An Eulerian-Lagrangian WENO Scheme for Nonlinear Conservation Laws

Chieh-Sen Huang[a,1], Todd Arbogast[b,2]

[a]*Department of Applied Mathematics, National Sun Yat-sen University, Kaohsiung 804, Taiwan, R.O.C.*
[b]*University of Texas at Austin; Institute for Computational Engineering and Sciences; 201 EAST 24th St., Stop C0200; Austin, TX 78712–1229 and Mathematics Department, RLM 8.100; 2515 Speedway, Stop C1200; Austin, TX 78712–1202; U.S.A.*

Abstract

We develop a formally high order Eulerian-Lagrangian WENO finite volume scheme for nonlinear scalar conservation laws that combines ideas of Lagrangian traceline methods with WENO reconstructions. The particles within a grid element are transported in the manner of a standard Eulerian-Lagrangian (or semi-Lagrangian) scheme using a fixed velocity v. A flux correction computation accounts for particles that cross the v-traceline during the time step. If $v = 0$, the scheme reduces to an almost standard WENO5 scheme. The CFL condition is relaxed when v is chosen to approximate either the characteristic or particle velocity. Excellent numerical results are obtained using relatively long time steps.

The v-traceback points can fall arbitrarily within the computational grid, and linear WENO weights may not exist for the point. A general WENO technique is described to reconstruct to any order the integral of a smooth function using averages defined over a general, nonuniform computational grid. Moreover, to high accuracy, local averages can also be reconstructed. By re-averaging the function to a uniform reconstruction grid that includes a point of interest, one can apply a standard WENO reconstruction to obtain a high order point value of the function.

Keywords: hyperbolic transport, semi-Lagrangian, finite volume, locally conservative, characteristics, traceline, re-average.
2000 MSC: 65M08, 65M25, 76M12

1. Introduction

The object of this paper is to develop an *Eulerian-Lagrangian Weighted Essentially Non-Oscillatory* (EL-WENO) finite volume scheme for solving nonlinear conservation laws. We use Strang splitting in space to handle multiple dimensions, so we concentrate on the following scalar conservation law in one space dimension:

$$u_t + (f(u))_x = 0, \quad x \in \mathbb{R},\ t > 0, \tag{1.1}$$

$$u(x, 0) = u_0(x), \quad x \in \mathbb{R}, \tag{1.2}$$

Email addresses: huangcs@math.nsysu.edu.tw (Chieh-Sen Huang), arbogast@ices.utexas.edu (Todd Arbogast)

[1]Supported in part under Taiwan National Science Council grants NSC 99-2115-M-110-006-MY3 and NSC 102-2115-M-110-010-MY3.

[2]Supported as part of the Center for Frontiers of Subsurface Energy Security, an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under award DE-SC0001114 (for work on the scheme itself) and the U.S. National Science Foundation (for work on the re-averaging technique and Euler systems) under grants DMS-0835745 and DMS-1418752.

where $f(u) = f(u; x, t)$ is the (possibly) nonlinear flux function.

The Essentially Non-Oscillatory (ENO), Weighted ENO (WENO), and Central WENO (CWENO) schemes are now classic methods [13, 14, 20, 22, 23, 25]. They are a class of formally high-order finite difference and finite volume schemes using adaptive stencils for accurately solving a conservation law with essentially no oscillation in the solution. Because they are explicit schemes designed on a fixed Eulerian grid, they are very efficient, but they are also subject to a (CFL) time step stability restriction. The *Eulerian-Lagrangian* or *semi-Lagrangian* schemes [1, 2, 5, 7, 8, 10, 11, 12, 24, 33, 34] follow the motion of the particles, and so alleviate the CFL time step restriction, although it can be difficult to design and implement accurate schemes for nonlinear problems.

Eulerian-Lagrangian WENO schemes for the linear advection equation have recently appeared in the literature, including the finite difference schemes of J.-M. Qiu, Christlieb, and Shu [26, 27, 28] and the finite volume scheme of the current authors and J. Qiu [19]. Extension to nonlinear equations is problematic. One does not have the particle velocity, $f(u)/u$, since it is nonlinearly related to the unknown solution u, and so one cannot find the exact tracelines of the fluid particles. A formally second order Eulerian-Lagrangian scheme was recently introduced by the current authors and Russell [3] for (1.1) when $f = f(u)$ only. The scheme requires the solution of a minimization problem to approximate the foot of each traceline over each time step interval (i.e., the location of each grid point traced backward in time). This part of the scheme can be costly and may not be robust. It is also not so clear how to generalize the scheme to two and three dimensional problems.

In this work, we propose an *approximate* Eulerian-Lagrangian scheme combined with high order WENO reconstructions. There are three keys to the work. The first key is to abandon any hope of finding the exact tracelines of the particles, and instead use a known velocity field $v(x, t)$ for the traceline computation, somewhat as is used in the Arbitrary Lagrangian-Eulerian (ALE) methods to follow the motion of a deforming body [9]. Since this will not give the correct tracelines, there will be a flux of particles crossing the traceline, and we will approximate this flux using WENO reconstructions. That is, the differential equation (1.1) is rewritten as

$$u_t + (vu)_x + (f(u) - vu)_x = 0, \tag{1.3}$$

and the first two terms are treated in a Lagrangian setting, leaving a flux correction to be computed on the term $(f(u) - vu)_x$. If $v \approx f(u)/u$, then the effective velocity for flux correction is relatively slow, and a relaxed CFL constraint is obtained.

The second key to the work is to develop a procedure to handle WENO reconstruction at the traceback points. However, the linear weights in a WENO reconstruction do not necessarily exist at an arbitrary point. To overcome this limitation, we use and extend a re-averaging technique recently introduced [19, 18]. The technique allows WENO reconstruction at an arbitrary point, and so we can obtain formally high order results that are yet essentially non-oscillatory. We make strong use of the general WENO interpolation theory of Carlini, Ferretti, and Russo [6] in this part of the work.

Under a non-constant velocity field, the traceback points of a uniform grid become nonuniform, and WENO reconstruction becomes more difficult. While not strictly necessary, the third key to the work is to avoid nonuniform spacing of the traceback points by using a locally frozen velocity tracing near each point, so that the points trace along parallel trajectories.

We remark that the moving mesh methods (see, e.g., [31, 36]) are similar to Eulerian-Lagrangian methods, and the adaptive WENO methods [29] are related to solution re-averaging, and so share many similar ideas to what we present herein.

It is possible to define any order EL-WENO scheme, but for the purposes of exposition, we restrict to the fifth order case. We present in complete generality the re-averaging technique for WENO reconstruction

to any order on nonuniform grids at an arbitrary point in the next section. The EL-WENO5 scheme for the nonlinear problem (1.1) is given in Section 3. Numerical results showing application of the scheme to linear and nonlinear problems are given in Sections 4 and 5. We include some problems in two space dimensions, by using a Strang splitting [32] in the spatial variables to reduce to a pair of one dimensional problems as is usually done for finite volume WENO methods. Even though Strang splitting is formally only second order accurate in space, the numerical examples will show fifth order accuracy, because the splitting error is normally quite small for the type of problems one wants to solve. (For a problem in which the splitting error is not small, one should be able to use Richardson Extrapolation or some kind of iterative technique to reduce the splitting error, but this is not the focus of this paper and we do not pursue the matter here, but see, e.g. [16, 17]). Numerical results for the Euler system are given in Section 6. Conclusions appear in the final section.

2. A Re-Averaging Technique

We present a fully general re-averaging technique in this section, suitable for high order reconstruction at arbitrary points. We first generalize the technique presented in [19] for high order reconstruction of integrals using the theory of [6].

2.1. Reconstruction of integrals and averages

Fix the positive integer n, which will be the degree of the low order polynomials used in the reconstruction. In this section, we take an arbitrary computational grid of points given by $x_0 < x_1 < \cdots < x_{2n-1}$. We assume that we have as data the cell average values of a function $u(x)$, which are

$$\bar{u}_i = \frac{1}{x_{i+1} - x_i} \int_{x_i}^{x_{i+1}} u(x)\, dx, \quad i = 0, 1, \ldots, 2n-2. \tag{2.1}$$

Our goal is to approximate over the central grid element $[x_{n-1}, x_n]$ the integral or primitive function of u, which is arbitrarily set to zero at x_0 and is

$$U(x) = \int_{x_0}^{x} u(y)\, dy.$$

That is, for $\alpha \in (x_{n-1}, x_n)$, we wish to approximate $U(\alpha)$ to $O(h^{2n})$, with h being the maximum grid spacing. As is well-known, we reinterpret the data as

$$\bar{U}_i = \sum_{j=0}^{i-1} \bar{u}_j (x_{j+1} - x_j), \quad i = 0, 1, \ldots, 2n-1,$$

and define the n Lagrange interpolation polynomials P_ℓ, each of degree n, on the various stencils $\{x_\ell, \ldots, x_{\ell+n}\}$, for $\ell = 0, 1, \ldots, n-1$, by the conditions

$$P_\ell(x_j) = \bar{U}_j, \quad j = \ell, \ell+1, \ldots, \ell+n, \quad \ell = 0, 1, \ldots, n-1.$$

We also define the higher order Lagrange polynomial $H(x)$, of degree $2n-1$, which uses all the data and satisfies

$$H(x_j) = \bar{U}_j, \quad j = 0, 1, \ldots, 2n-1.$$

Our goal is to find a fixed linear combination of the P_ℓ which matches H at α, for any data. If we can do this, then we have our $O(h^{2n})$ approximation of the integral, since the Lagrange interpolation satisfies, for some $\xi(\alpha) \in (x_0, x_{2n-1})$,

$$U(\alpha) = H(\alpha) + \frac{u^{(2n-1)}(\xi(\alpha))}{(2n)!} \prod_{j=0}^{2n-1} (\alpha - x_j) = H(\alpha) + O(h^{2n}). \tag{2.2}$$

Carlini, Ferretti, and Russo [6] proved that the linear weights γ_ℓ^α always exist for WENO interpolation, and that they are positive and sum to one, so that indeed

$$H(\alpha) = \sum_{\ell=0}^{n-1} \gamma_\ell^\alpha P_\ell(\alpha),$$

for any choice of data.

Theorem 2.1. *Given a computational grid $x_0 < x_1 < \cdots < x_{2n-1}$, suppose that $u(x) \in C^{2n-1}([x_0, x_{2n-1}])$, and let $\bar{u}_i$ be defined by* (2.1)*, for $i = 0, 1, \ldots, 2n-2$. The n interpolation polynomials P_ℓ, each of degree n, can be defined by the conditions*

$$P_\ell(x_j) = \sum_{k=0}^{j-1} \bar{u}_k \, (x_{k+1} - x_k), \quad j = \ell, \ell+1, \ldots \ell+n, \quad \ell = 0, 1, \ldots, n-1.$$

If $\alpha \in (x_{n-1}, x_n)$, then there exist positive linear weights γ_ℓ^α such that

$$\begin{aligned} &\sum_{\ell=0}^{n-1} \gamma_\ell^\alpha P_\ell(\alpha) - \sum_{k=0}^{n-2} \bar{u}_k \, (x_{k+1} - x_k) \\ &\qquad = \int_{x_{n-1}}^{\alpha} u(x)\, dx - \frac{u^{(2n-1)}(\xi(\alpha))}{(2n)!} \prod_{j=0}^{2n-1} (\alpha - x_j) \end{aligned} \tag{2.3}$$

for some $\xi(\alpha) \in (x_0, x_{2n-1})$. Moreover, if $u(x) \in C^{2n}([x_0, x_{2n-1}])$, then for any $x_{n-1} \le \alpha < \beta \le x_n$,

$$\frac{1}{\beta - \alpha} \int_\alpha^\beta u(x)\, dx = \frac{1}{\beta - \alpha} \sum_{\ell=0}^{n-1} (\gamma_\ell^\beta P_\ell(\beta) - \gamma_\ell^\alpha P_\ell(\alpha)) + O(h^{2n-1}). \tag{2.4}$$

The theorem is proved in [6], except for the estimate (2.4), which follows from (2.3) and the Mean Value Theorem.

Following [6], the linear weights can be computed easily as follows. We first set

$$W_\ell^m = \prod_{j=0}^{\ell-1} (x_m - x_j) \prod_{j=\ell+n+1}^{2n-1} (x_j - x_m), \tag{2.5}$$

and then solve the lower triangular matrix problem

$$\begin{bmatrix} W_0^0 & 0 & \cdots & 0 \\ W_0^1 & W_1^1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ W_0^{n-2} & W_1^{n-2} & \cdots & W_{n-2}^{n-2} \end{bmatrix} \begin{bmatrix} \hat{\gamma}_0 \\ \hat{\gamma}_1 \\ \vdots \\ \hat{\gamma}_{n-2} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \tag{2.6}$$

which is independent of α, for the scale factors $\hat{\gamma}_0, \ldots, \hat{\gamma}_{n-2}$. Finally,

$$\gamma_\ell^\alpha = \hat{\gamma}_\ell \prod_{j=0}^{\ell-1} (\alpha - x_j) \prod_{j=\ell+n+1}^{2n-1} (x_j - \alpha), \quad \ell = 0, 1, \ldots, n-2, \tag{2.7}$$

$$\gamma_{n-1}^\alpha = 1 - \sum_{\ell=0}^{n-2} \gamma_\ell^\alpha. \tag{2.8}$$

If the computational grid is uniform, the scale factors are given by

$$\hat{\gamma}_\ell = \binom{n-1}{\ell} \frac{n!}{(2n-1)!} h^{1-n}.$$

The nonlinear weights can be defined as a modification of the linear weights in the usual way to account for nonsmoothness of the function.

2.2. High order reconstruction of function values at arbitrary points

Let the computational grid be given by $\cdots < x_{-2} < x_{-1} < x_0 < x_1 < x_2 < \cdots$. The linear weights exist for high order reconstruction of the function value $u(x_j)$, for any grid point x_j. Suppose that we are interested in the arbitrary point $x \in (x_k, x_{k+1})$.

We define a *reconstruction grid* $\xi_0 < \xi_1 < \cdots < \xi_{2n}$ such that $\xi_n = x$, and we re-average by defining high order approximations to the reconstruction grid average values

$$\tilde{u}_i \approx \frac{1}{\xi_{i+1} - \xi_i} \int_{\xi_i}^{\xi_{i+1}} u(x)\, dx, \quad i = 0, 1, \ldots, 2n-1.$$

To do this, first find the largest indices j^* so $x_{j^*} \le \xi_i$ and j^{**} so $x_{j^{**}} < \xi_{i+1}$. Denote by $U_j(\alpha)$ the previous high order reconstruction of the integral of u over $[x_j, \alpha]$, where $x_j \le \alpha < x_{j+1}$ (which uses the computational grid). Then

$$\tilde{u}_i\,(\xi_{i+1} - \xi_i) = \sum_{j=j^*}^{j^{**}-1} \bar{u}_j\,(x_{j+1} - x_j) - U_{j^*}(\xi_i) + U_{j^{**}}(\xi_{i+1}),$$

where the sum is vacuous if $j^{**} = j^*$. Note that the reconstruction has used the values $\bar{u}_j$ for $j = j^* - n + 1, \ldots, j^{**} + n - 1$.

These re-averaged values can be used for a standard high order WENO reconstruction of the function at the point $\xi_n = x \in (x_k, x_{k+1})$. The stencil size may grow if one is not careful. If one uses, say, a uniform reconstruction grid of spacing $h^* = (1/n)\min(x - x_k, x_{k+1} - x)$, then the entire reconstruction grid is contained within $[x_k, x_{k+1}]$. In this case, $j^* = j^{**} = k$, and the stencil remains fixed at size $2n - 1$. However, rounding error will be an issue if h^* becomes very small.

In fact, there are two standard WENO reconstructions at a grid point x, one for a left stencil using $\{\tilde{u}_0, \tilde{u}_1, \ldots, \tilde{u}_{2n-2}\}$ and the other for a right stencil using $\{\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_{2n-1}\}$. In this work, we sometimes simply average these two reconstructions, and at other times we use a Godunov flux splitting and select the upstream stencil (especially for the Buckley-Leverett examples to follow) or we use the values in a Roe flux solver for the Euler system (see Section 6). As an alternative, for WENO5 reconstruction one could reconstruct to the center of the grid element (i.e., omit the point ξ_{2n} from the reconstruction grid, which is shifted so that $x = (\xi_{n-1} + \xi_n)/2$), since the linear weights exist at the center point for fifth order reconstruction.

We also chose in this work to use a reconstruction grid of spacing $h^* = h/2$. To obtain a high order WENO reconstruction at an arbitrary point, then, requires a larger stencil than the normal $2n-1$ computational grid average values, due to the re-averaging. We will need to extend the stencil to cover a support that is larger by twice nh^*. This means that we need an additional n points in the stencil, for a stencil size of $3n-1$ computational grid average values. However, with this choice, the overall stencil of the scheme, including the Runge-Kutta step described later, is the same as is used in a standard Eulerian WENO5 scheme.

3. The EL-WENO Scheme

We present the EL-WENO5 scheme in this section. Other order schemes could be developed along the same lines. A Strang splitting in space can be used to extend the one dimensional scheme to higher dimensions, but we omit the details (see, e.g., [19]).

We choose time levels $0 = t^0 < t^1 < t^2 < \cdots$. For simplicity of the notation, we will use whole indices for enumerating both the computational grid points and the grid elements. We choose the spacing $h > 0$ and set the grid points to be $x_i := ih$. The ith grid element is $E_i := [x_i, x_{i+1}]$. The function $u(x)$ is approximated on the grid by its element averages

$$\bar{u}_i \approx \frac{1}{h}\int_{x_i}^{x_{i+1}} u(x)\,dx. \tag{3.1}$$

3.1. The approximate velocity v for Lagrangian tracing

Lagrangian tracing of a particle at position x at time t^{n+1} through the velocity field $v(x,t)$ gives the position $\check{x}(t) = \check{x}(x;t)$, which we will call the v-trace. It satisfies

$$\frac{d\check{x}}{dt} = v(\check{x},t) \quad \text{and} \quad \check{x}(t^{n+1}) = x. \tag{3.2}$$

Generally we trace backward in time to t^n, and we call $\check{x}^n(x) := \check{x}(x;t^n)$ the v-traceback point for x. For a grid point x_i, we also denote $\check{x}_i(t) := \check{x}(x_i;t)$ and $\check{x}_i^n := \check{x}(x_i;t^n)$.

In an Eulerian-Lagrangian method, one traces all the particles within an Eulerian grid element. For the grid element $E = E_i = [x_i, x_{i+1}]$, we denote by $\mathcal{E}_E$ the resulting swept space-time region using the velocity v, which is

$$\mathcal{E}_E := \{(\check{x}(x,t),t) : t^n \le t \le t^{n+1} \text{ and } x \in E\}.$$

The boundary consists of four parts, E on the top when $t = t^{n+1}$ and

$$\begin{aligned}
\check{E} &:= \{\check{x}(x,t^n) : x \in E\} = [\check{x}_i^n, \check{x}_{i+1}^n] && \text{on the bottom when } t = t^n,\\
\mathcal{S}_{E,L} &:= \{(\check{x}_i(t),t) : t^n \le t \le t^{n+1}\} && \text{on the left side,}\\
\mathcal{S}_{E,R} &:= \{(\check{x}_{i+1}(t),t) : t^n \le t \le t^{n+1}\} && \text{on the right side.}
\end{aligned}$$

We call the interval $\check{E}$ the v-traceback of E.

3.2. Mass conservation over the v-traceback space-time region

We view the differential equation (1.1) in divergence form, i.e., as

$$\nabla_{t,x} \cdot \begin{pmatrix} u \\ f(u) \end{pmatrix} = 0. \tag{3.3}$$

For each grid element E, we integrate this in space-time over $\mathcal{E}_E$ and apply the divergence theorem. The result is

$$\begin{aligned}
&\int_E u(x,t^{n+1})\,dx - \int_{\check{E}} u(x,t^n)\,dx \qquad (3.4)\\
&\quad = -\int_{\mathcal{S}_{E,L}} \begin{pmatrix} u \\ f(u)\end{pmatrix}\cdot\begin{pmatrix} v \\ -1\end{pmatrix}\frac{d\sigma}{\sqrt{1+v^2}} - \int_{\mathcal{S}_{E,R}} \begin{pmatrix} u \\ f(u)\end{pmatrix}\cdot\begin{pmatrix} -v \\ 1\end{pmatrix}\frac{d\sigma}{\sqrt{1+v^2}}\\
&\quad = \int_{t^n}^{t_{n+1}} (f(u)-vu)\big|_{x=\check{x}_i(t)}\,dt - \int_{t^n}^{t_{n+1}} (f(u)-vu)\big|_{x=\check{x}_{i+1}(t)}\,dt.
\end{aligned}$$

In a strict Eulerian-Lagrangian method, one would trace the particles in space-time along the physically correct velocity, $f(u)/u$. With $v = f(u)/u$, there would be no flux terms on $\mathcal{S}_{E,L}$ and $\mathcal{S}_{E,R}$. In our setting, with $f(u)/u$ unknown, we need to account for this flux, as well as the integration of u over the v-traceback $\check{E}$.

Of course, the integral at time t^{n+1} in (3.4) is approximated as

$$\int_E u(x,t^{n+1})\,dx \approx h\bar{u}_i^{n+1}. \qquad (3.5)$$

The integral at time t^n is approximated the same as in the linear transport scheme described by the authors and Qiu in [19]. Briefly, one approximates

$$\int_{\check{E}} u(x,t^n)\,dx = \sum_j \int_{\check{E}\cap E_j} u(x,t^n)\,dx \approx \sum_j \int_{\check{E}\cap E_j} \mathcal{R}_j(\bar{u};x,t^n)\,dx, \qquad (3.6)$$

where $\mathcal{R}_j(\bar{u};x,t^n)$ is the WENO5 reconstruction of $\{\bar{u}^n_{j-2},\bar{u}^n_{j-1},\bar{u}^n_j,\bar{u}^n_{j+1},\bar{u}^n_{j+2}\}$ targeting higher order integration, as described in Section 2.1. It is a linear combination of the three quadratic polynomials that preserve average mass over each grid element of the interval $[x_{j-2},x_{j+1}]$, $[x_{j-1},x_{j+2}]$, and $[x_j,x_{j+3}]$, respectively. If the domain of integration $\check{E}\cap E_j$ includes an endpoint x_j and/or x_{j+1}, so that it is either $[x_j,x_j+\alpha h]$ or $[x_j+\alpha h,x_{j+1}]$ for $0\le\alpha\le 1$, then the three linear weights are, respectively,

$$\gamma_L = \frac{(2-\alpha)(3-\alpha)}{20},\quad \gamma_C = \frac{(2+\alpha)(3-\alpha)}{10},\quad \gamma_R = \frac{(1+\alpha)(2+\alpha)}{20}.$$

If instead $\check{E}\cap E_j = [x_j+\beta h, x_j+\alpha h]$ does not include an endpoint, one can preserve mass using the above linear weights and the formula

$$\int_{x_j+\beta h}^{x_j+\alpha h} \mathcal{R}_j(x)\,dx = \int_{x_j}^{x_j+\alpha h} \mathcal{R}_j(x)\,dx - \int_{x_j}^{x_j+\beta h} \mathcal{R}_j(x)\,dx.$$

These linear weights give formal fifth order accuracy for the solution, in the sense that we have sixth order accuracy for each piece of the integral. A standard nonlinear modification of the linear weights, as in any WENO reconstruction, is also employed.

3.3. Approximation of the flux between space-time regions

It remains to approximate the two integrals of the flux in (3.4) over the space-time paths $\mathcal{S}_{E,L}$ and $\mathcal{S}_{E,R}$. Consider the left integral, and approximate it using the three-point Gauss rule (which is accurate to order $O(\Delta t^6)$) as

$$\begin{aligned}
&\int_{t^n}^{t_{n+1}} (f(u)-vu)\big|_{x=\check{x}_i(t)}\,dt \qquad (3.7)\\
&\quad \approx \sum_{\ell=-1}^{1} \omega_\ell\big[f(u(\check{x}_i(t_\ell),t_\ell)) - v(\check{x}_i(t_\ell),t_\ell)\,u(\check{x}_i(t_\ell),t_\ell)\big],
\end{aligned}$$

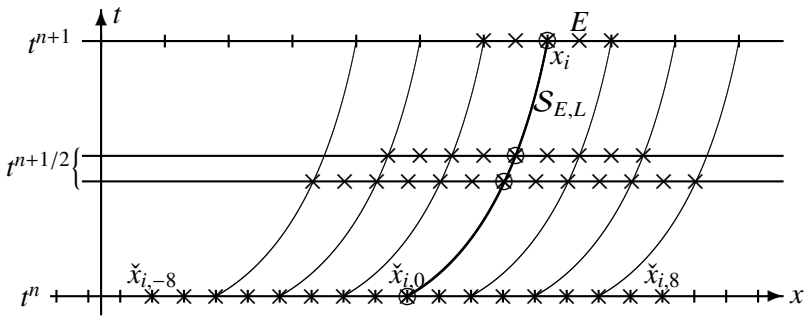


Figure 1: A depiction of RK4 used to reconstruct the flux along the v-trace. The computational grid is shown at time t^{n+1}. The left side of grid element E, the point x_i, is v-traced to $\check{x}_{i,0}$ at time t^n, tracing out the path $\mathcal{S}_{E,L}$. The point $\check{x}_{i,0}$ fixes the reconstruction grid of spacing $h^* = h/2$, shown at time t^n. Similarly, along the frozen velocity path, the point $x_{j+k/2}$ is traced on a parallel trajectory to $\check{x}_{i,k}$. The solution is reconstructed at these 17 points at time t^n, and these first stage points are used to obtain the second stage solution at 13 points at time $t^{n+1/2}$. These are used to obtain the third stage solution at 9 points, again at time $t^{n+1/2}$. Finally, the 5 points of the solution at stage four are obtained at time t^{n+1}. The solution at (x_i, t^{n+1}) is then computed.

where ω_ℓ are the Gauss weights and t_ℓ are the Gauss points for the interval $[t^n, t^{n+1}]$. Therefore, the main task is to find a proper extrapolation of values for $u(\check{x}_i(t), t)$ at the Gauss points in time along the v-trace. We use the technique of Levy, Puppo, and Russo [22] to handle the time evolution, which is to the apply a Runge-Kutta method and use the natural continuous extension of Zennaro [37]. Unlike the case of Eulerian methods, the Lagrangian case has a path of integration that evolves in space, and so some special treatment is required.

Along the v-trace, the evolution of u is governed by

$$\begin{aligned}\frac{du(\check{x}(t),t)}{dt} &= u_x(\check{x}(t),t)\frac{d\check{x}}{dt} + u_t(\check{x}(t),t) \\ &= u_x(\check{x}(t),t)\, v(\check{x}(t),t) - (f(u))_x\Big|_{(x,t)=(\check{x}(t),t)} := F(u;x,t).\end{aligned} \tag{3.8}$$

The solution of this problem by a Runge-Kutta method requires an evaluation of the right-hand side function $F(u)$, and therefore an evaluation of the x-derivatives of f and u, at the intermediate Runge-Kutta times along the v-trace path. First, we reconstruct the function values by using the re-averaging technique of the previous section and as discussed in Section 2.2, using a uniform reconstruction grid of spacing $h^* = h/2$. This gives a high order approximation of the solution u at the point $\check{x}_i^n$, which we denote by $\check{u}_{i,0}$. If $\check{x}_i^n \in [x_j, x_{j+1})$, then $\check{u}_{i,0}$ is reconstructed using eight of the nine values $\{\bar{u}_{j-4}^n, \ldots, \bar{u}_{j+4}^n\}$ (depending on which side of the center $\check{x}_i^n$ falls within its computational grid element). We can then evaluate $\check{f}_{i,0} = f(\check{u}_{i,0})$. To reconstruct derivatives at $\check{x}_i^n$, we need to similarly reconstruct nearby values $\check{u}_{i,k}$ and $\check{f}_{i,k}$ along the parallel v-traces (i.e., for the moment we freeze the velocity v) shifted to go through $\check{x}_i^n + kh^*$. A standard WENO reconstruction is used to obtain the derivatives of u and f needed in the evaluation of F.

To be more precise, suppose that we use a standard fourth order Runge-Kutta method (RK4). As depicted in Fig. 1, we need five values at each stage of Runge-Kutta to reconstruct the needed derivatives of u and f to evaluate F. For the first stage (when $t = t^n$), we therefore need to reconstruct the seventeen values $\check{u}_{i,k}$ at $\check{x}_i^n + kh^*$, for $k = -8, -7, \cdots, 8$, along the parallel v-traces. These give the thirteen intermediate slopes

$$\begin{aligned} g_{i+k}^{(1)} &= F(t^n, \mathcal{R}(\check{u}))\Big|_{x=\check{x}_{i+k}} \\ &= \left[(\mathcal{R}(\check{u}))_x\, v^n - (\mathcal{R}(f(\check{u})))_x\right]\Big|_{x=\check{x}_{i+k}}, \quad k = -6, -5, \cdots, 6. \end{aligned}$$

These are used at the second stage (when $t = t^n + \Delta t/2$) to evaluate thirteen solution values and nine slopes

$$\begin{aligned}
\breve{u}^{(2)}_{i+k} &= \mathcal{R}(\breve{u})\big|_{x=\breve{x}_{i+k}} + (\Delta t/2)\, g^{(1)}_{i+k}, \quad k = -6, -5, \cdots, 6,\\
g^{(2)}_{i+k} &= F(t^{n+1/2}, \mathcal{R}(\breve{u}^{(2)}))\big|_{x=\breve{x}_{i+k}}\\
&= \left[(\mathcal{R}(\breve{u}^{(2)}))_x\, v^{n+1/2} - (\mathcal{R}(f(\breve{u}^{(2)})))_x\right]\Big|_{x=\breve{x}_{i+k}}, \quad k = -4, -3, \cdots, 4.
\end{aligned}$$

These in turn are used at the third stage (also at time $t^{n+1/2}$) to produce the nine intermediate solution values and five slopes

$$\begin{aligned}
\breve{u}^{(3)}_{i+k} &= \mathcal{R}(\breve{u})\big|_{x=\breve{x}_{i+k}} + (\Delta t/2)\, g^{(2)}_{i+k}, \quad k = -4, -3, \cdots, 4,\\
g^{(3)}_{i+k} &= F(t^{n+1/2}, \mathcal{R}(\breve{u}^{(3)}))\big|_{x=\breve{x}_{i+k}}\\
&= \left[(\mathcal{R}(\breve{u}^{(3)}))_x\, v^{n+1/2} - (\mathcal{R}(f(\breve{u}^{(3)})))_x\right]\Big|_{x=\breve{x}_{i+k}}, \quad k = -2, -1, \cdots, 2.
\end{aligned}$$

Finally, the third stage values produce the fourth stage (time t^{n+1}) solution at five points and the one slope

$$\begin{aligned}
\breve{u}^{(4)}_{i+k} &= \mathcal{R}(\breve{u})\big|_{x=\breve{x}_{i+k}} + \Delta t\, g^{(3)}_{i+k}, \quad k = -2, -1, \cdots, 2,\\
g^{(4)}_{i} &= F(t^{n+1}, \mathcal{R}(\breve{u}^{(4)}))\big|_{x=\breve{x}_{i}} = \left[(\mathcal{R}(\breve{u}^{(4)}))_x\, v^{n+1} - (\mathcal{R}(f(\breve{u}^{(4)})))_x\right]\Big|_{x=\breve{x}_{i}}.
\end{aligned}$$

Finally, we have enough to approximate the desired solution $\tilde{u}^{n+1}_i$ at the point x_i at time t^{n+1} via

$$\tilde{u}^{n+1}_i = \mathcal{R}(\breve{u})\big|_{x=\breve{x}_{i}} + (\Delta t/6)[g^{(1)}_i + 2g^{(2)}_i + 2g^{(3)}_i + g^{(4)}_i].$$

Note that the full reconstruction requires the finite stencil $\bar{u}^n_{j-8}, \ldots, \bar{u}^n_{j+8}$ of seventeen values, which is the same as is used in an Eulerian CWENO5 scheme. It should be noted that for CWENO5, the seventeen reconstructed values of the solution can be reused for other indices. In our case, neighboring points may not trace a distance h away, and so no reuse of the reconstructed values can be assumed.

We actually need the solution at the Gauss points of the interval $[t^n, t^{n+1}]$. Rather than applying Runge-Kutta multiple times, we use the natural continuous extension [37] of the Runge-Kutta scheme, which provides a uniform accuracy of the solution in the time interval. Each ν-stage Runge-Kutta method of order p has a natural continuous extension of degree $d \le p$, in the sense that there exist ν polynomials $b_i(\theta)$, $i = 1, \ldots, \nu$ of degree at most d, such that

$$\begin{aligned}
&\tilde{u}(t^n + \theta\Delta t) := \mathcal{R}(\breve{u})^n + \Delta t \sum_{\ell=1}^{\nu} b_\ell(\theta) g^{(\ell)}, \quad 0 \le \theta \le 1,\\
&\tilde{u}(t^n) = \mathcal{R}(\breve{u}) \quad \text{and} \quad \tilde{u}(t^{n+1}) = \tilde{u}^{n+1},
\end{aligned}$$

so that the uniform error of an $\ell \le d$ derivative is $O((\Delta t)^{d+1-\ell})$. In our case of RK4,

$$\begin{aligned}
b_1(\theta) &= \tfrac{2}{3}\theta^3 - \tfrac{3}{2}\theta^2 + \theta,\\
b_2(\theta) = b_3(\theta) &= -\tfrac{2}{3}\theta^3 + \theta^2,\\
b_4(\theta) &= \tfrac{2}{3}\theta^3 - \tfrac{1}{2}\theta^2.
\end{aligned}$$

We can thus obtain the solution $\breve{u}_i(t_\ell)$ at the Gauss times t_ℓ along the v-trace, needed in the evaluation of (3.7). Technically, RK4 is only $O(\Delta t^4)$ accurate. However, the fact that (3.7) is a spatial difference of two similar quantities gives another $O(h)$, and the Gauss weights are themselves $O(\Delta t)$, so the overall error in the flux approximation is $O(\Delta t^5\, h)$, which is sufficient.

3.4. Summary of the scheme

In summary, the scheme is to compute

$$\bar{u}_i^{n+1} = \frac{1}{h}\Big\{ \sum_j \int_{\check{E}\cap E_j} \mathcal{R}_j(\bar{u}; x, t^n)\, dx \tag{3.9}$$
$$+ \sum_{\ell=-1}^{1} \omega_\ell \Big(\big[f(\check{u}_i(t_\ell), t_\ell) - v(\check{x}_i(t_\ell), t_\ell)\, \check{u}_i(t_\ell) \big]$$
$$- \big[f(\check{u}_{i+1}(t_\ell), t_\ell) - v(\check{x}_{i+1}(t_\ell), t_\ell)\, \check{u}_{i+1}(t_\ell) \big] \Big) \Big\}.$$

The intuition is that if a fixed grid scheme can do well using $v = 0$ for the v-trace, which is a fixed grid Eulerian scheme, then using a non-trivial but physically representative v can only improve the solution. We test this assertion in the numerical examples, which follow.

4. Application to Linear Transport

In this section we present some results for linear transport, i.e., when $f(u; x, t) = a(x, t)u$. In this case, we should take $v = a$, and then the tracelines coming from the ordinary differential equation (3.2) can perhaps be found as accurately as one would wish, and a true Eulerian-Lagrangian scheme should be used, as in [19]. Nevertheless, application of the approximate scheme presented herein is useful in at least three ways. First, it provides an illustration of the new scheme in an easier to interpret linear context, as a prelude to the nonlinear results presented later. Second, in some cases it may not be desirable to invest computational effort to obtain highly accurate tracelines. Finally, one can view the new scheme as an extension of WENO schemes from fixed in time Eulerian grids to essentially arbitrary space-time grids, and the linear case is simply one application.

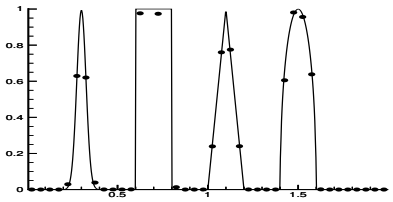
In anticipation of the application to nonlinear problems, in this section (except where noted) a simple approximate velocity $v(x, t)$ is used that is locally constant over each time step interval. Given a grid point x_i at time t^{n+1}, we use one step of a fourth order Runge-Kutta (RK4) scheme to approximate $\check{x}_i^n$, and then define

$$v(x_i, t) = \frac{x_i - \check{x}_i^n}{\Delta t}, \quad t^n \le t \le t^{n+1}. \tag{4.1}$$

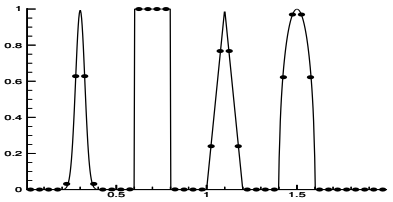
Using this velocity, then, the true traceline in space-time is approximated by a linear path joining (x_i, t^{n+1}) to $(\check{x}_i^n, t^n)$.

4.1. Example 1: Constant velocity

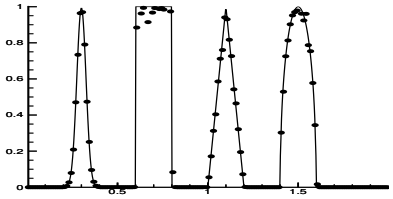
We first test our scheme on a standard test case, called *Shu's linear test*, which uses a nontrivial initial condition and a constant velocity $a = 1$. Since the traceback region can be found exactly, even with our approximate velocity function (4.1) (i.e., $v = a = 1$), we perturb the traceback points randomly with a uniform distribution in $[-0.2h, 0.2h]$ so that there is some nonzero flux across the tracelines. We see from Fig. 2 that the scheme loses all accuracy if no flux correction is applied. However, we recover the solution by including the flux correction along the v-trace, even though we have severely inaccurate predictions of the traceback regions.



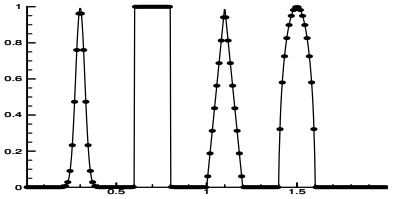
$m = 40$, 1 time step, no flux computation.



$m = 40$, 1 time step, with flux computation



$m = 160$, 4 time steps, no flux computation.



$m = 160$, 4 time steps, with flux computation.

Figure 2: Ex. 1, Shu's linear test. The traceback regions are perturbed uniformly in $[-0.2h, 0.2h]$ and m is the number of grid elements used. Without the flux computation, the solution loses its original shape. However, the flux computation recovers the correct solution.

4.2. Example 2: $a = \sin(x)$

We next test our scheme using a spatially varying $a(x,t) = \sin(x)$ over $[0, 2\pi]$, with the exact solution

$$u(x,t) = \frac{\sin\left(2\arctan(e^{-t}\tan(x/2))\right)}{\sin(x)}.$$

The initial time is 0.1 and the final time is $T = 1$.

As a point of reference, with exact tracing of the point $\check{x}_i$ in (4.1), there is no flux error, and the results show good fifth order convergence, both for a fixed number of 20 time steps ($\Delta t = 1/20$, so the Courant number varies from $1/2\pi = 0.16$ to $16/\pi = 5.09$) and a variable time step of $\Delta t = 5h$ (Courant number 5).

We now consider approximate tracing of $\check{x}_i$, so there will be some flux error. We use a fourth order Runge-Kutta (RK4) over the entire time step to solve (3.2), which turns out to do a relatively poor job for this problem. In the first data sets of Tables 1–2 we report results that do not correct the flux. Tables 1 uses a fixed 20 time steps for the simulation, whereas Table 2 uses a variable $\Delta t = 5h$ (for the first two data sets). The solutions are not too bad when the mesh is coarse; however, as we refine the mesh, the solutions do not converge to the correct order, since $\check{x}_i$ becomes progressively worse with no flux correction.

When the flux correction technique is applied, we see in the second data sets of Tables 1–2 good fifth order convergence. It is thus crucial to apply the flux computation when the traceback region is not computed accurately. In general, we do not expect the traceback region to be accurately determined, and so the use of the flux computation gives an accurate recovery of the solution. The test with a fixed Δt (Table 1) is accurate because the velocity is independent of time, and so in this example, time step refinement is not critical for accuracy of the tracing and flux correction steps.

We now turn to a set of tests that use a parabolic approximation of the v-tracelines, rather than straight lines. That is, here we approximate $v(x,t)$ in the definition of the trace (3.2) not by (4.1) but by a linear function in time. We accomplish this by using two steps of RK4. The parabola is chosen to pass through the initial grid point x_i at time t^{n+1}, the output from the first step of $RK4$, the point $\check{x}_i^{n+1/2}$ at time $t^{n+1/2}$, and the final output of the second step of RK4, the point $\check{x}_i^n$ at time t^n. That is,

$$\check{x}(x_i;t) = x_i + \frac{4\check{x}_i^{n+1/2} - \check{x}_i^n - 3x_i}{\Delta t}(t^{n+1} - t) + 2\frac{\check{x}_i^n + x_i - 2\check{x}_i^{n+1/2}}{\Delta t^2}(t^{n+1} - t)^2,$$

Table 1: Ex. 2: $a = \sin(x)$. Error and convergence order at $T = 1$ with 20 time steps and $h = 2\pi/m$. We use either a linear v-traceline (4.1) or a quadratic v-traceline (4.2), and we do not apply the flux correction for the first set of data.

m	L_h^1 error	order	L_h^∞ error	order
linear v-traceline, no flux correction				
20	9.43337E-03	——	9.66783E-03	——
40	5.74445E-04	4.038	6.78784E-04	3.832
80	2.00448E-05	4.841	2.24237E-05	4.920
160	5.92988E-07	5.079	7.28289E-07	4.944
320	9.14134E-08	2.698	9.33877E-08	2.963
640	9.03774E-08	0.016	8.05034E-08	0.214
linear v-traceline, with flux correction				
20	9.43276E-03	——	9.66501E-03	——
40	5.74441E-04	4.037	6.78144E-04	3.833
80	1.99823E-05	4.845	2.23159E-05	4.925
160	5.37348E-07	5.217	6.50065E-07	5.101
320	1.04488E-08	5.684	1.73898E-08	5.224
640	2.62322E-10	5.316	5.59758E-10	4.957
quadratic v-traceline, with flux correction				
20	9.43331E-03	——	9.66782E-03	——
40	5.74396E-04	4.038	6.78791E-04	3.832
80	2.00074E-05	4.843	2.23526E-05	4.924
160	5.39046E-07	5.214	6.54327E-07	5.094
320	1.07331E-08	5.650	1.79475E-08	5.188
640	2.05330E-10	5.708	4.65195E-10	5.270

and so

$$v(t) = -\frac{4\check{x}_i^{n+1/2} - \check{x}_i^n - 3x_i}{\Delta t} - 4\frac{\check{x}_i^n + x_i - 2\check{x}_i^{n+1/2}}{\Delta t^2}(t^{n+1} - t). \tag{4.2}$$

The results are given in the third data set of Table 1, using a fixed 20 time steps for the simulation. The results are comparable to, but slightly better than those using a linear v-traceline. The results using a variable $\Delta t = 10h$ appear in the third data set of Table 2. In spite of using twice the Courant number of the linear v-traceline results, we see better results for the quadratic v-traceline. In fact, results in the third set of Table 2 are at the same range of accuracy as those in [19, Tables 6.3–6.4], which compute the solution using an exact tracing (and so no flux correction is needed).

4.3. Example 3: $a = \sin(t)$

We now test a time-varying velocity $a(x,t) = \sin(t)$ on $[0,2]$ up to time $T = 4$, for which the exact solution is

$$u(x,t) = u_0(x + 1 + \cos(t)), \quad \text{where} \quad u_0(x) = 0.75 + 0.25\sin(\pi x).$$

Using Courant number 20, from Table 3, we see a degradation in the convergence rate to about $O(h^4)$ due to incorrect tracing when the flux is uncorrected. However, as the table shows, the flux computation recovers the fifth order convergence rate.

When we use a fixed number of 20 steps in time, but increase the accuracy of the linear v-tracing (4.1) by using m steps of fourth order Runge-Kutta instead of only one, we do not see the fifth order convergence

Table 2: Ex. 2: $a = \sin(x)$. Error and convergence order at $T = 1$ with $\Delta t = 5h$ or $10h$ and $h = 2\pi/m$. We use either a linear v-traceline (4.1) or a quadratic v-traceline (4.2). We do not apply the flux correction for the first set of data, and we use the longer time step for the third data set.

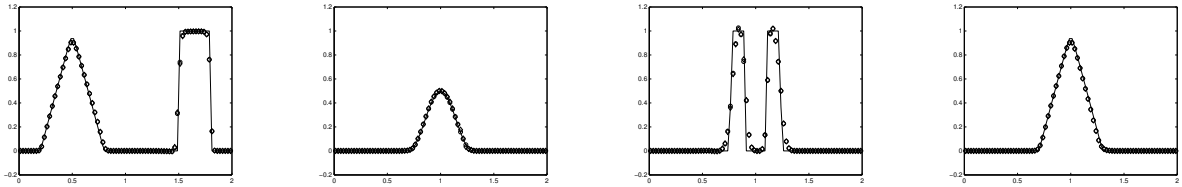
m	L_h^1 error	order	L_h^∞ error	order
$\Delta t = 5h$, linear v-traceline, no flux correction				
20	2.30710E-01	——	8.02471E-02	——
40	7.74127E-03	4.897	3.37387E-03	4.572
80	4.44777E-04	4.121	2.99323E-04	3.495
160	2.84807E-05	3.965	2.27471E-05	3.718
320	1.98369E-06	3.844	1.71154E-06	3.732
640	1.25728E-07	3.980	1.11553E-07	3.940
$\Delta t = 5h$, linear v-traceline, with flux correction				
20	6.52335E-02	——	4.54550E-02	——
40	7.69580E-04	6.405	5.32557E-04	6.415
80	1.87596E-05	5.358	1.76413E-05	4.916
160	4.03245E-07	5.540	5.34048E-07	5.046
320	1.23639E-08	5.027	1.69982E-08	4.974
640	3.43488E-10	5.170	5.44508E-10	4.964
$\Delta t = 10h$, quadratic v-traceline, with flux correction				
20	5.68699E-01	——	3.43370E-00	——
40	1.10144E-02	5.690	8.87433E-03	8.596
80	1.43537E-04	6.262	7.59726E-05	6.868
160	2.10872E-06	6.089	1.03687E-06	6.195
320	3.47400E-08	5.924	1.82299E-08	5.830
640	5.59844E-10	5.955	3.45997E-10	5.719

Table 3: Ex. 3: $a = \sin(t)$. Errors and convergence order at $T = 4$ with $\Delta t = 20h$ and $h = 2/m$, using a linear v-traceline (4.1). We do not apply the flux correction.

m	L_h^1 error	order	L_h^∞ error	order
no flux correction				
20	1.03193E-02	——	8.16950E-03	——
40	5.91701E-04	4.124	4.63738E-04	4.139
80	3.61528E-05	4.033	2.83856E-05	4.030
160	2.24693E-06	4.008	1.76479E-06	4.008
320	1.40242E-07	4.002	1.10149E-07	4.002
640	8.76233E-09	4.000	6.88191E-09	4.001
with flux correction				
20	1.17491E-01	——	1.05662E-01	——
40	5.65872E-04	7.698	4.43710E-04	7.896
80	1.01980E-05	5.794	8.00549E-06	5.792
160	1.70051E-07	5.906	1.33576E-07	5.905
320	2.73038E-09	5.961	2.14476E-09	5.961
640	4.31873E-11	5.982	3.39493E-11	5.981

(a) Cell average values. (b) Contour plot.

Figure 3: Ex. 4, $g(t) = 2\cos(\pi t/T)$. Second order Strang splitting solution at $T = 1.5$ using $\Delta t = 8h$ and $h = 1/40$.



(a) $x = 1 + h/2$. (b) $x = 0.5 + h/2$. (c) $y = 1.3 + h/2$. (d) $y = 0.5 + h/2$.

Figure 4: Ex. 4, $g(t) = 2\cos(\pi t/T)$. Cross-sections of the numerical solution at (a) $x = 1 + h/2$, (b) $x = 0.5 + h/2$, (c) $y = 1.3 + h/2$, and (d) $y = 0.5 + h/2$. The circles are the solution without flux computation, the diamonds are the new scheme's solution, and the true solution is the solid line.

rate in this case. Time error dominates after about $m = 80$, since there is flux error in the direction of time. In the previous Ex. 2, we were able to use a fixed number of time steps, because $a(x, t)$ in that example was independent of time. The use of a quadratic v-traceline (4.2) should improve the solution, but its convergence will stall out eventually as well.

4.4. Example 4: Swirling and deforming flow

A more severe test is obtained by using a two-dimensional, swirling and deforming flow. Following [21], we take the velocity in the form

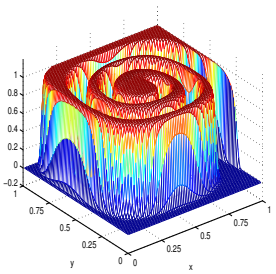
$$a_1(x, y) = \sin^2\left(\frac{\pi x}{2}\right)\sin(\pi y)\, g(t), \quad a_2(x, y) = -\sin^2\left(\frac{\pi y}{2}\right)\sin(\pi x)\, g(t). \tag{4.3}$$

This flow satisfies $a_1 = a_2 = 0$ on the boundaries of the domain $(0, 2) \times (0, 2)$. The function $g(t)$ is used to introduce time dependence in the flow field, and we use $g(t) = 2\cos(\pi t/T)$ on the time interval $0 \le t \le T$. The flow slows down and reverses direction at time $T/2$ so that the initial data is recovered at time T. We use $T = 1.5$. The initial condition includes a slotted disk, a cone, and a "smooth" hump, similar to that used by LeVeque [21].

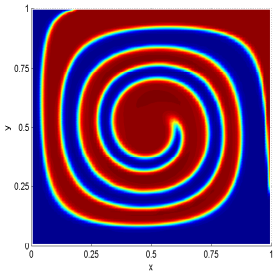
We use a second order Strang splitting in space to solve the problem (see [19]), using $\Delta t = 8h$ and $h = 1/40$, which has a Courant number of 8 in each spatial direction. The result is shown in Fig. 3. There is very little difference between the results with and without flux computation. This suggests that our choice of the approximate velocity $v(x, t)$ given in (4.1) is quite reasonable. However, we can still find some minor improvement in Fig. 4, which shows several cross-sections of the solution. The flux computation improves the solution a little, but because the predecessor sets are found very accurately, the flux errors are very small.

As a final test, we scale the problem to the unit square and take $g(t) \equiv 1$, so the flow merely rotates. The initial condition is

$$u(x, y, 0) = \begin{cases} 1 & \text{if } (x-1)^2 + (y-1)^2 < 0.8^2, \\ 0 & \text{otherwise.} \end{cases}$$



(a) Cell average values.



(b) Contour plot.

Figure 5: Ex. 4, $g(t) = 1$. Solution at time $t = 2.5$ using $\Delta t = 8h$ ($\text{CFL}_{\Delta t} = 8$).

Fig. 5 shows the computed solution at time $t = 2.5$ using $\Delta t = 8h$ and $h = 1/80$. The level of numerical diffusion on this 80×80 grid is extremely low, since it takes only 25 steps to reach the final time.

5. Application to Nonlinear Conservation Laws

It remains to make a choice for $v(x,t)$. We are actually solving the perturbed equation (1.3), so the natural choice for v would seem to be to minimize the characteristic speed of this equation. That is, we might take $v(x,t) = f'(u)$, approximated by some fixed $u(x,t)$. For simplicity, we take the constant velocity

$$v_{\text{characteristic}}(x_i,t) := f'(\bar{u}_i^n), \quad t^n \le t \le t^{n+1}. \tag{5.1}$$

Since u is in fact constant along a characteristic, this choice should give the smallest error in the flux correction step.

However, the scheme will collapse when two characteristics collide, and so we will have a CFL constraint. But note that this CFL constraint is more relaxed than the usual one for a scheme on a fixed Eulerian grid. The usual CFL constraint limits the time step to the time when the characteristic at a grid point x_i arrives the next grid point x_{i+1} or x_{i-1}, i.e., $\Delta t \le \dfrac{h}{\max|f'(u)|}$. But for our scheme, the CFL constraint is essentially the one for (1.3). This velocity perturbation was also used by Stockie, Mackenzie, and Russell [31] for their moving mesh method (see, e.g., [31, (2.2)]). While these authors change the mesh for time t^{n+1}, we fix the mesh but find the proper traceback region $\check{E}$ at time t^n. The new CFL constraint that they identify in [31, (2.7)] is the same as the CFL constraint needed by our scheme. However, compared to their scheme, we do not need to deal with a true moving mesh, and we have developed a high order scheme. To be precise, we can state the relaxed CFL condition as

$$\Delta t \le \Delta t_{\text{CFL}} := \frac{h}{\max|f'(u) - v|}. \tag{5.2}$$

Following our presentation, there is another natural choice for $v(x,t)$, which is to take the particle velocity $v(x,t) = f(u)/u$, wherein $u(x,t)$ must again be fixed somehow. This gives the smallest relative particle velocity, and the smallest need for flux correction. Moreover, the relaxed CLF condition should allow longer time steps, which means that we accumulate less numerical diffusion due to projection back to the Eulerian grid. Although our numerical tests are perhaps not conclusive, this later choice seems to work better overall. This may be related to the fact that the tracelines never collide. With this choice, we define v at x_i to be simply the constant velocity

$$v_{\text{traceline}}(x_i,t) := \frac{f(\bar{u}_i^n)}{\bar{u}_i^n}, \quad t^n \le t \le t^{n+1}. \tag{5.3}$$

The choice of $v(x,t)$ depends on the solution of the given problem, and our numerical examples suggest that either choice provides a good solution, especially as compared to traditional fixed grid methods.

5.1. Example 5: Burgers' equation

Our first nonlinear example solves the simple Burgers' equation

$$u_t + (\tfrac{1}{2}u^2)_x = 0 \quad \text{and} \quad u(x,0) = u_0(x) = 0.5 + \sin(\pi x), \quad x \in [0,2]. \tag{5.4}$$

We first test the new scheme with the help of the scheme defined in [3]. That scheme has an implementable algorithm that finds relatively very accurate traceback points for nonlinear problems. It does so by solving a constrained optimization problem. Even though we use these traceback points, our new scheme is different from that scheme in two main ways. First, our new scheme is formally higher order accurate, since it uses a WENO reconstruction to evaluate the traceback integral. Second, the flux correction procedure is performed. We define $v(x_i,t)$ as the constant speed which gives the provided traceback point $\breve{x}_i^n$, as in (4.1).

Our results in Table 4 show the solution error at time $T = 0.25$ for both the case of omitting the flux correction and including it. Even though the traceback points are relatively accurate, the flux computation is needed to obtain high accuracy and the proper rate of convergence. In this test, the Courant number $\max(|f'(u)|\Delta t/h)$ is only 0.75.

Table 4: Ex. 5: Burgers' equation. Errors and convergence order at $T = 0.25$ with $\Delta t = 0.5h$ and $h = 2/m$. The traceback points are given by the optimization algorithm of [3].

m	L_h^1 error	order	L_h^∞ error	order
no flux correction				
20	6.14073E-03	——	5.73113E-02	——
40	2.83075E-03	1.117	1.57321E-02	1.865
80	1.87939E-03	0.591	1.26506E-02	0.315
160	8.67269E-04	1.116	7.15002E-03	0.823
320	4.13884E-04	1.067	3.45804E-03	1.048
with flux correction				
20	1.97619E-03	——	3.46260E-02	——
40	4.75684E-04	2.055	5.15510E-03	2.748
80	1.26027E-04	1.916	3.19050E-03	0.692
160	1.13948E-05	3.467	2.65091E-04	3.589
320	4.82000E-07	4.563	1.51346E-05	4.131

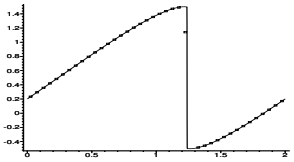
We now test our scheme using $v(x_{i+1},t) = \bar{u}_i^n = f'(\bar{u}_i^n)$ to find the predecessor point for x_{i+1}, which is somewhat upstream weighted. Before a shock forms at time $1/\pi$, no characteristics collide, and we see a very clean fifth order of convergence in the first data set in Table 5 using $\Delta t = 5h$ (Courant number 7.5).

After the shock forms at time $1/\pi$, we need to reduce the time step. We reran the test using $\Delta t = 2h$ (Courant number 3), which satisfies the relaxed CFL condition (5.2), since any consecutive pair of traceback points do not collide in the simulation. A good order of convergence was observed in the second data set in Table 5. Comparing to using $\Delta t = 5h$, we verify that the longer time step does indeed produce the more accurate results for smooth solutions, as is typical of Lagrangian methods [4]. In Fig. 6 we see that the shock forms cleanly and propagates properly to time $T = 3/(2\pi)$.

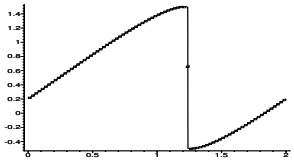
We remark that without the flux correction, these tests with $\Delta t = 5h$ and $2h$ (and even with a smaller $\Delta t = 0.2h$) fail to continue because the characteristics collided, due to an incorrect approximation of u as the errors accumulate.

Table 5: Ex. 5: Burgers' equation. Errors and convergence order at $T = 0.25$ with $h = 2/m$ and $\Delta t = 5h$ or $2h$. The traceback points are computed using $v(x_{i+1}, t) = \bar{u}_i^n = f'(\bar{u}_i^n)$. Flux correction is applied.

m	L_h^1 error	order	L_h^∞ error	order
		$\Delta t = 5h$		
40	9.66590E-03	——	4.84538E-02	——
80	7.22025E-04	3.743	7.76105E-03	2.642
160	3.78096E-05	4.255	7.10940E-04	3.448
320	1.53392E-06	4.623	2.75992E-05	4.687
640	4.70951E-08	5.026	9.98311E-07	4.789
		$\Delta t = 2h$		
20	3.91754E-03	——	5.00787E-02	——
40	6.64697E-04	2.559	3.66442E-03	3.773
80	8.51260E-05	2.965	1.06433E-03	1.784
160	5.10336E-06	4.060	1.47945E-04	2.847
320	1.84403E-07	4.791	6.02406E-06	4.618
640	6.80441E-09	4.760	1.75723E-07	5.099



$m = 40$



$m = 80$

Figure 6: Ex. 5, Burgers' equation. The discrete and true solution u at time $T = 3/(2\pi)$ after the shock has formed and propagated, using $\Delta t = 2h$, $h = 2/m$, and $m = 40$ grid elements on the left and $m = 80$ on the right. The traceback points are computed using $v(x_{i+1}, t) = \bar{u}_i^n = f'(\bar{u}_i^n)$, and the flux correction is applied.

5.2. Example 6: Buckley-Leverett problem

In this example we solve the Buckley-Leverett problem, which has the flux function

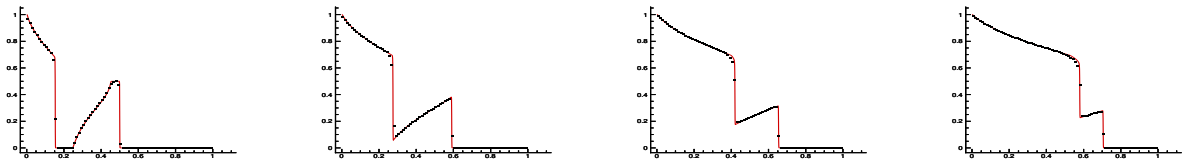
$$f(u) = \frac{u^2}{u^2 + (1-u)^2}.$$

The initial condition is

$$u_0(x) = \begin{cases} 1 - 20x & \text{for } 0 \le x \le 0.05, \\ 0.5 & \text{for } 0.25 \le x \le 0.4, \\ 0 & \text{otherwise,} \end{cases} \tag{5.5}$$

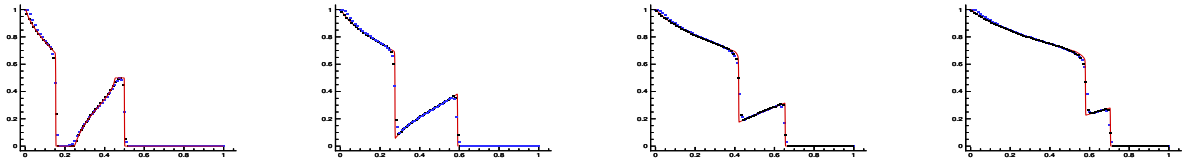
which is nonzero in two regions which merge over time.

The results using $m = 80$ grid elements and $\Delta t = 0.4h$ are shown in Fig. 7. We see that the nonzero regions develop shocks and rarefactions, and that these merge into each other appropriately. The results are very sharp, and superior to the nonWENO Eulerian-Lagrangian scheme of [3], both because the new scheme is higher order and because we can take longer time steps. A Godunov type of flux splitting is used for this problem. If no flux-splitting is used, the profiles look nearly identical; however, a small negative value of the solution develops at the trailing edge of the rarefaction fan (which is initially at $x = 0.25$). The flux splitting resolves this problem.



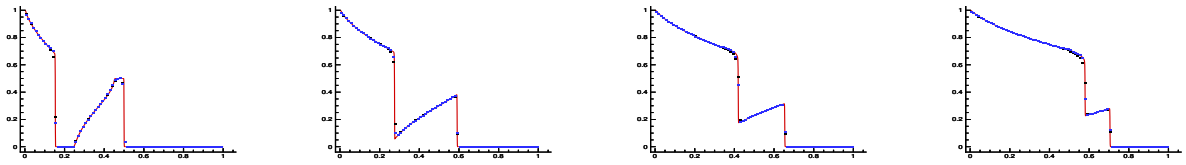
$t = 0.1$ $t = 0.2$ $t = 0.3$ $t = 0.4$

Figure 7: Ex. 6, Buckley-Leverett. The red line is the CWENO5 reference solution with $h = 1/1280$ and $\Delta t = 1/15360$. The black squares are the results using $m = 80$ grid elements and $\Delta t = 0.4h = 1/200$.



$t = 0.1$ $t = 0.2$ $t = 0.3$ $t = 0.4$

Figure 8: Ex. 6, Buckley-Leverett. The red line is the CWENO5 reference solution with $h = 1/1280$ and $\Delta t = 1/15360$. The black squares are the results using $m = 80$ grid elements and $\Delta t = h/6 = 1/480$. The blue squares are the CWENO5 results using the same m and Δt.

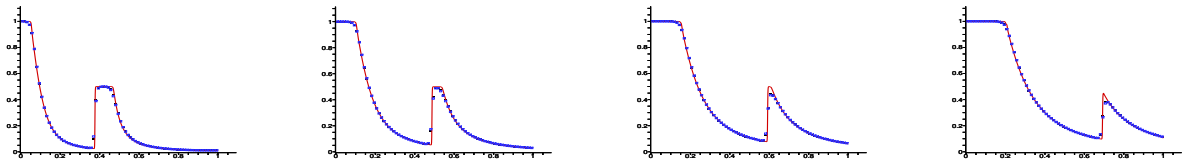


$t = 0.1$ $t = 0.2$ $t = 0.3$ $t = 0.4$

Figure 9: Ex. 6, Buckley-Leverett. The red line is the CWENO5 reference solution with $h = 1/1280$ and $\Delta t = 1/15360$. The black squares are the results using $m = 80$ grid elements and $\Delta t = 0.4h$ with the choice $v = f'(\bar{u}_i^n)$. The blue squares are the results using the same $m = 80$ but $\Delta t = 0.6h$ with the choice $v = f(\bar{u}_i^n)/\bar{u}_i^n$.

Eulerian-Lagrangian methods can use longer time steps than Eulerian methods. Since the use of longer time steps give results with less numerical diffusion, the solution is improved. It is instructive to compare our new scheme to, say, an Eulerian CWENO5 scheme when both use the same time-step size. That is, we take a smaller time step that is necessary in our new scheme to assess the build-up of numerical diffusion versus CWENO5. The results, using $m = 80$ grid elements and $\Delta t = h/6 = 1/480$, are shown in Fig. 8. Indeed, we see that the shorter time step has degraded the solution of the new scheme a bit as compared to Fig. 7. However, the result is a bit better than CWENO5. Essentially CWENO5 is much like our new scheme but using $v = 0$, and so these results suggest that the use of Lagrangian techniques (nonzero v) is helpful in itself in obtaining a good solution, and not merely because we can take a longer time step.

Since $\max |f'(u)| = 2$ (when $u = 0.5$), the Eulerian CFL constraint is $\Delta t \le 0.5h$. It is difficult to determine the relaxed Eulerian-Lagrangian CFL constraint (5.2), since v is defined approximately and there is a discontinuity in the solution for this example. In the worst case, it is perhaps possible that $\max |f'(u) - v(\bar{u}^n)| = 2$ is no better, since $v = 0$ is possible. Nevertheless, since $\max |f(u)/u| = 1/(2\sqrt{2} - 2) \approx 1.20711$ when $u = 1/\sqrt{2}$, we should expect that the relaxed CFL constraint for the choice $v = f(u)/u$ is more like $\Delta t \le 2(\sqrt{2} - 1)h \approx 0.8284h$. This choice should allow longer time steps than the choice $v(u) = f'(u)$. We illustrate this expectation in Fig. 9, which shows results using $v = f'(\bar{u}_i^n)$ with $\Delta t = 0.4h$ and $v = f(\bar{u}_i^n)/\bar{u}_i^n$ with the longer $\Delta t = 0.6h$. The latter produces somewhat better results, since the time step is longer.



$t = 0.1$ $t = 0.2$ $t = 0.3$ $t = 0.4$

Figure 10: Ex. 7, Non-convex flux. The red line is the CWENO5 reference solution with $h = 1/1280$ and $\Delta t = 1/15360$. The black squares are the results using $m = 160$ grid elements, $h = 2/m$, $\Delta t = 0.08h$, and $v = f'(\bar{u}_i^n)$. The blue squares are the results using the same $m = 160$, smaller $\Delta t = 0.05h$, and $v = f(\bar{u}_i^n)/\bar{u}_i^n$. The former result is slightly better.

5.3. Example 7: A non-convex flux

We next present an example with the non-convex flux function

$$f(u) = \begin{cases} 10u, & \text{if } u \le 0.01, \\ \sqrt{u}, & \text{otherwise,} \end{cases}$$

and the initial condition (5.5) over the interval $x \in [0, 2]$. Since this flux function has a characteristic speed that is slower than the traceline speed, one might expect to observe better results when choosing $v = f'(u)$. This is the case, since we can use a longer time step in the former case ($\Delta t = 0.08h$ versus $\Delta t = 0.05h$). The results appear in Fig. 10.

5.4. Example 8: 2-D Burgers' equation

The final example in this section is a simple two dimensional Burgers' equation

$$u_t + (u^2/2)_x + (u^2/2)_y = 0, \quad x \in [0, 2],\ y \in [0, 2],\ t > 0, \tag{5.6}$$

with the initial condition

$$u(x, y, 0) = \begin{cases} \sin^2(\pi x)\sin^2(\pi y), & \text{for } (x, y) \in (0, 1)^2, \\ 0, & \text{otherwise.} \end{cases}$$

The problem is solved using a second order Strang splitting in space and the velocity v based on the characteristic velocity u at time t^n. Details of the Strang splitting can be found in [19]. Briefly, within (say) the x-sweep of the Strang splitting algorithm, we first need to reconstruct u to high order in the y-direction at the Gauss points, as $\mathcal{R}(\bar{u}^n)$ (to be more precise, we use the reconstructed value [19, (5.11)]). This is then the x-average value used to set $v = \mathcal{R}(\bar{u}^n)$. The solution, with $m = 80$ grid elements in each direction and $\Delta t = h$, is shown in Fig. 11 at times $t = 0$, 1, 2, and 3. The solution shows no distortion due to the Strang splitting.

6. Application to the Euler System

For a polytropic gas, the energy is $E = p/(\gamma - 1) + \rho u^2/2$, where p, ρ, and u are the particle pressure, density, and velocity, and the adiabatic index $\gamma = (f + 2)/f = 1.4$, where $f = 5$ is the number of degrees of freedom of each gas particle. The one-dimensional dynamics is described by the Euler equations

$$\begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E + p) \end{pmatrix}_x = 0 \quad \Longleftrightarrow \quad \boldsymbol{\xi}_t + (\mathbf{f}(\boldsymbol{\xi}))_x = 0, \tag{6.1}$$

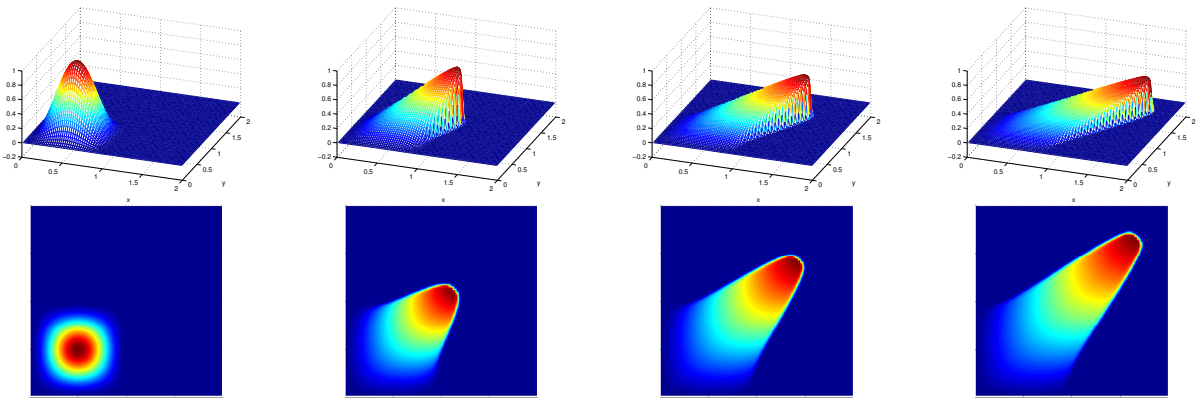


Figure 11: Ex. 8, 2-D Burgers'. Solution using $m = 80$ grid elements in each direction and $\Delta t = h$.

where $\boldsymbol{\xi} = (\rho, \rho u, E)$.

Since our EL-WENO schemes are explicit schemes, we can treat the vector valued equations componentwise. Since each conserved unknown quantity travels along a different path, we might choose three independent fixed trace velocities v_i, $i = 1, 2, 3$, one for each conserved quantity ρ, $m = \rho u$, and E, respectively. Then the ith conserved unknown along the ith traceline velocity is governed by the full system (6.1) rewritten in terms of the diagonal trace matrix $V(v_1, v_2, v_3) = \begin{pmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & v_3 \end{pmatrix}$ as

$$\boldsymbol{\xi}_t + (V(v_1, v_2, v_3)\boldsymbol{\xi})_x + (\mathbf{f}(\boldsymbol{\xi}) - V(v_1, v_2, v_3)\boldsymbol{\xi})_x = 0. \tag{6.2}$$

However, it is computationally difficult to account for the interactions of the quantities in this situation. We would need to account for information at time t^{n+1} coming from multiple spatial locations at time t^n. Moreover, particles should have interacted with each other at intermediate times along the way. It seems the only reasonable choice is to select a single approximate traceline velocity v.

We might continue to select three independent traceline velocities, using v_i for the ith conserved quantity, but accounting for the interactions between the quantities by now evolving all quantities along this same traceline. That is, solve three Euler systems, one for each conserved quantity $i = 1, 2, 3$, by applying our scheme to each component equation, as in Section 3, to

$$\boldsymbol{\xi}_t + (V(v_i)\boldsymbol{\xi})_x + (\mathbf{f}(\boldsymbol{\xi}) - V(v_i)\boldsymbol{\xi})_x = 0, \tag{6.3}$$

where $V(v) = V(v, v, v)$. We call this strategy the simultaneous solution method. It requires us to triple our computational cost because we solve the same Euler system three times. For computational efficiency, it seems that we should use the same v for all three component systems, i.e., to take $v = v_1 = v_2 = v_3$.

When implementing the method, we need several WENO reconstructions. For this part of the computation, it is usual to respect the characteristic structure of the system by reconstructing the needed information using the linearized system. We expand (6.1) into a system of the form $U_t + A(U)U_x = 0$, that is, into

$$\begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}_t + \begin{pmatrix} 0 & 1 & 0 \\ \frac{1}{2}(\gamma - 3)u^2 & (3 - \gamma)u & \gamma - 1 \\ \frac{1}{2}(\gamma - 1)u^3 - \dfrac{u(E + p)}{\rho} & \dfrac{(E + p)}{\rho} - (\gamma - 1)u^2 & \gamma u \end{pmatrix} \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}_x = 0.$$

The linearized equation can be diagonalized as $\Lambda^0 = L^0 A(U^0)(L^0)^{-1} = V(u^0 - c^0, u^0, u^0 + c^0)$, giving

$$L^0 \boldsymbol{\xi}_t + V(u^0 - c^0, u^0, u^0 + c^0) L^0 \boldsymbol{\xi}_x = 0,$$

where $c = \sqrt{(\gamma p)/\rho}$ is the sound speed. All WENO reconstructions described in Sections 2 and 3 are computed in this local characteristic decomposition using the variables $L^0\boldsymbol{\xi} = L^0(\rho, \rho u, E)$, which do not interact in the linearized system.

As in Section 2.2, we have left and right values at a reconstruction point. We use the numerical flux of a Roe solver for the flux correction functions

$$\begin{pmatrix} g_1(\rho, \rho u, E) \\ g_2(\rho, \rho u, E) \\ g_3(\rho, \rho u, E) \end{pmatrix} = \begin{pmatrix} \rho u - v\rho \\ \rho u^2 + p - v\rho u \\ u(E + p) - vE \end{pmatrix} \tag{6.4}$$

before we compute the two integrals of the flux in (3.4) over the space-time paths $\mathcal{S}_{E,L}$ and $\mathcal{S}_{E,R}$. The Roe numerical flux is defined as

$$\begin{pmatrix} g_1^{\text{Roe}} \\ g_2^{\text{Roe}} \\ g_3^{\text{Roe}} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} g_1^+ + g_1^- \\ g_2^+ + g_2^- \\ g_3^+ + g_3^- \end{pmatrix} + \frac{1}{2} |A| \begin{pmatrix} \rho^+ - \rho^- \\ \rho^+ u^+ - \rho^- u^- \\ E^+ - E^- \end{pmatrix}, \tag{6.5}$$

where $g_j^\pm = g_j(\rho^\pm, \rho^\pm u^\pm, E^\pm)$ and $|A| = L^{-1} V(|u^0 - c^0|, |u^0|, |u^0 + c^0|) L$. Note that $\rho^\pm$, $u^\pm$, and $E^\pm$ in each g_j are evolved along the v-trace path.

We need to make a choice for the trace velocity v. Since we insist on using a single trace velocity, we must approximate (6.3) so that $\mathbf{f}(\boldsymbol{\xi}) \approx V(v_i)\boldsymbol{\xi}$. Using the linearization described above, we have that

$$\boldsymbol{\xi}_t + \left(V(v)\boldsymbol{\xi}\right)_x + \left((L^0)^{-1} V(u^0 - c^0, u^0, u^0 + c^0) L^0 \boldsymbol{\xi} - V(v)\boldsymbol{\xi}\right)_x \approx 0$$

Using that $L^0 V(v) = V(v) L^0$ since $V(v)$ is a multiple of the identity, we have that

$$L^0 \boldsymbol{\xi}_t + \left(V(v)\, L^0 \boldsymbol{\xi}\right)_x + \left([V(u^0 - c^0, u^0, u^0 + c^0) - V(v)] L^0 \boldsymbol{\xi}\right)_x \approx 0, \tag{6.6}$$

The natural choice might be to take v as one of $v_1 = u^0 - c^0$, $v_2 = u^0$, and $v_3 = u^0 + c^0$; that is, to take the natural characteristic velocities of the system. From the numerical results later, it seems that $v = u$ is perhaps the best choice, although we are not certain why this is the case. Mathematically, this choice is the average of the three velocities, and so it is the most neutral single trace velocity v for the entire system. However, this is also the velocity of the mass particles, which must be correctly evolved in a physical sense to have an accurate solution. It is a happy coincidence that these two points of view give the same traceline velocity $v = u$.

Below we test numerically the use of several choices of the trace velocity v, as well as test the simultaneous solution method requiring three times the computational effort. For a single trace velocity, we test the main choices $v = u$, $v = u - c$, and $v = u + c$. We also test a cyclic variant in which we follow all three characteristics, but not tripling the workload, by using $v = u - c$ in one time step, $v = u$ in the next, and finally $v = u + c$ in the third time step, cyclically in time.

6.1. Example 9: A smooth problem for the Euler equations

In this example, the initial condition is set to be $\rho(x, 0) = 1 + 0.2\sin(\pi x)$, $u(x, 0) = 1$, $p(x, 0) = 1$, with 2-periodic boundary conditions. The exact solution is $\rho(x, t) = 1 + 0.2\sin(\pi(x - t))$, $u = 1$, $p = 1$. We compute the solution up to time $T = 2$ using $v = u$. The numerical errors and convergence orders for the density is given in Table 6, along with results for the standard finite volume WENO5 scheme. We see fifth order convergence for this Euler problem. The EL-WENO scheme produces smaller errors than WENO5, except for the coarsest mesh.

Table 6: Ex. 9, Euler equation. Error and convergence order at $T = 2$ for EL-WENO and WENO5 using $\Delta t = 0.1h$.

m	L_h^1 error	order	L_h^∞ error	order
EL-WENO				
20	8.01638E-03	——	6.21031E-03	——
40	1.70116E-05	8.880	1.67663E-05	8.533
80	5.29924E-07	5.005	5.29640E-07	4.984
160	1.63992E-08	5.014	1.49767E-08	5.144
WENO5				
20	1.12975E-03	——	9.00365E-04	——
40	3.58933E-05	4.976	3.29523E-05	4.772
80	1.14455E-06	4.971	1.06582E-06	4.950
160	4.03128E-08	4.827	4.07652E-08	4.708

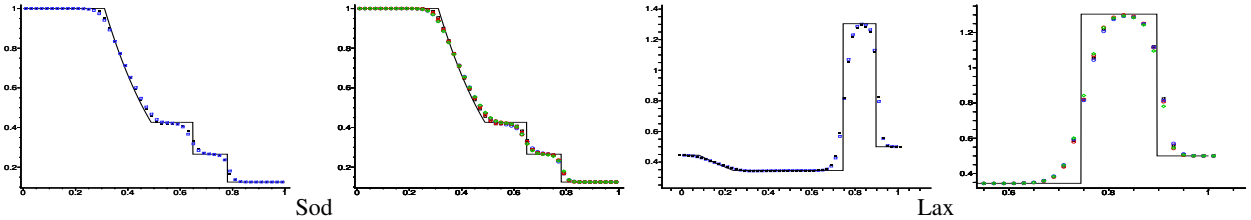


Figure 12: Ex. 10, 1-D shock tube tests. The density profile for the Sod and Lax tests at time $T = 0.16$ using $\Delta t = 0.1h$ and $m = 50$ grid elements. On the left of each set, we provide EL-WENO results using $v = u$ (black squares) and WENO5 (blue squares). The quality of the two numerical solutions are comparable. On the right of each set, we provide EL-WENO results using $v = u + c$ (blue circles), $v = u$ (black squares), $v = u - c$ (red circles) and $v = u + c$, $v = u$, and $v = u - c$ cyclically (red squares), and finally $v_1 = u - c$, $v_2 = u$, $v_3 = u + c$ simultaneously (green diamonds).

6.2. Example 10: Riemann problems for the Euler equations

For this series of one-dimensional shock tube tests, we specify a discontinuous initial condition, written in terms of the primitive variables ρ, u, and p. As is typical, we only report the density ρ; the other variables show comparable accuracy. The one-dimensional shock tube test of Sod uses the initial condition

$$\rho, u, p = \begin{cases} \rho_l = 1,\ u_l = 0,\ p_l = 1, & \text{for } x < 1/2, \\ \rho_r = 1/8,\ u_r = 0,\ p_r = 1/10, & \text{for } x > 1/2, \end{cases}$$

and the test of Lax uses the initial condition

$$\rho, u, p = \begin{cases} \rho_l = 0.445,\ u_l = 0.698,\ p_l = 3.528, & \text{for } x < 1/2, \\ \rho_r = 0.5,\ u_r = 0,\ p_r = 0.571, & \text{for } x > 1/2. \end{cases}$$

The results shown in Fig. 12 were obtained using $m = 50$ grid elements. The quality of the numerical solution from the EL-WENO scheme using $v = u$ is comparable to WENO5 results. For EL-WENO, we tested five different ways of choosing v. They are using v as each of the three eigenvalues $u + c$, u, and $u - c$; the cyclic use of v chosen to be $u + c$, u, and $u - c$; and finally the use of $v_1 = u - c$, $v_2 = u$, and $v_3 = u + c$ simultaneously (which requires three times the computational work). The results indicate that using $v = u$ is the better choice for both the Sod and Lax tests.

The EL-WENO results in Figs. 13 show the effect of grid refinement. The results are obtained using $v = u$ for all three equations.

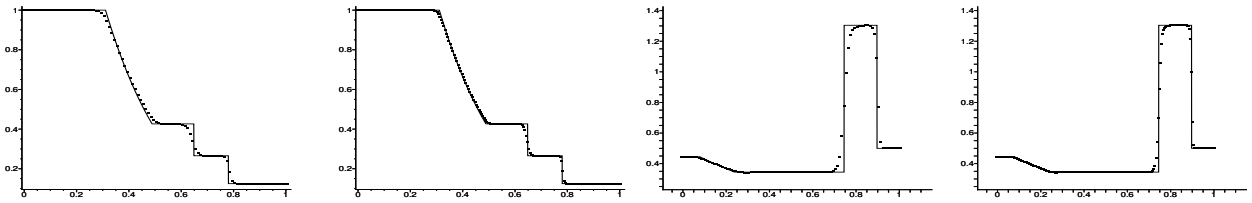


Figure 13: Ex. 10, Sod and Lax 1-D shock tube tests. The density profile for EL-WENO with $v = u$ at time $T = 0.16$ using $\Delta t = 0.1h$ and $m = 100$ and 200 grid elements.

Figure 14: Ex. 10, Lax's 1-D shock tube test. The density profile at the maximum allowable time step for EL-WENO with $v = u$ using $\Delta t = 0.38h$ (left) and WENO5 using $\Delta t = 0.19h$ (right) at time $T = 0.16$ using $m = 50$ grid elements.

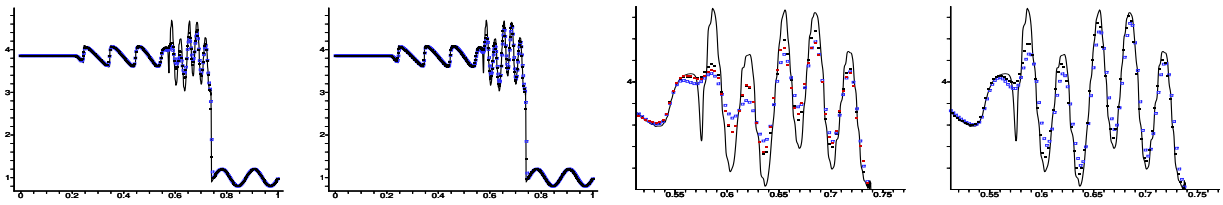
We were not able to use a very large time step in these problems. The reason seems to be that we trace only one of the three characteristic velocities. In Fig. 14, we compare our EL-WENO scheme with trace velocity $v = u$ to WENO5 for the Lax problem using $m = 50$ grid elements and the maximum time step that gives a stable (i.e., nonoscillatory) result. EL-WENO can use $\Delta t = 0.38h$ while WENO5 can use only $\Delta t = 0.19h$. By computing a finely discretized WENO5 solution, we found the accurate u and c for this problem, and thereby determined that the theoretical Eulerian CFL constraint is $\Delta t = h/|u + c| = 0.21h$. However, the EL-WENO scheme should be stable as long as the two untraced characteristic velocities $u \pm c$ do not travel across the space-time traceback region; that is, the velocity $\pm c$ relative to the trace velocity $v = u$ determines the CFL constraint. This implies a CFL limit for the Lagrangian scheme of about $\Delta t = h/|c| = 0.30h$. We could do a bit better in practice, but we believe that the failure to trace properly all three characteristic velocities fundamentally limits the time step in our EL-WENO scheme.

The Lax problem could be run with twice the time step of the Eulerian scheme. The corresponding CFL numbers for Sod's 1-D shock tube test are $\Delta t = 0.79h$ for EL-WENO and $\Delta t = 0.45h$ for WENO5, which is an increase of about 1.75 times. In practice, we saw that EL-WENO could use $\Delta t = 0.60h$ before exhibiting oscillation (1.33 times longer). We could increase this to $\Delta t = 0.75h$ (1.67 times longer) if we increased the re-averaging grid spacing parameter to $h^* = 1.25h$. In general the Sod problem is easier than the Lax problem, so it is not surprising that we see better improvement for the harder problem.

6.3. Example 11: Shu and Osher's shock interaction with entropy waves

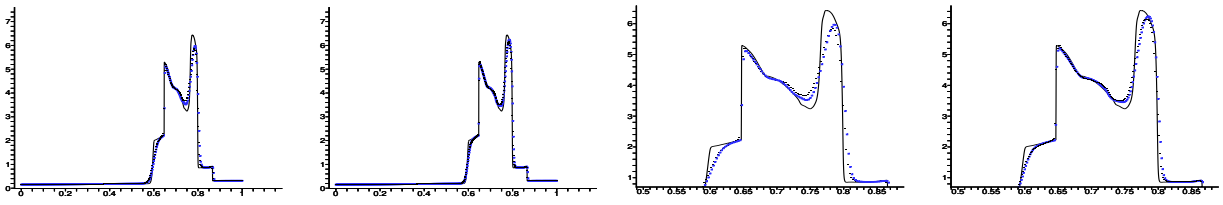
In our next example, we consider the challenging test case of Shu and Osher [30], in which a Mach 3 shock interacts with entropy sine waves in the density. We scale the problem to the domain $(0, 1)$, and the initial conditions are

$$\rho, u, p = \begin{cases} \rho_l = 3.857143,\ u_l = 2.629369,\ p_l = 10.333333, & \text{for } 0 < x < 1/10, \\ \rho_r = 1 + \epsilon \sin(5(10x - 5)),\ u_r = 0,\ p_r = 1, & \text{for } 1/10 \le x < 1, \end{cases}$$



$m = 300$ $m = 400$ $m = 300$, zoom $m = 400$, zoom

Figure 15: Ex. 11, Shu and Osher's test with a shock interacting with entropy waves. The density profile at time $T = 0.18$ using $m = 300$ and $m = 400$ grid elements, with a magnified view of the results shown on the right. Shown are results for EL-WENO using $v = u$ (black squares), EL-WENO using $v \in \{u - c, u, u + c\}$ cyclically (red squares), WENO5 (blue squares), and the fine resolution reference solution (solid line).



$m = 400$ $m = 600$ $m = 400$, zoom $m = 600$, zoom

Figure 16: Ex. 12, Woodward and Colella's double blast test. The density profile at time $T = 0.038$ using $m = 400$(left) and 600(right) grid elements, with a magnified view of the results shown on the right. Shown are results for EL-WENO (black squares), WENO5 (blue squares), and the fine resolution reference solution (solid line).

where $\epsilon = 0.2$. We use $\Delta t = 0.1h$ and compute to the final time $T = 0.18$. Using the mesh sizes $m = 300$ and $m = 400$ grid elements, the results appear in Fig. 15. Results for $m = 300$ grid elements show partial resolution of the entropy waves. The EL-WENO results are better than WENO5. Also results using $v = u$ are better than those of using $v \in \{u - c, u, u + c\}$ cyclically. Grids with $m = 400$ are sufficient to capture the entropy waves.

6.4. Example 12: Woodward and Colella's double blast test

Our final one-dimensional example is the double blast test of Woodward and Colella, which uses the initial condition

$$\rho, u, p = \begin{cases} \rho_l = 1,\ u_l = 0,\ p_l = 1000, & \text{for } x < 1/10, \\ \rho_m = 1,\ u_m = 0,\ p_m = 1/100, & \text{for } 1/10 < x < 9/10, \\ \rho_r = 1,\ u_r = 0,\ p_r = 100, & \text{for } 9/10 < x. \end{cases}$$

This is a more challenging problem. Nevertheless, reasonably good results are obtained by our EL-WENO scheme using $m = 400$ and 600 grid elements, as shown in Fig. 16, which look competitive to results for the WENO5 scheme. We see that EL-WENO results are better at the left peak near $x = 0.65$ and near $x = 0.6$ and $x = 0.8$. The EL-WENO results are not as good as WENO5 results near $x = 0.75$, however, EL-WENO improves when we refine the mesh to $m = 600$. The reference solution was computed by a MUSCL scheme with grid size $m = 4000$.

6.5. Example 13: 2-D Euler equation

In terms of $\xi = (\rho, \rho u, \rho v, E)^\top$, we solve the nonlinear system of Euler equations

$$\xi_t + f(\xi)_x + g(\xi)_y = 0, \tag{6.7}$$
$$f(\xi) = (\rho u, \rho u^2 + p, \rho uv, u(E + p))^\top, \quad g(\xi) = (\rho v, \rho uv, \rho v^2 + p, v(E + p))^\top.$$

Here ρ is the density, (u, v) is the velocity, E is the total energy, p is the pressure, which is related to the total energy by $E = \frac{p}{\gamma-1} + \frac{1}{2}\rho(u^2 + v^2)$ with $\gamma = 1.4$. The initial condition is set to be $\rho(x, y, 0) = 1 + 0.2\sin(\pi(x + y)), u(x, y, 0) = 0.7, v(x, y, 0) = 0.3, p(x, y, 0) = 1$, with a 2-periodic boundary condition. The exact solution is $\rho(x, y, t) = 1 + 0.2\sin(\pi(x + y(u + v)t))$, $u = 0.7$, $v = 0.3$, $p = 1$. We compute the solution up to time $T = 2$. The errors and numerical orders of accuracy for our Eulerian-Lagrangian WENO scheme appear in Table 7. We see fifth order convergence again for this 2-D Euler problem.

Table 7: Ex. 13, 2-D Euler. Error and convergence order at $T = 2$ for the EL-WENO schemes, $\Delta t = 0.5h$.

m	L_h^1 error	order	L_h^∞ error	order
20	1.19994E-03	——	4.87147E-04	——
40	3.76379E-05	4.995	1.78158E-05	4.773
80	1.16955E-06	5.008	5.55179E-07	5.004
160	3.58354E-08	5.028	1.58611E-08	5.129
320	1.05790E-09	5.082	4.59943E-10	5.108

6.6. Example 14: Double Mach reflection

Our final test problem is originally from [35]. The computational domain is $[0, 4] \times [0, 1]$. A reflecting wall lies at the bottom, starting from $x = 1/6$. Initially a right-moving Mach 10 shock is positioned at $x = 1/6, y = 0$ and makes a 60 degrees angle with the x-axis. For the bottom boundary, the exact postshock condition is imposed for the part from $x = 0$ to $x = 1/6$ and a reflective boundary condition is used for the rest. At the top boundary, the flow values are set to describe the exact motion of a Mach 10 shock. We compute the solution up to $T = 0.2$. In Fig. 17 we present the results of EL-WENO with 360×90 grid elements.

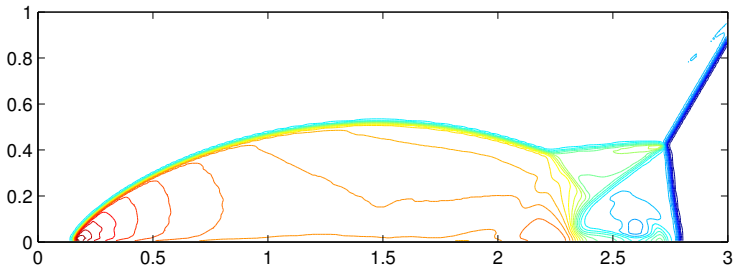


Figure 17: Ex. 13, Double Mach reflection. Density using a 360×90 grid plotted as thirty equally spaced density contours from 1.731 to 20.92.

7. Discussion and Conclusions

We presented a formally high order accurate Eulerian-Lagrangian WENO finite volume scheme for nonlinear scalar conservation laws that combines ideas of Lagrangian traceline methods with WENO reconstructions. Since it is not feasible to follow particles along the tracelines of the true velocity field $f(u)/u$, we instead follow the v-tracelines using a fixed, given velocity field $v(x, t)$. Particles are transported in the manner of a standard Eulerian-Lagrangian scheme using the velocity v. A flux correction computation is included in the scheme to account for particles that cross the v-traceline during the time step. The scheme reduces to an almost standard WENO5 scheme if v is chosen to be zero (the difference from a standard

scheme is that we chose to advance in time using the Runge-Kutta method combined with the Natural Continuous Extension). When v is chosen appropriately, the new scheme is an improvement, and in fact the relaxed CFL condition (5.2) results.

It is not clear what is the best choice for v. If v is chosen to approximate the characteristic velocity $f'(u)$, then the CFL constraint is relaxed as much as possible (see (5.2)). Moreover, the flux computation should be most accurate. However, if v is chosen to approximate the particle velocity $f(u)/u$, then no flux correction would be required and, in turn, no CFL constraint would apply.

Excellent numerical results were obtained for scalar problems in one and two space-dimensions using relatively long time steps. They showed that the flux correction step is critical to obtaining an accurate scheme, and also the need for a good choice for v. The numerical tests also demonstrated a unique feature of Eulerian-Lagrangian methods, which is that longer time steps give more accurate results, due to less build-up of numerical diffusion each time step [4]. Our scheme is computationally more expensive than a standard Eulerian scheme; however, it should be noted that we made no attempt at this stage of the work to optimize the scheme. Nevertheless, our scheme may be valuable in certain cases such as when a longer time step compensates for the extra work required, and when the physical problem dictates that one needs to minimize numerical diffusion.

We extended the method to systems of equations, in particular, the Euler system. It may not be computationally efficient to track each variable along its natural particle path, since all variables interact strongly. We found good results using various choices of the fixed, given velocity field v, and even combinations of different choices for v. In our numerical examples, the better results came from the average of the characteristic velocities, which is the velocity of the physical fluid particles. However, we were not able to take very long time steps for the Euler system. The spreading of conserved quantities due to multiple characteristic velocities enforces a CFL constraint on the method when only a single trace velocity v is used. This limited our Lagrangian approach to time steps only up to two times larger than Eulerian methods in our tests. That is, failure to trace properly all three characteristic velocities fundamentally limits the time step in an Eulerian-Lagrangian scheme. Further research is needed here.

We also described a general WENO technique that gives high order reconstruction of an integral of $u(x)$ from averages $\bar{u}_i$ of u defined on a general computational grid. These integrals in turn allow one to compute to high order local averages over any subinterval. Thus one can re-average $\bar{u}_i$, defined on the computational grid, to local averages on any given reconstruction grid. Further, these re-averaged values on the reconstruction grid can be used in any standard WENO reconstruction. In our case, the v-traceback point $\check{x}_i$ can be arbitrary, and linear WENO weights may not exist for that point. By re-averaging to a uniform reconstruction grid that includes $\check{x}_i$, we can achieve the needed high order reconstruction of $u(\check{x}_i)$.

The WENO reconstruction to an arbitrary point, in particular, could be made more computationally efficient. The trace point is only computed approximately, both because the trace velocity approximates the true particle or characteristic velocity and a numerical solver is needed to evolve the traceline. It seems reasonable to perturb the point slightly to the nearest point of a fine grid of equally spaced points for which the linear weights have been precomputed.

The scheme presented in this paper is based on finite volumes, and can be considered as an extension of the linear scheme of Huang, Arbogast, and Qiu presented in [19]. For the interested reader, a similar scheme based on finite differences is given in [15]. It should be considered as an extension of the semi-Lagrangian finite difference WENO scheme of Qiu and Shu [27, 28] developed for linear problems. As a finite difference scheme, it is based on a standard sliding average formulation with the flux function defined using WENO reconstructions of (semi-Lagrangian) characteristic tracings of grid points, and so it has a small CFL time step stability restriction for linear problems (and, as here, a relaxed CFL condition for nonlinear problems).

Like as was done in this work, to handle nonlinear problems, an approximate, locally frozen trace velocity and a flux correction step are used. A special two-stage WENO reconstruction procedure is also developed that is biased to the upstream direction.

In conclusion, because they can use very long time steps, Eulerian-Lagrangian schemes are highly competitive for linear scalar conservation laws. We believe that our extension to nonlinear scalar conservation laws is the natural approach, and provides good results with reduced numerical diffusion using long time steps. The extension to the Euler system is also natural, and it can also use longer time steps, but it may not be computationally competitive due to inherent difficulties involving multiple tracelines. Along the way, we also developed novel techniques for WENO reconstruction that should find application in other contexts.

[1] T. Arbogast, A. Chilakapati, and M. F. Wheeler, *A characteristic-mixed method for contaminant transport and miscible displacement*, in Computational Methods in Water Resources IX, vol. 1, T. F. Russell et al., eds., Computational Mechanics Public., Southampton, U.K., 1992, pp. 77–84.

[2] T. Arbogast and C. Huang, *A fully mass and volume conserving implementation of a characteristic method for transport problems*, SIAM J. Sci. Comput., 28 (2006), pp. 2001–2022.

[3] T. Arbogast, C.-S. Huang, and T. F. Russell, *A locally conservative Eulerian-Lagrangian method for a model two-phase flow problem in a one-dimensional porous medium*, SIAM J. Sci. Comput., 34 (2012), pp. A1950–A1974.

[4] T. Arbogast and W. Wang, *Convergence of a fully conservative volume corrected characteristic method for transport problems*, SIAM J. Numer. Anal., 48:3 (2010), pp. 797–823.

[5] T. Arbogast and M. F. Wheeler, *A characteristics-mixed finite element method for advection dominated transport problems*, SIAM J. Numer. Anal., 32 (1995), pp. 404–424.

[6] E. Carlini, R. Ferretti, and G. Russo, *A weighted essentially nonoscillatory, large time-step scheme for Hamilton-Jacobi equations*, SIAM J. Sci. Comput., 27 (2005), pp. 1071–1091.

[7] M. A. Celia, T. F. Russell, I. Herrera, and R. E. Ewing, *An Eulerian-Lagrangian localized adjoint method for the advection-diffusion equation*, Advances in Water Resources, 13 (1990), pp. 187–206.

[8] H. K. Dahle, R. E. Ewing, and T. F. Russell, *Eulerian-Lagrangian localized adjoint methods for a nonlinear advection-diffusion equation*, Comput. Methods Appl. Mech. Engrg., 122 (1995), pp. 223–250.

[9] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodríguez-Ferran, *Arbitrary Lagrangian-Eulerian methods*, in Encyclopedia of Computational Mechanics, E. Stein, R. de Borst, and T. J. R. Hughes, eds., vol. 1: Fundamentals, John Wiley & Sons, Ltd., 2004, ch. 14.

[10] J. Douglas, Jr. and C.-S. Huang, *The convergence of a locally conservative Eulerian-Lagrangian finite difference method for a semilinear parabolic equation*, BIT, 41 (2001), pp. 480–489.

[11] J. Douglas, Jr., F. Pereira, and L.-M. Yeh, *A locally conservative Eulerian-Lagrangian numerical method and its application to nonlinear transport in porous media*, Comput. Geosci., 4 (2000), pp. 1–40.

[12] J. Douglas, Jr. and T. F. Russell, *Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures*, SIAM J. Numer. Anal., 19 (1982), pp. 871–885.

[13] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, *Uniformly high-order accurate essentially nonoscillatory schemes III*, J. Comput. Phys., 71 (1987), pp. 231–303.

[14] A. Harten and S. Osher, *Uniformly high-order accurate nonoscillatory schemes I*, SIAM J. Numer. Anal., 24 (1987), pp. 279–309.

[15] C.-S. Huang, T. Arbogast, and Ch.-H. Hung, *A semi-Lagrangian finite difference WENO scheme for scalar nonlinear conservation laws*, J. Comput. Phys. (2016), to appear. DOI 10.1016/j.jcp.2016.06.027.

[16] J. Geiser, *Fourth-Order Splitting Methods for Time-Dependant Differential Equations*, Numer. Math. Theor. Meth. Appl., 1:3 (2008), pp. 321–339.

[17] J. Geiser, *Consistency of Iterative Operator-Splitting Methods: Theory and Applications*, Numer. Methods Partial Diff. Eqns., 26 (2010), pp. 135–158.

[18] C.-S. Huang, T. Arbogast, and C.-H. Hung, *A re-averaged WENO reconstruction and a third order CWENO scheme for hyperbolic conservation laws*, J. Comput. Phys., 262 (2014), pp. 291–312.

[19] C.-S. Huang, T. Arbogast, and J. Qiu, *An Eulerian-Lagrangian WENO finite volume scheme for advection problems*, J. Comput. Phys., 231 (2012), pp. 4028–4052. DOI 10.1016/j.jcp.2012.01.030.

[20] J.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228.

[21] R. J. LeVeque, *High-resolution conservative algorithms for advection in incompressible flow*, SIAM. J. Numer. Anal., 33 (1996), pp. 627–665.

[22] D. Levy, G. Puppo, and G. Russo, *Central WENO schemes for hyperbolic systems of conservation laws*, Math. Model. Numer. Anal., 33 (1999), pp. 547–571.
[23] X. D. Liu, S. Osher, and T. Chan, *Weighted essentially non-oscillatory schemes*, J. Comput. Phys., 115 (1994), pp. 200–212.
[24] O. Pironneau, *On the transport-diffusion algorithm and its applications to the Navier-Stokes equations*, Numer. Math., 38 (1981/82), pp. 309–332.
[25] J. Qiu and C.-W. Shu, *On the construction, comparison, and local characteristic decomposition for high-order central WENO schemes*, J. Comput. Phys., 183 (2002), pp. 187–209.
[26] J.-M. Qiu and A. Christlieb, *A conservative high order semi-Lagrangian WENO method for the Vlasov equation*, J. Comput. Phys., 229 (2010), pp. 1130–1149.
[27] J.-M. Qiu and C.-W. Shu, *Conservative high order semi-Lagrangian finite difference WENO methods for advection in incompressible flow*, J. Comput. Phys., 230 (2011), pp. 863–889.
[28] ———, *Conservative semi-Lagrangian finite difference WENO formulations with applications to the Vlasov equation*, Communications in Comput. Phys., 10 (2011), pp. 979–1000.
[29] C. Shen, J.-M. Qiu, and A. Christlieb, *Adaptive mesh refinement based on high order finite difference WENO scheme for multi-scale simulations*, J. Comput. Phys., 230 (2011), pp. 3780–3802.
[30] C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes II, *J. Comput. Phys.* **83** (1989) 32–78.
[31] J. M. Stockie, J. A. Mackenzie, and R. D. Russell, *A moving mesh method for one-dimensional hyperbolic conservation laws*, SIAM J. Sci. Comput., 22 (2001), pp. 1791–1813.
[32] G. Strang, *On the construction and comparison of difference schemes*, SIAM. J. Numer. Anal., 5 (1968), pp. 506–517.
[33] H. Wang and M. Al-Lawatia, *A locally conservative Eulerian-Lagrangian control-volume method for transient advection-diffusion equations*, Numer. Methods Partial Diff. Eqns., 22 (2006), pp. 577–599.
[34] H. Wang, D. Liang, R. E. Ewing, S. L. Lyons, and G. Qin, *An ELLAM approximation for highly compressible multicomponent flows in porous media.*, Comput. Geosci., 6 (2002), pp. 227–251.
[35] P. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, *J. Comput. Phys.*, 54 (1984), pp. 115–173.
[36] X. Yang, W. Huang, and J. Qiu, *A moving mesh WENO method for one-dimensional conservation laws*, SIAM J. Sci. Comput., 34 (2012), pp. A2317–A2343.
[37] M. Zennaro, *Natural continuous extensions of Runge-Kutta methods*, Math. Comp., 46 (1986), pp. 119–133.