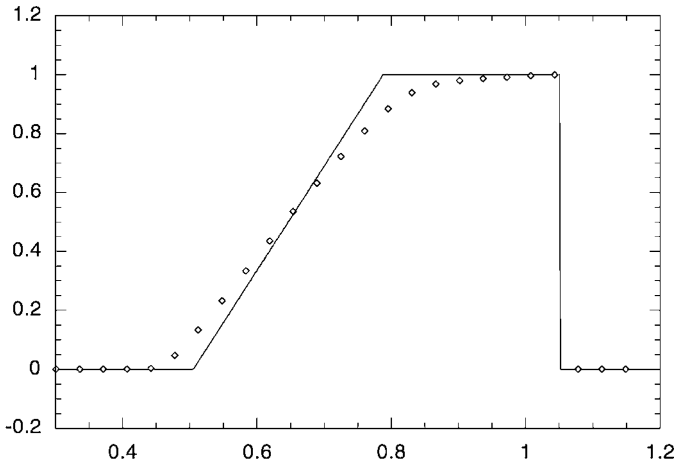

**FIG. 19.** Plot of WENO5 level-set tracking solution at $\xi = \frac{-3}{8\sqrt{2}}$ and $t = 0.2$ with $N_x = N_y = 40$ ($\diamond$) and exact solution (solid line).

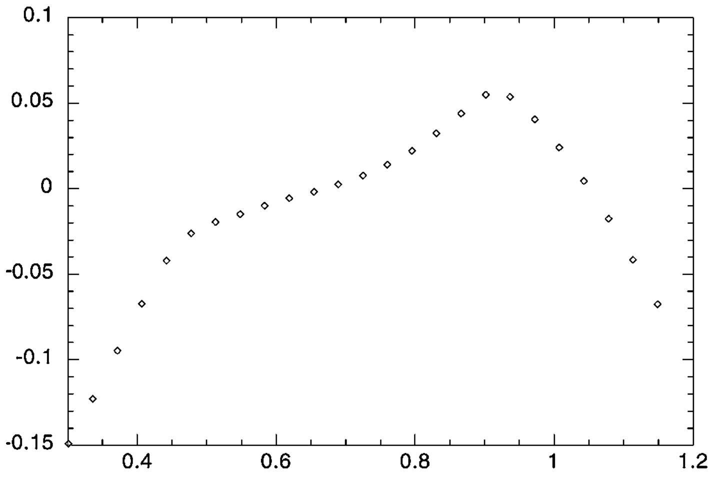

**FIG. 20.** Plot of level-set function at $\xi = \frac{-3}{8\sqrt{2}}$ and $t = 0.2$ with $N_x = N_y = 40$.

## 5. EFFICIENCY

As stated earlier, the present scheme takes roughly three times the effort of a standard shock-capturing algorithm. So, how can this be viewed as a competitive scheme? If one can track all shocks, and maintain high-order convergence, then the method will be very efficient. The goal of a numerical method should be to reduce the CPU time and the error of the numerical solution. The CPU time of an algorithm is a fairly straightforward quantity, say the CPU time per cell update times the number of cells times the number of time steps. The error is something that is not only problem depenent but also dependent on the norm in which the error is measured. Examining the $L_1$ error norm in the various examples in this paper reveals that to achieve the same error, a shock-capturing scheme would typically need a much finer grid than the same tracking algorithm. For the example in Section 4.3, the capturing solution with $N_x = 160$ is still not as accurate as the tracking solution with $N_x = 20$. In this case, the capturing algorithm would take roughly 3 orders of magnitude more CPU time than the tracking scheme to achieve the same $L_1$ error. Still, for problems that do not lend themselves to tracking every last discontinuity at high order, the tracking scheme will capture any untracked discontinuities and the formal high order of convergence will be lost. This is seen in the example in Section 4.4, where the rarefaction corners are captured, resulting in a loss of high-order convergence. In these cases, it would be beneficial to have a more effecient method. This is probably also true for the case of systems of conservation laws, where it very diffiult to track all discontinuities. There are several possibilities for a more efficient method. A few are discussed next.

It has been shown by Sethian [24] and others that the level-set equation need only be updated in a band of points near the zero of the level-set function. This type of update for the level-set function would bring the CPU time of the level-set equation down by a factor of roughly $N_x$.

Also, as in the original ghost fluid algorithm, one can extrapolate from the real region into the ghost region. Then one needs only to update the real region and a band of points in the ghost region near the discontinuity.

Together the two ideas above can be implemented to give a CPU time nearly identical to the standard capturing schemes. This unfortunately complicates the implementation and may make it more difficult to maintain very high order convergence rates (one would need to demonstrate high-order "narrow band" methods for level sets, and high-order extrapolation in the ghost state). Again, these ideas are probably most important in complex problems, such as the Euler equations. Further examination of these ideas will be given elsewhere [2].

## 6. CONCLUSIONS AND DISCUSSION

A simple level-set algorithm for solving scalar hyperbolic conservation laws is presented. For scalar problems, high rates of convergence are demonstrated in linear and nonlinear problems, even near discontinuities and in multidimensions. Note that this scheme takes roughly three times as many numerical operations and memory as a standard capturing scheme, but the benefit greatly outweighs any extra computational time or storage if all discontinuities can be tracked. Also, this extrapolation-free method can be applied to the original ghost fluid method [8], by initially extrapolating the density, etc., in a smooth fashion, and only projecting the ghost node states into the proper boundary condition states. Note that one can also use this algorithm for tracking discontinuous derivatives in Hamilton–Jacobi equations, with the modification that the "shock" speed will be a function of the derivatives of the solution states, since it has been shown [20, 21] that there is a relation between Hamilton–Jacobi equations and scalar conservation laws.

Work is currently being done to track multidimensional shock fronts in gas dynamics with a level-set formulation. The largest difference is that, in a system, there are characteristics that pass through the shock front, and so the boundary treatment must take this into account. Also, the two states will not be symmetric, as in the scalar case; one state, say $\mathbf{u}_2$, will be considered a shock state of the other, $\mathbf{u}_1$. A simple way to apply the appropriate ghost region state for $\mathbf{u}_2$ is to project this ghost region state into an appropriate shock state of the unshocked fluid, $\mathbf{u}_1$. This shock state is determined once one has an appropriate shock speed. The shock speed can be determined from a local Riemann problem. This

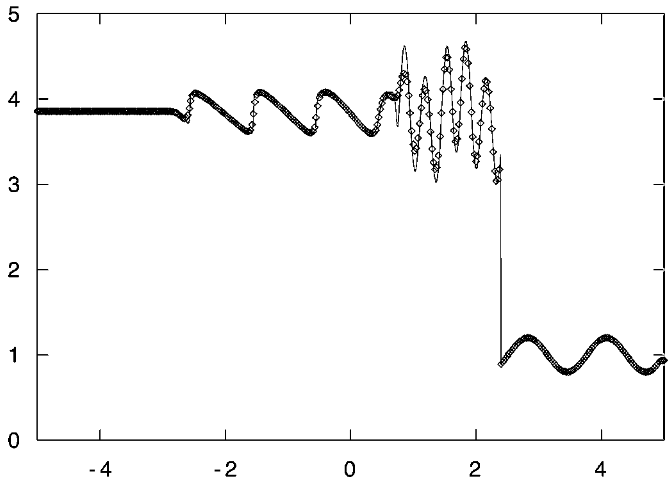

**FIG. 21.** Plot of WENO5 level-set tracking solution at $t = 1.8$ with $N_x = N_t = 400$ ($\diamond$) and converged level-set tracking solution with $N_x = N_t = 1600$ (solid line).

can be determined exactly (e.g., Godunov's method), or it can be determined from an approximate method (e.g., Roe's method). The ghost region for $\mathbf{u}_1$ is not critical, since in the shock-attached reference frame, $\mathbf{u}_1$ is supersonic. Again, one must ensure that both $\mathbf{u}_1$ and $\mathbf{u}_2$ are entropy-satisfying states in their respective ghost regions. Preliminary results indicate that this method converges to the proper solution, with improved convergence and accuracy. Figure 21 shows the density field for a level-set tracking algorithm, using a nondecomposition-based Lax–Friedrichs scheme [15, 28], with a shock speed given from a Roe averaged $\overline{u} + \overline{c}$, corresponding to Example 8 and Fig. 14b of [27]. Importantly, note that there are no "intermediate" shock points near the lead shock at $x \approx 2.4$. Again, this work is investigated in greater detail elsewhere [2].

## REFERENCES

1. M. Arora and P. L. Roe, On postshock oscillations due to shock capturing schemes in unsteady flows, *J. Comput. Phys.* **130**, 25 (1997).
2. T. D. Aslam, A level set algorithm for tracking discontinuities in hyperbolic conservation laws II: Systems of Equations, in preparation.
3. I.-L. Chern, J. Glimm, O. McBryan, B. Plohr, and S. Yaniv, Front tracking for gas dynamics, *J. Comput. Phys.* **62**, 83 (1986).
4. P. Charrier and B. Tessieras, On front-tracking methods applied to hyperbolic systems of nonlinear conservation laws, *SIAM J. Numer. Anal.* **23**(3), 461 (1986).
5. R. Donat, Studies on error propagation for certain nonlinear approximations to hyperbolic equations: Discontinuities in derivatives, *SIAM J. Numer. Anal.* **31**(3), 655 (1994).
6. R. Donat and S. Osher, Propagation of error into regions of smoothness for non-linear approximations to hyperbolic equations, *Comput. Meth. Appl. Mech. Eng.* **80**, 59 (1990).
7. B. Engquist and B. Sjogreen, The convergence rate of finite difference schemes in the presence of shocks, *SIAM J. Numer. Anal.* **35**(6), 2464 (1998).
8. R. P. Fedkiw, T. Aslam, B. Merriman, and S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), *J. Comput. Phys.* **152**, 457 (1999).
9. A. Harten, ENO schemes with subcell resolution, *J. Comput. Phys.* **82**, 148 (1989).
10. A. Harten and S. Osher, Uniformly high-order accurate nonoscillatory schemes, I, *SIAM J. Numer. Anal.* **24**(2), 279 (1987).
11. G.-S. Jiang and D. Peng, Weighted ENO schemes for Hamilton–Jacobi equations, UCLA CAM Report 97-29 (1997) *SIAM J. Sci. Comput.* **21**(6), 2126 (2000).
12. G.-S. Jiang and C.-W. Shu, Efficient implementation of weighted ENO schemes, *J. Comput. Phys.* **126**, 202 (1996).
13. R. J. Leveque, *Numerical Methods for Conservation Laws* (Birkhauser-Verlag, Basel 1992).
14. R. J. LeVeque and K.-M. Shyue, Two-dimensional front tracking based on high resolution wave propagation methods, *SIAM J. Sci. Comput.* **16**, 348 (1995).
15. X.-D. Liu and S. Osher, Convex ENO high order multi-dimensional schemes without field-by-field decomposition or staggered grids, *J. Comput. Phys.* **142**, 304 (1998).
16. X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, *J. Comput. Phys.* **115**, 200 (1994).
17. D.-K. Mao, A treatment of discontinuities in shock-capturing finite difference methods, *J. Comput. Phys.* **92**, 422 (1991).
18. D.-K. Mao, A treatment of discontinuities for finite difference methods, *J. Comput. Phys.* **103**, 359 (1992).
19. D.-K. Mao, A treatment of discontinuities for finite difference methods in the two-dimensional case, *J. Comput. Phys.* **104**, 377 (1993).
20. S. Osher and J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations, *J. Comput. Phys.* **79**, 12 (1988).

21. S. Osher and C.-W. Shu, High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations, *SIAM J. Numer. Anal.* **28**(4), 907 (1991).

22. J. J. Quirk, A contribution to the great Riemann solver debate, *Int. J. Numer. Meth. Fluids* **18**, 555 (1994).

23. T. W. Roberts, The behavior of flux difference splitting schemes near slowly moving shock waves, *J. Comput. Phys.* **90**, 141 (1990).

24. J. A. Sethian, *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Material Sciences* (Cambridge University Press, Cambridge, UK, 1996).

25. M. Sever (Mock), Order of dissipation near rarefaction centers, in *Progress and Supercomputing in Computational Fluid Dynamics, Proceedings of U.S.–Isreal Workshop* (Birkhauser, Boston, 1985), p. 395.

26. C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, *J. Comput. Phys.* **77**, 439 (1988).

27. C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes II, *J. Comput. Phys.* **83**, 32 (1989).

28. S. Xu, T. Aslam, and D. S. Stewart, High resolution numerical simulation of ideal and non-ideal compressible reacting flows with embedded internal boundaries, *Combust. Theory Model.* **1**(1), 113 (1997).

29. H. Yang, An artificial compression method for ENO schemes: The Slope modification method, *J. Comput. Phys.* **89**, 125 (1990).