Commun. Comput. Phys.
doi: 10.4208/cicp.2019.js60.15

Vol. **26**, No. 5, pp. 1530-1574
November 2019

High Order Arbitrary Lagrangian-Eulerian Finite Difference WENO Scheme for Hamilton-Jacobi Equations†

Yue Li[1], Juan Cheng[2,3,*], Yinhua Xia[4] and Chi-Wang Shu[5]

[1] *Graduate School, China Academy of Engineering Physics, Beijing 100088, P.R. China.*

[2] *Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, Beijing 100088, P.R. China.*

[3] *Center for Applied Physics and Technology, Peking University, Beijing 100871, P.R. China.*

[4] *School of Mathematics Sciences, University of Science and Technology of China, Hefei, Anhui 230026, P.R. China.*

[5] *Division of Applied Mathematics, Brown University, Providence, RI 02912, USA.*

Received 8 April 2019; Accepted (in revised version) 1 May 2019

Abstract. In this paper, a high order arbitrary Lagrangian-Eulerian (ALE) finite difference weighted essentially non-oscillatory (WENO) method for Hamilton-Jacobi equations is developed. This method is based on moving quadrilateral meshes, which are often used in Lagrangian type methods. The algorithm is formed in two parts: spatial discretization and temporal discretization. In the spatial discretization, we choose a new type of multi-resolution WENO schemes on a nonuniform moving mesh. In the temporal discretization, we use a strong stability preserving (SSP) Runge-Kutta method on a moving mesh for which each grid point moves independently, with guaranteed high order accuracy under very mild smoothness requirement (Lipschitz continuity) for the mesh movements. Extensive numerical tests in one and two dimensions are given to demonstrate the flexibility and efficiency of our moving mesh scheme in solving both smooth problems and problems with corner singularities.

AMS subject classifications: 65M06, 35F21

Key words: ALE method, finite difference method, WENO method, Hamilton-Jacobi equation.

†This paper is dedicated to the honor of Professor Jie Shen on the occasion of his 60th Birthday.

*Corresponding author. *Email addresses:* liyue9443@outlook.com (Y. Li), cheng_juan@iapcm.ac.cn (J. Cheng), yhxia@ustc.edu.cn (Y. Xia), chi-wang_shu@brown.edu (C.-W. Shu)

1 Introduction

In this paper, we deal with the development of high order accurate weighted essentially non-oscillatory (WENO) finite difference schemes for solving one and two dimensional time-dependent Hamilton-Jacobi (HJ) equations on a moving mesh,

$$\phi_t + H(\nabla\phi, \mathbf{x}, t) = 0, \quad \text{in } \Omega\times(0,T), \quad \mathbf{x}\in\Omega, \tag{1.1}$$

where H is a nonlinear function which is at least Lipschitz continuous with respect to $\nabla\phi$. H may also depend on ϕ in applications, however the main difficulty for numerical solutions is the possibly nonlinear dependency of H on $\nabla\phi$.

The viscosity solution of (1.1) is introduced by Crandall and Lions [7], in which it is proved that there exists a unique bounded and Lipschitz continuous viscosity solution for the problem (1.1). We should note that the derivatives of the viscosity solution can be discontinuous (i.e. the development of corner-type singularities) even when the initial condition is smooth.

Hamilton-Jacobi equations are widely used in many areas, including computer vision and image processing, and front propagation problems in models for flame propagation and the growth of crystal [20, 24]. In the numerical simulations of multidimensional fluid flow, there are two typical choices: the Lagrangian framework, in which the mesh moves with the local fluid velocity, and the Eulerian framework, in which the fluid flows through a grid fixed in space. Hirt et al. proposed an arbitrary Lagrangian Eulerian (ALE) method [11]. Its basic idea is that the computational grid is no longer fixed or attached to fluid particles, but can move arbitrarily with respect to the coordinate system. In particular, it is often desirable that the mesh moves with the fluid at the interface (or at least in the normal direction of the interface or free surface), which ensures high resolution at the interface, while the movement of the mesh elsewhere could have more freedom. An arbitrary Lagrangian-Eulerian approach is very advantageous to solve multimaterial and moving boundary problems with incompressible flows [5] and compressible flows [19].

The monotone first-order accurate numerical method for solving Hamilton-Jacobi equations was first presented by Crandall and Lions [8]. Later, Osher and Sethian used the connection between conservation laws and Hamilton-Jacobi equations to construct high-order accurate "artifactfree" numerical methods [21]. After Osher and Shu [22] proposed a general framework for the numerical solution of Hamilton-Jacobi equations using successful methods from hyperbolic conservation laws, many high order numerical methods such as the essentially non-oscillatory (ENO) method [22], the weighted essentially non-oscillatory (WENO) method [13] and the discontinuous Galerkin (DG) method [12, 17] have been proposed along this route.

ENO and WENO schemes are high order accurate finite difference or finite volume schemes designed for problems with piecewise smooth solutions containing discontinuities. The key point of these methods is the adoption of an adaptive stencil for high order interpolation or reconstruction so as to avoid shocks, high or discontinuous gradient

regions whenever possible. WENO schemes, when applied to the Hamilton-Jacobi equations, can produce high order accuracy in the smooth regions of the solution and sharp, non-oscillatory solutions near the corner singularities. The classical WENO scheme for the HJ equations [13] was developed using a convex combination of all candidate stencils instead of just one as in the ENO scheme [22]. Lately, more variants of WENO methods have been proposed to solve the HJ equations. Hermite WENO (HWENO) schemes and related methods have been developed in [23,29,30] to achieve more compact stencils for the same order of accuracy. Bryson and Levy [4] presented central schemes for solving the HJ equations on structured meshes. Zhang and Shu [28] and Levy et al. [16] developed high order WENO and central WENO schemes on triangular meshes respectively. More recently, Zhu and Shu [31] proposed a new type of multi-resolution WENO schemes with increasingly higher order of accuracy for solving conservation laws, by using the information defined on a hierarchy of nested central spatial stencils. The linear weights of this type of WENO schemes can be any positive numbers on the condition that they sum to one. This new WENO scheme is simple to construct and can be easily implemented to arbitrary high order of accuracy and in high dimensions on non-cartesian meshes. In this paper, we adopt this method to construct our WENO interpolation.

There have already been many WENO and DG schemes designed for solving hyperbolic conservation laws on moving meshes, e.g. those in [3,9,27]. The aim of this paper is to combine the ALE and WENO approaches to develop a high order ALE-WENO finite difference scheme solving the Hamilton-Jacobi equations on the moving quadralateral meshes. At present, the majority of the research on the numerical solution of HJ equations has been based on fixed meshes, while the research on moving meshes is relatively sparse. In particular, it is a challenge for the moving mesh methods to achieve high order accuracy and stability at the same time. There are already some ALE-DG methods with different strategies in the literature. Most of these methods are based on the strategy of evolution on the old mesh to the next time step, followed by a remapping procedure to get the solution on the new mesh. The success of such schemes depends both on the evolution scheme and on the remapping procedure. There are also methods which combine the evolution and the mesh movement into a single step, however most such schemes would require strong smoothness of the mesh movement (the mesh movement function should have high order derivatives) in order to retain high order accuracy. Tang et al. proposed an adaptive mesh method to solve the HJ equations by transforming a uniform mesh in the logical domain to cluster grid points at the regions of the physical domain where the solution or its derivative is singular or nearly singular in [26]. Mackenzie and Nicola developed the numerical solution of first order Hamilton-Jacobi equations using the combination of a discontinuous Galerkin finite element method and an adaptive *r*-refinement (mesh movement) strategy in [18]. Klingenberg, Schnucke and Xia have recently proposed an interesting class of ALE-DG methods with moving meshes, which belongs to the class of methods combining the evolution and the mesh movement in a single step, uses the DG method for its spatial discretization and standard strong-stability-preserving Runge-Kutta time discretization, for solving Hamilton-Jacobi equations [14].

The novelty of this method is that high order accuracy and stability can be proved under very mild conditions on mesh movements, in particular, no smoothness beyond Lipschitz continuity for the mesh movement function (which is assumed to be piecewise linear) is required. Inspired by the method in [14], we will propose a class of high order finite difference ALE-WENO schemes for Hamilton-Jacobi equations on moving meshes in this paper. We will use the third order accurate scheme as an example to this class of high order ALE-WENO schemes. In order to be compatible with standard Lagrangian methods for solving fluid equations, we would like to use quadrilateral meshes which move with arbitrary mesh velocities. The spatial approximation of our scheme is based on the new multi-resolution WENO interpolation procedure of Zhu and Shu [31]. The temporal discretization is based on the class of strong stability preserving (SSP), also referred to as total variation diminishing (TVD), high order Runge-Kutta method [10, 25]. Our scheme is stable and high order accurate both in one- and in two-dimensional spaces, under arbitrary mesh velocities, subject only to the very mild Lipschitz continuity restriction. In this paper, we also explore the advantage of this ALE method by following (nearly) Lagrangian (characteristic) mesh movement to achieve smaller errors with the same number of mesh points in comparison with fixed meshes.

This paper is organized as follows. In Section 2, the one dimensional and two dimensional finite difference WENO schemes solving Hamilton-Jacobi equations in the ALE framework are presented, and the high order SSP method in temporal discretization is described. In Section 3, we present the numerical results solving several typical Hamilton-Jacobi equations on three types of moving meshes, namely smooth, random, and nearly Lagrangian (following characteristics) meshes. Finally, concluding remarks are given in Section 4.

2 The finite difference WENO scheme for Hamilton-Jacobi equations in the arbitrary Lagrangian-Eulerian framework

In order to describe the finite difference method for solving the Hamilton-Jacobi equation in the ALE framework, we need to take the motion of the grid into account. We first introduce a variable ω_i to describe the moving speed of the node x_i (in one-dimension) and a variable $\omega_{i,j}=(\omega_{x_{i,j}},\omega_{y_{i,j}})$ to describe the moving speed of the node $(x_{i,j},y_{i,j})$ (in two-dimensions) from the time level n (denoted as t_n) to $n+1$ (denoted as t_{n+1}), which lead to the following discrete scheme. In the design of the scheme we need to take into account the speed of the grid movement, therefore there will be some changes in the formula that determines the numerical Hamiltonian.

We assume that the family of the mesh $\{T_n, n=0,\cdots,L\}$ at all the time levels gives the same mesh topology, i.e., the mesh T_{n+1} has the same number of nodes and the same connectivity as T_n. Under such restriction, we can set up a moving mesh connecting the node at the time level n and the corresponding node at the time level $n+1$ linearly, and a global (in space and time) mesh movement function which is linear in space for any time

t. This linear mapping between T_n and T_{n+1} is the crucial ingredient for the ALE-DG method in [14] to maintain stability and accuracy while allowing only very mild regularity of the mesh movement function (Lipschitz continuity is enough). In one dimension, we first give the definition of the mesh velocity ω_i^n and the node $x_i(t)$,

$$\omega_i^n := \frac{x_i^{n+1}-x_i^n}{\Delta t}, \qquad x_i(t) := x_i^n + \omega_i^n (t-t_n), \quad t \in [t_n, t_{n+1}], \tag{2.1}$$

and then we can define the global mesh velocity $\omega(x,t)$ in the interval $K_i(t)=[x_{i-1}(t),x_i(t)]$ for $t\in[t_n,t_{n+1}]$ by the linear interpolation

$$\omega(x,t) = \left(\omega_i^n - \omega_{i-1}^n\right)\left(\frac{x-x_{i-1}(t)}{h_{i-1/2}(t)}\right) + \omega_{i-1}^n, \qquad t\in[t_n,t_{n+1}], \quad x\in K_i(t), \tag{2.2}$$

where

$$h_{i-1/2}(t) = x_i(t) - x_{i-1}(t), \quad \text{for } t\in[t_n,t_{n+1}]. \tag{2.3}$$

Similarly, in two dimensions, we have the definition as

$$\begin{aligned}
&\omega_{x_{i,j}}^n := \frac{x_{i,j}^{n+1}-x_{i,j}^n}{\Delta t}, \qquad x_{i,j}(t) := x_{i,j}^n + \omega_{x_{i,j}}^n (t-t_n), \quad t\in[t_n,t_{n+1}],\\
&\omega_{y_{i,j}}^n := \frac{y_{i,j}^{n+1}-y_{i,j}^n}{\Delta t}, \qquad y_{i,j}(t) := y_{i,j}^n + \omega_{y_{i,j}}^n (t-t_n), \quad t\in[t_n,t_{n+1}].
\end{aligned} \tag{2.4}$$

A similar time-dependent affine linear mapping between two neighboring time levels can also be defined in all the cells in two dimensions. With the well-defined velocity of the grid, we can begin to design our numerical scheme.

For the one dimensional HJ equation

$$\phi_t + H(\phi_x, x, t) = 0, \tag{2.5}$$

its semi-discrete scheme is given by

$$\frac{\partial}{\partial t}\phi_i(t) + \hat{H}(\phi_{x_i}^-, \phi_{x_i}^+; x, t) = 0, \tag{2.6}$$

where $\phi_{x_i}^-$ and $\phi_{x_i}^+$ are the left-biased and right-biased approximations to the derivative of ϕ at the node i, which are obtained from the interpolation polynomials with stencils biased to the left and to the right respectively. A monotone numerical Hamiltonian $\hat{H}$ is one which is monotonically non-decreasing in the first argument and monotonically non-increasing in the second argument, which can be symbolically represented as

$$\hat{H}(\uparrow,\downarrow;x,t).$$

We will apply the simple Lax-Friedrichs numerical Hamiltonian as our first order monotone numerical Hamiltonian, suitably taking the mesh movement into account:

$$\hat{H}(u_i^-, u_i^+; x_i, t) = H\left(\frac{u_i^- + u_i^+}{2}, x_i, t\right) - \frac{1}{2}\omega_i(u_i^- + u_i^+) - \frac{1}{2}\alpha_i(u_i^+ - u_i^-), \tag{2.7}$$

where

$$\alpha_i = \max_{a\le u\le b}\{|H_u(u,x,t)-\omega_i|\}, \quad a=\min\{u_i^-,u_i^+\}, \quad b=\max\{u_i^-,u_i^+\},$$
$$\alpha = \max_i\{\alpha_i\}. \tag{2.8}$$

Here H_u denotes the partial derivative of $H(u,x,t)$ with respect to u, and ω_i is defined by (2.1).

If we simply use the first order approximation to obtain u^- and u^+

$$u_i^- = \frac{\phi_i-\phi_{i-1}}{h_{i-1/2}}, \qquad u_i^+ = \frac{\phi_{i+1}-\phi_i}{h_{i+1/2}}$$

in the scheme (2.6), we will get a monotone scheme which will be stable and convergent, but it will only be first order accurate [8]. Our objective is to use the monotone Hamiltonian as a building block to obtain a higher order scheme.

A crucial ingredient to obtain higher order accuracy is to obtain higher order accurate approximations u_i^- and u_i^+ to $(\phi_x)_i$. This will be described in Section 2.1. Another crucial ingredient to obtain higher order accuracy is to use a suitable high order time discretization. This will be described in Section 2.3.

For the two dimensional Hamilton-Jacobi equation

$$\phi_t + H(\phi_x,\phi_y,x,y,t) = 0, \tag{2.9}$$

its semi-discrete scheme is given by

$$\frac{\partial}{\partial t}\phi_{i,j}(t) + \hat{H}((\nabla\phi_{i,j})_1,(\nabla\phi_{i,j})_2,(\nabla\phi_{i,j})_3,(\nabla\phi_{i,j})_4;x_{i,j},y_{i,j},t) = 0, \tag{2.10}$$

where $(\nabla\phi_{i,j})_\ell = (\phi_{x_{i,j}},\phi_{y_{i,j}})_\ell$, $\ell=1,2,3,4$, are the approximations to the derivatives of ϕ at the node $(x_{i,j},y_{i,j})$ obtained from the interpolation polynomials with stencils biased to the four quadrants shown in Fig. 1. Just as in the one dimensional case, $\hat{H}$ needs to be a monotone Hamiltonian also. The Lax-Friedrichs type monotone Hamiltonian on unstructured grids developed by Abgrall in [1] is a generalization of the Lax-Friedrichs type monotone Hamiltonian on the Cartesian mesh. For our moving mesh case, the Lax-Friedrichs monotone Hamiltonian should be defined as

$$\begin{aligned}&\hat{H}((\nabla\phi_{i,j})_1,(\nabla\phi_{i,j})_2,(\nabla\phi_{i,j})_3,(\nabla\phi_{i,j})_4,x_{i,j},y_{i,j},t)\\ =&H(u_{i,j},v_{i,j};x_{i,j},y_{i,j},t) - \omega_{x_{i,j}}u_{i,j} - \omega_{y_{i,j}}v_{i,j} - D(\phi_{x_{i,j}},\phi_{y_{i,j}}),\end{aligned} \tag{2.11}$$

where

$$u_{i,j} = \frac{\sum_{\ell=1}^4\theta_\ell(\phi_{x_{i,j}})_\ell}{2\pi}, \qquad v_{i,j} = \frac{\sum_{\ell=1}^4\theta_\ell(\phi_{y_{i,j}})_\ell}{2\pi},$$
$$D(\phi_{x_{i,j}},\phi_{y_{i,j}}) = \frac{\alpha_{i,j}}{4}\sum_{\ell=1}^4\beta_{\ell+\frac{1}{2}}\left(\frac{(\nabla\phi_{i,j})_\ell+(\nabla\phi_{i,j})_{\ell+1}}{2}\right)\cdot\vec{n}_{\ell+\frac{1}{2}}, \tag{2.12}$$

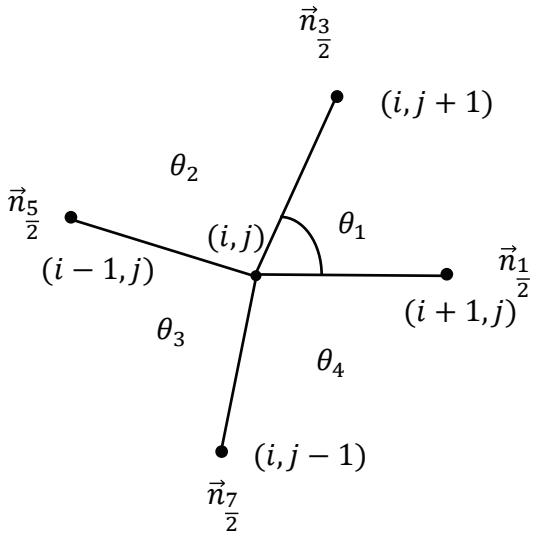


Figure 1: The node $(x_{i,j},y_{i,j})$ and its angular sectors.

$$\beta_{\ell+\frac{1}{2}}=\tan\left(\frac{\theta_\ell}{2}\right)+\tan\left(\frac{\theta_{\ell+1}}{2}\right), \tag{2.13}$$

$$\alpha_{i,j}=\max_{a\le u\le b,c\le v\le d}\{|H_1(u,v,x,y,t)-\omega_{x_{i,j}}|,|H_2(u,v,x,y,t)-\omega_{y_{i,j}}|\},$$

$$\alpha=\max_{i,j}\{\alpha_{i,j}\}. \tag{2.14}$$

Here, θ_ℓ, for $\ell=1,\cdots,4$, are the angles between two neighboring edges connecting the node (i,j), and $\vec{n}_{\ell+\frac{1}{2}}$, for $\ell=1,\cdots,4$, are the unit vectors along the edges passing through the node (i,j), see Fig. 1. $a=\min\{(\phi_{x_{i,j}})_1,(\phi_{x_{i,j}})_2,(\phi_{x_{i,j}})_3,(\phi_{x_{i,j}})_4\}$, $b=\max\{(\phi_{x_{i,j}})_1,(\phi_{x_{i,j}})_2,\ (\phi_{x_{i,j}})_3,(\phi_{x_{i,j}})_4\}$; $c=\min\{(\phi_{y_{i,j}})_1,(\phi_{y_{i,j}})_2,(\phi_{y_{i,j}})_3,(\phi_{y_{i,j}})_4\}$, $d=\max\{(\phi_{y_{i,j}})_1,(\phi_{y_{i,j}})_2,\ (\phi_{y_{i,j}})_3,(\phi_{y_{i,j}})_4\}$. Finally, H_1 and H_2 refer to the partial derivatives of $H(u,v,x,y,t)$ with respect to u and v respectively.

If we simply use the gradient of the linear function interpolating ϕ in the ℓ-th quadrant shown in Fig. 1 as $(\nabla\phi_{i,j})_\ell$, and use α instead of $\alpha_{i,j}$ defined in (2.14), in the scheme (2.10), it can be shown that the scheme (2.10) is a monotone scheme, and is stable and convergent [1]. However, it will only be first order accurate. Our objective is again to use this monotone Hamiltonian as a building block to obtain a higher order scheme.

We compute the mesh movement velocity $\omega_{i,j}=(\omega_{x_{i,j}},\omega_{y_{i,j}})$ by (2.4) at every time level. Also, we use a two dimensional WENO interpolation procedure, described in detail in Section 2.2, to determine $(\nabla\phi_{i,j})_\ell$ for $\ell=1,\cdots,4$ in the spatial discretization. After that, we will use the SSP time discretization described in Section 2.3 to obtain a high order scheme both in space and in time.

2.1 One dimensional WENO procedure in the spatial discretization

We will use the multi-resolution type WENO procedure designed recently by Zhu and Shu in [31] for our spatial discretization. This procedure is particularly simple for irregular and moving meshes. In this paper, we will design third order schemes as examples, however the designing procedure can be applied to arbitrarily high order accuracy. In order to get third order approximation to $\phi^-_{x_i}$ and $\phi^+_{x_i}$, we would like to interpolate the function ϕ by a third degree polynomial $p(x)$ with a stencil biased to the left and to the right, respectively. We will describe the construction of the polynomial $p(x)$ associated with the cell $I_c=[x_i,x_{i+1}]$ as shown in Fig. 2, which is used to approximate $\phi^+_{x_i}$ as well as $\phi^-_{x_{i+1}}$. The WENO procedure to obtain $p(x)$ consists of the following steps.

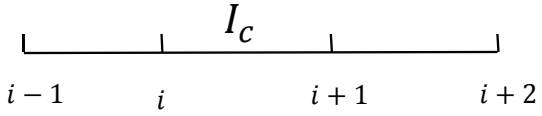


Figure 2: The cell I_c and its neighbors in the one dimensional space.

Step 1. We choose two hierarchical central spatial stencils $T_1=\{i,i+1\}$ and $T_2=\{i-1,i,i+1,i+2\}$, the nodes of which are shown in Fig. 2. Then it is easy to interpolate the point values of ϕ to obtain a first degree polynomial $p_1(x)$ on T_1 and a third degree polynomial $q_2(x)$ on T_2.

Step 2. We define a third degree polynomial $p_2(x)$ by

$$p_2(x)=\frac{1}{\gamma_2}q_2(x)-\frac{\gamma_1}{\gamma_2}p_1(x),\quad \text{where}\quad \gamma_1+\gamma_2=1.$$

Here γ_1 and γ_2 are the linear weights, which can be chosen as arbitrary positive numbers on the condition that they sum to one. Accuracy for the eventual WENO polynomial will be better in smooth regions if γ_2 is chosen larger, while the ability to control spurious oscillations near singularities will be better if γ_1 is chosen larger. As a good balance, we follow [31] and choose $\gamma_1=\frac{1}{11}$, $\gamma_2=\frac{10}{11}$.

Step 3. Compute the smoothness indicators β_1 and β_2, which are used to measure how smooth the functions $p_1(x)$ and $p_2(x)$ are in the cell I_c respectively. We use the similar definition for the smoothness indicators as that in [31], except that we are now solving Hamilton-Jacobi equations which is an integrated version of conservation laws, hence we start the measurement of smoothness from the second derivative, instead of the first derivative as in [31]. The smoothness indicator for $p_2(x)$ is given by

$$\beta_2=\int_{x_i}^{x_{i+1}}(x_{i+1}-x_i)\left(\frac{\partial^2 p_2(x)}{\partial x^2}\right)^2dx+\int_{x_i}^{x_{i+1}}(x_{i+1}-x_i)^3\left(\frac{\partial^3 p_2(x)}{\partial x^3}\right)^2dx. \tag{2.15}$$

Since $p_1(x)$ is a first degree polynomial, if we use the formula (2.15) (replacing $p_2(x)$ by $p_1(x)$) to determine β_1, we would get $\beta_1=0$. This does not cause any problems in the accuracy test, the designed high order accuracy can be achieved. However, it does lead to more smearing to problems with corner singularities. This is not surprising, as a smaller smoothness indicator would lead to a larger nonlinear weight, hence a zero smoothness indicator would lead to a (relatively) very large nonlinear weight near any corner discontinuities, essentially converting the WENO approximation to that of the low order approximation from $p_1(x)$ alone. We would therefore like to enlarge the smoothness indicator of $p_1(x)$ to represent the smoothness of the function in a slightly larger neighborhood than the stencil T_1 of $p_1(x)$, following the practice in [31]. Two candidates for this "slightly larger" neighborhood correspond to adding the left neighboring point, resulting in $S_1=\{i-1,i,i+1\}$, and adding the right neighboring point, resulting in $S_2=\{i,i+1,i+2\}$, respectively. We will then have two quadratic interpolation polynomials, and will use one of their smoothness indicators following the formula (2.15) to calculate β_1. In order to enlarge the smoothness indicator of $p_1(x)$ as little as possible, we would like to use the smaller of these two candidates, namely

$$\beta_1=\min\{\beta(1),\beta(2)\}, \tag{2.16}$$

where

$$\begin{aligned}\beta(1)=&\left[\frac{2\phi_{i-1}}{(x_{i-1}-x_i)(x_{i-1}-x_{i+1})}+\frac{2\phi_i}{(x_i-x_{i-1})(x_i-x_{i+1})}\right.\\&\left.+\frac{2\phi_{i+1}}{(x_{i+1}-x_{i-1})(x_{i+1}-x_i)}\right]^2(x_{i+1}-x_i)^2,\\\beta(2)=&\left[\frac{2\phi_i}{(x_i-x_{i+1})(x_i-x_{i+2})}+\frac{2\phi_{i+1}}{(x_{i+1}-x_i)(x_{i+1}-x_{i+2})}\right.\\&\left.+\frac{2\phi_{i+2}}{(x_{i+2}-x_i)(x_{i+2}-x_{i+1})}\right]^2(x_{i+1}-x_i)^2.\end{aligned}$$

Step 4. Compute the nonlinear weights based on the linear weights and the smoothness indicators, which follows the WENO-Z strategy in [2]. The nonlinear weights are given as,

$$w_k=\frac{\tilde{w}_k}{\sum_{\ell=1}^{2}\tilde{w}_\ell},\quad k=1,2, \tag{2.17}$$

where

$$\tilde{w}_\ell=\gamma_\ell\left(1+\left(\frac{\tau}{\epsilon+\beta_\ell}\right)^2\right),\quad \ell=1,2,$$

with

$$\tau=|\beta_2-\beta_1|, \tag{2.18}$$

and ϵ is taken as 10^{-4} in all the simulations.

Step 5. The new final interpolation polynomial $p(x)$ is given by

$$p(x)=w_1p_1(x)+w_2p_2(x), \tag{2.19}$$

which gives a fourth order approximation to $\phi(x)$ in smooth regions and is essentially non-oscillatory for its derivative near derivative singularities. We will use

$$\phi^+_{x_i}=\frac{\partial p(x)}{\partial x}|_{x=x_i}, \quad \phi^-_{x_{i+1}}=\frac{\partial p(x)}{\partial x}|_{x=x_{i+1}},$$

which are third order approximations in smooth regions.

2.2 Two dimensional WENO procedure in the spatial discretization

In order to get a third order approximation to $\nabla\phi$, we would like to use a third degree interpolation polynomial $p(x,y)$. In this subsection, we follow the similar idea as in the one dimensional case to make third order WENO approximation to the nodes 1, 2, 3, 4 of the target cell I_c respectively, as shown in Fig. 3. In the following, we take the node 1 as an example.

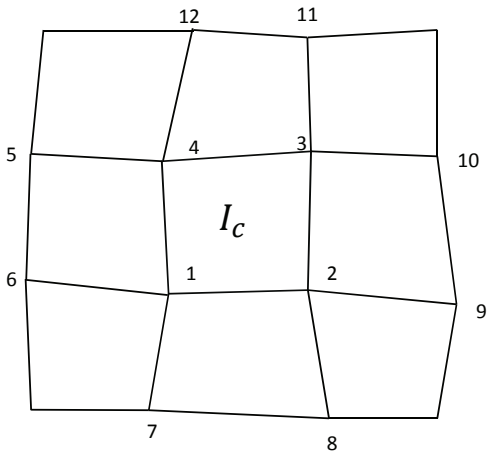


Figure 3: The cell I_c and its neighbors in the two dimensional space.

Step 1. Let (x_0,y_0) be the barycenter of the target cell I_c. We define $\xi=\frac{(x-x_0)}{\sqrt{|I_c|}}$, $\eta=\frac{(y-y_0)}{\sqrt{|I_c|}}$, where $|I_c|$ is the area of the quadrangle I_c. Then, we can write the third degree polynomial $q_2(x,y)$ as

$$q_2(x,y)=\sum_{j=0}^{3}\sum_{s+r=j}a_{rj}\xi^s\eta^r.$$

It has 10 degrees of freedom, so we would need to use information from at least 10 nodes from the cell I_c and its neighbors.

For the small stencil, we choose it as $T_1=\{1,2,4\}$, and then we can get a first degree polynomial $p_1(x,y)$ by interpolation. To make the stencil symmetrical, we choose the big stencil as $T_2=\{1,2,\cdots,12\}$ (see Fig. 3) to obtain a third degree polynomial $q_2(x,y)$. As a cubic polynomial has ten degrees of freedom, and we have the information from 12 nodes, we will adopt a least square procedure. Experience (see [29]) indicates that it is crucial to require exact collocation for the four vertices of the target cell I_c, otherwise the approximation may become unstable. The approximation $q_2(x,y)$ is thus obtained as

$$\begin{aligned} &\min_{q\in Q}\sum_{\ell=1}^{12}(q(x_\ell,y_\ell)-\phi_\ell)^2 \\ &\text{subject to} \quad q(x_\ell,y_\ell)=\phi_\ell, \quad \ell=1,2,3,4, \end{aligned}$$

where $Q=\{q(x,y)\in P^3\}$ (polynomials of degree at most 3).

Step 2. For the fourth-order approximation, similarly to the one dimensional case, we define the polynomial $p_2(x,y)$ by

$$p_2(x,y)=\frac{1}{\gamma_2}q_2(x,y)-\frac{\gamma_1}{\gamma_2}p_1(x,y), \tag{2.20}$$

where $\gamma_1+\gamma_2=1$. Here we again take the linear weights as $\gamma_1=\frac{1}{11}$, $\gamma_2=\frac{10}{11}$.

Step 3. Compute the smoothness indicators to measure how smooth the function $p_1(x,y)$ and $p_2(x,y)$ are in the cell I_c. For $p_2(x,y)$, the smooth indicator β_2 is determined by

$$\beta_2=\sum_{\ell_1+\ell_2\geq 2}\int_{I_c}|I_c|^{\ell_1+\ell_2-2}\left(\frac{\partial^{\ell_1+\ell_2}}{\partial x^{\ell_1}\partial y^{\ell_2}}p_2(x,y)\right)^2 dxdy. \tag{2.21}$$

We refer to [6] for the details of the implementation of β_2.

As before, we would need to upgrade β_1 for $p_1(x,y)$, which would be zero if we adopt the formula (2.21). For the same reason as in the one dimensional situation, we need to use the neighboring nodes around T_1 to obtain a quadratic interpolation for the calculation of β_1. Specifically, we select the small stencils as $S_1=\{1,2,3,4,5,6\}$, $S_2=\{1,2,3,4,7,8\}$, $S_3=\{1,2,3,4,9,10\}$ and $S_4=\{1,2,3,4,11,12\}$ which are shown in Fig. 3. We would like to construct four quadratic interpolation polynomials $p^k(x,y)$

$$p^k(x,y)=\sum_{j=0}^{2}\sum_{s+r=j}a_{rj}^k\xi^s\eta^r, \qquad k=1,2,3,4$$

such that

$$p^k(x_\ell,y_\ell)=\phi_\ell, \qquad \ell\in S_k.$$

Then we have

$$\beta(k)=\sum_{\ell_1+\ell_2=2}\int_{I_c}|I_c|^{\ell_1+\ell_2-2}\left(\frac{\partial^{\ell_1+\ell_2}}{\partial x^{\ell_1}\partial y^{\ell_2}}p^k(x,y)\right)^2dxdy.$$

Finally we choose

$$\beta_1=\min_{k=1,2,3,4}\{\beta(k)\}.$$

Step 4. Compute the nonlinear weights in the same way as in the one dimensional situation. This step is identical to Step 4 of the one dimensional algorithm described in Section 2.2 and is therefore not repeated here.

Step 5. The final interpolation polynomial $p(x,y)$ is given by

$$p(x,y)=w_1p_1(x,y)+w_2p_2(x,y), \tag{2.22}$$

which approximates $\phi(x,y)$ to fourth order accuracy, and its derivatives approximate that of $\phi(x,y)$ to third order accuracy. Therefore, we can get a third order approximation to $(\phi_{x_{i,j}})_1$ at the node 1 by taking the derivative of $p(x,y)$ in the x direction, and to $(\phi_{y_{i,j}})_1$ at the node 1 by taking the derivative of $p(x,y)$ in the y direction. Finally $(\nabla\phi_{i,j})_1=((\phi_{x_{i,j}})_1,(\phi_{y_{i,j}})_1)$.

By using the above steps, we can obtain a similar interpolation polynomial $p(x,y)$ to approximate each of $(\nabla\phi_{i+1,j})_2$, $(\nabla\phi_{i+1,j+1})_3$ and $(\nabla\phi_{i,j+1})_4$ at the nodes 2,3,4 respectively. Then we can apply them in the numerical Hamiltonian (2.11) to obtain third order spatial accuracy in the smooth regions and non-oscillatory performance near corner singularities.

2.3 High order SSP Runge-Kutta time discretization

For all of the spatial discretizations discussed in the previous section, the time derivative term is left undiscretized. A popular high order time discretization method is the class of strong stability preserving (SSP), also referred to as the total variation diminishing (TVD), high order Runge-Kutta time discretizations [10, 25].

In this paper, we use the third order SSP Runge-Kutta method as follows,

$$\begin{aligned}
&\phi^{(1)}=\phi^n+\Delta tL(\phi^n,t^n),\\
&\phi^{(2)}=\frac{3}{4}\phi^n+\frac{1}{4}(\phi^{(1)}+\Delta tL(\phi^{(1)},t^n+\Delta t)),\\
&\phi^{n+1}=\frac{1}{3}\phi^n+\frac{2}{3}(\phi^{(2)}+\Delta tL(\phi^{(2)},t^n+\frac{1}{2}\Delta t)),
\end{aligned} \tag{2.23}$$

where the operator L represents $-\hat{H}(\phi^-_{x_i},\phi^+_{x_i};x_i,t)$ given by (2.7) in the one dimensional scheme, and represents $-\hat{H}((\nabla\phi_{i,j})_1,(\nabla\phi_{i,j})_2,(\nabla\phi_{i,j})_3,(\nabla\phi_{i,j})_4;x_{i,j},y_{i,j},t)$ given by (2.11)

in the two dimensional scheme. Thus, up to now we have a fully discrete numerical ALE-WENO scheme (2.23).

We calculate the time step Δt by the CFL condition,

$$\Delta t = \frac{c_1 h}{\alpha}, \tag{2.24}$$

where c_1 is the CFL number, here we choose it as 0.6. The coefficient α is computed by (2.8) or (2.14) in the one dimensional or two dimensional case respectively. h is the grid size. In the one dimensional case, h is the minimum size of all the cells in the computational domain, and in the two dimensional case, h is the minimum diameter of the inscribed circles for all the quadrilateral cells in the domain. In some of the situations, such as the mesh moving along characteristics, α in (2.24) could be very small, leading to a very large Δt determined by the stability constraint (2.24). In such cases, it is prudent to reduce Δt in order to ensure temporal accuracy. In our numerical tests of characteristic-line type moving mesh, when Δt determined by (2.24) is larger than $5h$, we will set it to be $5h$, unless otherwise stated. In the temporal discretization, in order to ensure the accuracy of the algorithm on a moving mesh, as suggested in [14], the grid movement speed should satisfy the following boundedness and Lipschitz continuity properties,

$$|\omega_i| \le c_2, \qquad |\omega_i'| = \left| \frac{\omega_{i+1} - \omega_i}{x_{i+1} - x_i} \right| \le c_3. \tag{2.25}$$

In the following one- and two- dimensional numerical examples, we use $c_2 = 10$ and $c_3 = 10$ to enforce these conditions.

Generally we can easily choose the grid motion which can guarantee the condition (2.25). However, for the random grids and the grids which move along the characteristic lines that we will discuss later, since their grid moving speeds are not given a priori, they might not automatically satisfy the condition (2.25). In such cases, we would need to make some modifications on either x_i^{n+1} or ω_i to make the condition (2.25) satisfied. To be more specific, in the one dimensional random grid case, the grid point locations x_i^n and x_i^{n+1} at the n-th and $(n+1)$-th time levels are given by an independent set of random perturbations from the uniform mesh. In order to satisfy (2.25), there should be a *lower* bound for the time step, that is, we would need $\Delta t \ge \Delta t_0$ where Δt_0 is given by

$$\Delta t_0 = \max \left\{ \frac{|x_i^{n+1} - x_i^n|}{c_2}, \frac{|(x_i^{n+1} - x_i^n) - (x_{i-1}^{n+1} - x_{i-1}^n)|}{c_3 (x_i^n - x_{i-1}^n)} \right\}. \tag{2.26}$$

We can estimate the mesh velocity ω_i by using the allowable lower bound of the time step Δt_0.

$$\omega_i = \frac{x_i^{n+1} - x_i^n}{\Delta t_0}. \tag{2.27}$$

Then we can compute a conservative estimate of the coefficient α^* by using $\phi_{x_i}^{\pm}$ and this mesh velocity ω_i,

$$\alpha^* = \max_i \left\{ \left| H' \left(\frac{\phi_{x_i}^- + \phi_{x_i}^+}{2} \right) \right| + |\omega_i| \right\}. \tag{2.28}$$

We use this coefficient α^* to compute a more demanding time step Δt to ensure stability,

$$\Delta t = \frac{c_1 h}{\alpha^*}. \tag{2.29}$$

When this Δt is used, the scheme is stable. However, to ensure that the condition (2.25) is satisfied, we should have $\Delta t \geq \Delta t_0$. When this fails, we would need to modify the random grid points at the time level $n+1$ to reduce their difference from those at the time level n. To be more specific, we consider the following two cases.

Case 1. $\Delta t \geq \Delta t_0$**.** This Δt satisfies the condition (2.25), hence we do not need to modify the random grid points at the time level $n+1$. We compute the actual mesh velocity ω_i as

$$\omega_i = \frac{x_i^{n+1} - x_i^n}{\Delta t}, \tag{2.30}$$

then we use this ω_i to calculate the coefficient α_i by using the formula (2.8), by which we then compute the numerical Hamiltonian by (2.7). The Runge-Kutta method is then used to calculate ϕ for the next time step.

Case 2. $\Delta t < \Delta t_0$**.** This Δt does not satisfy the condition (2.25), hence we would need to modify the mesh x^{n+1} by the formula

$$x_i^{n+1} = x_i^n + \Delta t \omega_i \tag{2.31}$$

using the estimated ω_i by (2.27). This new mesh x^{n+1} would lead to the satisfaction of the boundedness condition (2.25). We could then move on to compute the coefficient α_i by using the formula (2.8), by which we then compute the numerical Hamiltonian by (2.7). The Runge-Kutta method is then used to calculate ϕ for the next time step.

Similar to the case with the random mesh, for the case that the mesh moves along the characteristic lines, in order to make the mesh velocity satisfying the condition (2.25), we might need to make some modification to the mesh moving speed if necessary. If we have the mesh x_i^n and the mesh velocity ω_i dictated by the characteristic speed, using the time step Δt given by the CFL condition (2.24), we can calculate the mesh x_i^{n+1} by (2.31) also we can compute the lower bound of the allowable time step Δt_0 by (2.26), and the coefficient α^* by (2.28). Similarly as in the random mesh case, we might need to make some modifications to ω_i to ensure the condition $\Delta t \geq \Delta t_0$.

Case 1. $\Delta t \geq \Delta t_0$**.** In this case, Δt satisfies the conditions (2.24) and (2.25), hence we can keep the mesh movement speed ω_i and the mesh at the next time level x_i^{n+1} without further modification.

Case 2. $\Delta t < \Delta t_0$**.** In this case, using the formula (2.28) and (2.29), we obtain a more conservative Δt and hence a new mesh x_i^{n+1}. This leads to a new estimate of Δt_0, as well as a new mesh velocity by Δt_0 obtained from (2.26)

$$\omega_i = \frac{x_i^{n+1} - x_i^n}{\Delta t_0}. \tag{2.32}$$

Using this new ω_i and formula (2.31) we can calculate the new mesh x_i^{n+1}, which is still denoted by x_i^{n+1}.

We could then move on to compute the coefficient α_i by using the formula (2.8), by which we can compute the numerical Hamiltonian by (2.7). The Runge-Kutta method is used to calculate ϕ for the next time step.

In the two dimensional algorithm, similar adjustments for either the mesh at the next time level (for the random mesh case), or the mesh movement velocity (for the characteristic mesh movement case) would be made to ensure that the similar boundedness condition as (2.25) and the similar stability CFL condition as (2.29) are both satisfied.

3 Numerical examples

In this section we provide numerical examples to demonstrate the performance of our ALE finite difference scheme. We shall consider three types of grid movements listed as follows.

Type 1. Smoothly moving meshes.

For this type of meshes, the grid movement function is explicitly given and is a smooth function in space and time. For meshes in this type, the boundedness assumption (2.25) is automatically satisfied. We shall consider the mesh movement in one dimension as the following function

$$x_i(t) = x_i(0) + \frac{\sigma_0 (b-a) t}{T} \sin\left(\frac{2\pi (x_i(0) - a)}{b-a}\right), \tag{3.1}$$

where $[a,b]$ is the computational domain and T is the terminal time. We take $\sigma_0 = 0.1$, and $x_i(0)$ is the initial grid distribution which is assumed to be uniform (that is, $x_i(0) = a + ih$ where h is a constant).

The expression of the mesh movement in two dimensions that we consider is:

$$x_{i,j}(t)=x_{i,j}(0)+\frac{\sigma_1(b-a)t}{T}\sin\left(\frac{2\pi(x_{i,j}(0)-a)}{b-a}\right)\sin\left(\frac{2\pi(y_{i,j}(0)-c)}{d-c}\right),$$
$$y_{i,j}(t)=y_{i,j}(0)+\frac{\sigma_2(d-c)t}{T}\sin\left(\frac{2\pi(x_{i,j}(0)-a)}{b-a}\right)\sin\left(\frac{2\pi(y_{i,j}(0)-c)}{d-c}\right), \tag{3.2}$$

where $[a,b]\times[c,d]$ is the computational domain, and $(x_{i,j}(0),y_{i,j}(0))$ is the initial grid distribution which is assumed to be uniform (that is, $x_{i,j}(0)=a+ih_x$ and $y_{i,j}(0)=c+jh_y$ where both h_x and h_y are constants). T is again the terminal time. In the following examples, we choose $\sigma_1=\sigma_2=0.1$.

We remark that some adjustments might be necessary in order to avoid the deterioration of mesh quality and to satisfy the mesh movement boundedness condition (2.25), as explained in Section 2.3. We also remark that, for simplicity, we have kept the boundary points fixed (no movement in time). This is however not necessary for our algorithm.

Type 2. The randomly moving meshes.

The expression of the randomly moving mesh in one dimension is:

$$x_i^n=x_i^0+\delta r_i^n h, \tag{3.3}$$

where x_i^0 is the initial uniform mesh, namely $x_i^0=a+ih$ where h is a constant, and $r_i^n\in[-0.5,0.5]$ is a sequence of independently, uniformly distributed random numbers. The constant δ controls the deviation of the random mesh from the uniform mesh. We take $\delta=\frac{2}{3}$ in our computation below.

Likewise, the expression of the mesh movement in two dimension is:

$$x_{i,j}^n=x_{i,j}^0+\delta r_{x_{i,j}}^n h_x,\quad y_{i,j}^n=y_{i,j}^0+\delta r_{y_{i,j}}^n h_y, \tag{3.4}$$

where (x^0,y^0) is the initial grid distribution which is assumed to be uniform (that is, $x_{i,j}^0=a+ih_x$ and $y_{i,j}^0=c+jh_y$ where both h_x and h_y are constants), and $r_{x_{i,j}}^n,r_{y_{i,j}}^n\in[-0.5,0.5]$ are two sequences of independent, uniformly distributed random numbers. We again take $\delta=\frac{2}{3}$ in the following two dimensional examples. For simplicity, we have kept the boundary points fixed (no movement in time).

Type 3. The grid moves with the velocity determined by the characteristic lines, that is,

$$\omega_i^n=H_u(u_i,x_i,t^n),\qquad u_i=\frac{\phi_{x_i}^-+\phi_{x_i}^+}{2}$$

in the one-dimensional case, and

$$\omega_{x_{i,j}}^n=H_u(u_{i,j},v_{i,j};x_{i,j},y_{i,j},t^n),\qquad \omega_{y_{i,j}}^n=H_v(u_{i,j},v_{i,j};x_{i,j},y_{i,j},t^n),$$

where $u_{i,j}$ and $v_{i,j}$ are defined in (2.12), in the two-dimensional case. Again, some adjustments might be necessary in order to avoid the deterioration of mesh quality and to satisfy the mesh movement boundedness condition (2.25), as explained in Section 2.3.

We test our numerical schemes solving HJ equations in both one dimension and two dimensions using uniform meshes and the above described different types of moving meshes, to assess the performance of our third order ALE-WENO finite difference scheme.

Example 3.1. The one dimensional linear equation is

$$\begin{cases} \phi_t+\phi_x=0, & x\in[0,1], \\ \phi(x,0)=\sin(2\pi x), \end{cases} \tag{3.5}$$

with the periodic boundary condition. This example is the simplest Hamilton-Jacobi equation. We remark that even though this equation is also a simple conservation law, the method we use here is different from the conservative finite difference WENO schemes for solving conservation laws. We use the ALE-WENO scheme (2.23) to compute the equation up to the time $t=1$. The numerical errors and orders of accuracy are shown in Tables 1-4.

We can see that in this problem, the numerical scheme on both the uniform meshes and on each type of the moving meshes can achieve at least the designed third order accuracy. We remark that since the velocity along the characteristic line for this example is a constant 1, if we take the velocity of the mesh also to be 1 as the exact characteristic speed, we are in effect solving the trivial equation $\frac{d\phi}{dt}=0$ in the Lagrangian framework, and the numerical error turns out to be machine zero. Therefore, we have taken the mesh velocity $\omega(x)$ as 0.9, which is close to but not equal to the true characteristic speed. We can observe that the magnitude of the errors with the same number of mesh points is smaller for the characteristic-line-type moving mesh than for the uniform mesh, indicating that the moving mesh method may be more efficient and accurate than the fixed mesh method if the mesh movement is chosen in an appropriate way.

Table 1: Errors and numerical orders of accuracy at $t=1$ for Example 3.1 on the fixed uniform meshes with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	4.10E-02		4.56E-02		7.52E-02	
64	5.60E-03	2.87	7.78E-03	2.55	1.71E-02	2.14
128	2.61E-04	4.42	4.24E-04	4.20	1.33E-03	3.69
256	1.04E-05	4.65	1.53E-05	4.79	5.82E-05	4.51
512	7.88E-07	3.73	8.40E-07	4.19	1.16E-06	5.65

Table 2: Errors and numerical orders of accuracy at $t=1$ for Example 3.1 on the smoothly moving meshes (3.1) with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	4.66E-02		5.16E-02		7.95E-02	
64	6.94E-03	2.75	8.92E-03	2.53	1.78E-02	2.16
128	5.18E-04	3.75	8.26E-04	3.43	2.53E-03	2.81
256	1.87E-05	4.79	3.41E-05	4.60	1.60E-04	3.98
512	1.16E-06	4.02	1.30E-06	4.71	3.76E-06	5.41

Table 3: Errors and numerical orders of accuracy at $t=1$ for Example 3.1 on the randomly moving meshes (3.3) with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.34E-02		3.83E-02		6.21E-02	
64	4.57E-03	2.87	6.24E-03	2.62	1.33E-02	2.22
128	2.96E-04	3.95	4.50E-04	3.79	1.36E-03	3.30
256	1.28E-05	4.54	1.72E-05	4.71	6.00E-05	4.50
512	1.06E-06	3.59	1.16E-06	3.89	1.64E-06	5.20

Table 4: Errors and numerical orders of accuracy at $t=1$ for Example 3.1 on the characteristic-line-type moving grid with $\omega(x)=0.9$ and with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.71E-03		3.99E-03		8.58E-03	
64	3.79E-04	2.84	6.88E-04	2.54	2.05E-03	2.07
128	2.32E-05	4.03	5.36E-05	3.68	2.31E-04	3.15
256	9.91E-07	4.55	1.60E-06	5.07	7.21E-06	5.00
512	7.20E-08	3.78	7.70E-08	4.38	1.14E-07	5.98

Example 3.2. We solve the one-dimensional linear equation with variable coefficient

$$\begin{cases} \phi_t+\sin(x)\phi_x=0, & x\in[0,2\pi], \\ \phi(x,0)=\sin(x), \end{cases} \tag{3.6}$$

with periodic boundary condition. The exact solution for this problem is

$$\phi(x,t)=\sin\left(2\tan^{-1}\left(e^{-t}\tan\left(\frac{x}{2}\right)\right)\right).$$

We use the fully discrete scheme (2.23) to compute the equation up to the time $t=1$. The errors and numerical orders of accuracy are shown in Tables 5-8.

We can see the scheme with the uniform mesh or with each type of grid motions has achieved the expected third order accuracy, and it is obvious that the magnitude of the

Table 5: Errors and numerical orders of accuracy at $t=1$ for Example 3.2 on the fixed uniform meshes with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.82E-03		6.13E-03		1.44E-02	
64	2.86E-04	3.74	4.67E-04	3.71	1.04E-03	3.79
128	1.86E-05	3.94	2.98E-05	3.97	7.90E-05	3.72
256	1.81E-06	3.36	3.17E-06	3.23	1.02E-05	2.95
512	2.43E-07	2.90	4.08E-07	2.96	1.29E-06	2.98

Table 6: Errors and numerical orders of accuracy at $t=1$ for Example 3.2 on the smoothly moving meshes (3.1) with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	9.75E-04		1.57E-03		4.07E-03	
64	6.70E-05	3.86	1.29E-04	3.60	3.95E-04	3.36
128	2.96E-06	4.50	3.99E-06	5.02	8.83E-06	5.48
256	4.09E-07	2.85	5.77E-07	2.79	1.33E-06	2.74
512	5.87E-08	2.80	8.33E-08	2.79	1.81E-07	2.87

Table 7: Errors and numerical orders of accuracy at $t=1$ for Example 3.2 on the randomly moving meshes (3.3) with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	5.34E-03		7.54E-03		1.60E-02	
64	4.65E-04	3.52	7.81E-04	3.27	2.56E-03	2.64
128	3.72E-05	3.64	6.37E-05	3.62	2.67E-04	3.26
256	3.45E-06	3.43	5.56E-06	3.52	1.58E-05	4.08
512	3.69E-07	3.23	6.02E-07	3.21	1.99E-06	2.99

Table 8: Errors and numerical orders of accuracy at $t=1$ for Example 3.2 on the characteristic-line-type moving grid with $\omega(x)=\sin x$ and with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.84E-05		5.46E-05		2.24E-04	
64	1.06E-06	5.18	2.00E-06	4.77	1.21E-05	4.21
128	5.92E-08	4.16	7.38E-08	4.76	2.26E-07	5.73
256	5.99E-09	3.30	6.71E-09	3.46	1.34E-08	4.08
512	7.03E-10	3.09	7.89E-10	3.09	1.54E-09	3.12

error on the grid moving along the characteristic line is much smaller than that on the fixed uniform grid with the same number of mesh nodes.

Example 3.3. We solve the one-dimensional Burgers equation

$$\begin{cases} \phi_t+\frac{1}{2}(\phi_x+1)^2=0, & x\in[-1,1], \\ \phi(x,0)=-\cos(\pi x), \end{cases} \tag{3.7}$$

with a periodic boundary condition. We calculate the equation first to $t=\frac{0.5}{\pi^2}$ by using our ALE-WENO scheme (2.23). At this time, the solution is still smooth. We test the scheme on different types of grids, and list the errors and numerical order of accuracy in Tables 9-12. Again, we observe that the designed third order accuracy is achieved for all types of moving meshes, and it is more efficient to use the characteristic-line-type grid motion.

Table 9: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.3 on the fixed uniform meshes with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.08E-03		3.56E-03		1.06E-02	
64	3.00E-04	2.79	6.47E-04	2.46	2.50E-03	2.08
128	1.07E-05	4.82	2.47E-05	4.71	1.28E-04	4.29
256	4.35E-07	4.62	6.07E-07	5.35	2.18E-06	5.87
512	3.69E-08	3.56	5.01E-08	3.60	1.55E-07	3.82

Table 10: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.3 on the smoothly moving meshes (3.1) with N grid points.

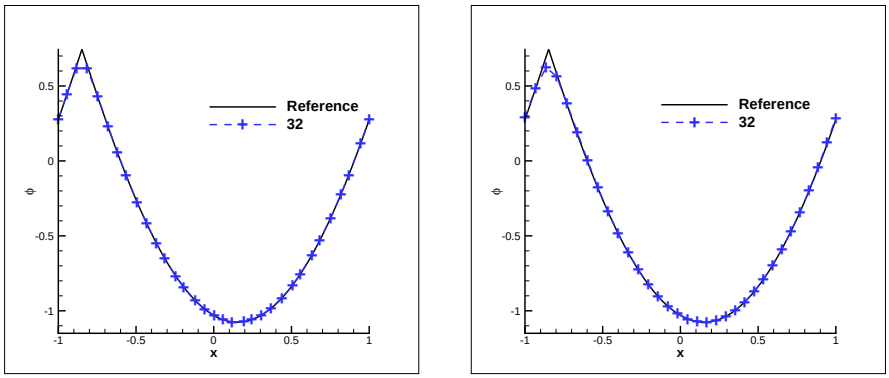
N	L_1 error	order	L_2 error	order	L_∞ error	order
32	6.93E-03		1.09E-02		2.83E-02	
64	1.09E-03	2.67	2.02E-03	2.43	6.12E-03	2.21
128	6.31E-05	4.11	1.44E-04	3.80	6.26E-04	3.29
256	2.36E-06	4.74	4.27E-06	5.08	1.90E-05	5.04
512	1.19E-07	4.31	1.61E-07	4.73	5.28E-07	5.17

Table 11: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.3 on the randomly moving meshes (3.3) with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.03E-03		3.50E-03		1.09E-02	
64	2.60E-04	2.96	5.40E-04	2.70	2.06E-03	2.40
128	1.09E-05	4.58	2.49E-05	4.44	1.29E-04	4.00
256	4.70E-07	4.53	6.65E-07	5.23	2.28E-06	5.82
512	3.93E-08	3.58	5.37E-08	3.63	1.75E-07	3.70

Table 12: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.3 on the characteristic-line-type moving grid with $\omega(x)=\phi_x+1$ and with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.92E-04		6.03E-04		2.08E-03	
64	4.40E-05	2.73	1.05E-04	2.52	4.74E-04	2.13
128	1.89E-06	4.54	4.56E-06	4.53	2.68E-05	4.14
256	7.80E-08	4.60	9.68E-08	5.56	2.71E-07	6.63
512	9.49E-09	3.04	1.15E-08	3.07	2.30E-08	3.56



(a) on the fixed uniform mesh

(b) on the smoothly moving mesh

(c) on the randomly moving mesh

(d) on the characteristic-line-type moving mesh

Figure 4: ϕ at $t=\frac{1.5}{\pi^2}$ with $N=32$ grid points for Example 3.3.

When we calculate the equation to the time $t=\frac{1.5}{\pi^2}$, the solution is not smooth any more. We use the the numerical solution on the fixed uniform grid with 2048 nodes as the reference solution, and plot the results of our ALE-WENO scheme (2.23) in Fig. 4. We can see that our scheme can achieve high resolution in this example. Especially, it is obvious

that on the characteristic-line-type moving grid, the mesh becomes dense near the corner singularity and this helps to get better resolution there than on the fixed uniform grid and other types of moving grids.

Example 3.4. We solve the one-dimensional nonlinear problem

$$\begin{cases} \phi_t - \cos(\phi_x + 1) = 0, & x \in [-1,1], \\ \phi(x,0) = -\cos(\pi x), \end{cases} \tag{3.8}$$

with periodic boundary condition. This is a nonconvex Hamiltonian problem. We use our ALE-WENO scheme (2.23) to calculate the solution first up to the time $t = \frac{0.5}{\pi^2}$, when the solution is still smooth, and we list the errors and numerical orders of accuracy in Tables 13-16. We can clearly see the same performance as in the previous examples at the error order accuracy and the superiority of the characteristic-line-type moving mesh.

Table 13: Errors and numerical orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.4 on the fixed uniform meshes with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	4.27E-04		7.52E-04		1.82E-03	
64	6.41E-05	2.74	1.28E-04	2.55	4.32E-04	2.07
128	2.78E-06	4.53	5.68E-06	4.50	2.63E-05	4.04
256	1.58E-07	4.14	2.15E-07	4.72	7.35E-07	5.16
512	1.78E-08	3.15	2.57E-08	3.07	9.92E-08	2.89

Table 14: Errors and numerical orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.4 on the smoothly moving meshes (3.1) with N grid points.

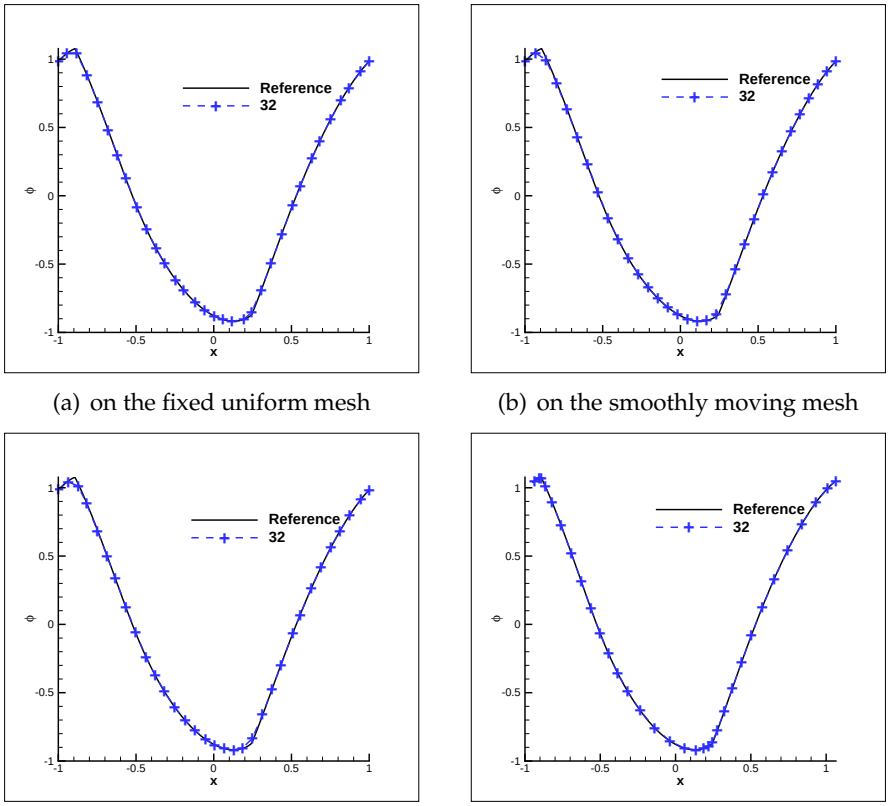
N	L_1 error	order	L_2 error	order	L_∞ error	order
32	6.91E-03		9.19E-03		2.06E-02	
64	6.90E-04	3.32	1.09E-03	3.07	3.42E-03	2.59
128	3.23E-05	4.42	5.85E-05	4.23	2.57E-04	3.74
256	1.64E-06	4.30	2.35E-06	4.64	7.00E-06	5.20
512	1.32E-07	3.64	1.88E-07	3.64	8.95E-07	2.97

Table 15: Errors and numerical orders of accuracy at $t = \frac{0.5}{\pi^2}$ for Example 3.4 on the randomly moving meshes (3.3) with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	5.09E-04		8.40E-04		2.13E-03	
64	7.16E-05	2.83	1.35E-04	2.64	4.40E-04	2.27
128	3.40E-06	4.40	6.22E-06	4.44	2.65E-05	4.05
256	2.12E-07	4.01	2.96E-07	4.40	9.66E-07	4.78
512	2.34E-08	3.18	3.75E-08	2.98	1.64E-07	2.56

Table 16: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.4 on the characteristic-line-type moving grid with $\omega(x)=\sin(\phi_x+1)$ and with N grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.53E-05		5.82E-05		1.70E-04	
64	8.11E-06	2.12	1.76E-05	1.72	7.36E-05	1.21
128	5.33E-07	3.93	1.03E-06	4.10	4.92E-06	3.90
256	3.79E-08	3.81	4.97E-08	4.37	1.32E-07	5.22
512	4.64E-09	3.03	6.22E-09	3.00	1.71E-08	2.95



(a) on the fixed uniform mesh

(b) on the smoothly moving mesh

(c) on the randomly moving mesh

(d) on the characteristic-line-type moving mesh

Figure 5: ϕ at $t=\frac{1.5}{\pi^2}$ with $N=32$ grid points for Example 3.4.

Similar to Example 3.3, we compute the solution to the time $t=\frac{1.5}{\pi^2}$, when a corner singularity has already appeared. We again choose the numerical solution on the fixed uniform mesh with 2048 nodes as the reference solution in the figures. The numerical results are shown in Fig. 5. We observe similar performance as in Example 3.3. In par-

ticular, the characteristic-line-type moving mesh is more efficient than the other types of moving meshes.

Example 3.5. We solve the problem

$$\begin{cases} \phi_t+\frac{1}{4}(\phi_x^2-1)(\phi_x^2-4)=0, \quad x\in[-2.0,2.0], \\ \phi(x,0)=-2|x|. \end{cases} \tag{3.9}$$

The boundary conditions are taken as that of the initial condition there, as waves have not reached the boundary yet at the terminal time. This example comes from [28], and is a very important test case as many schemes can not get numerical convergence to the correct viscosity solution for this problem. We compute the equation up to $t=1$ by using our ALE-WENO scheme (2.23), and plot the results in Fig. 6, in which we compare the result of the characteristic-line-type moving grids with $N=32$ grid points to that with $N=256$ grid points, against the "exact" reference solution, and compare the results of the characteristic-line-type moving grid and the fixed uniform grid both with $N=32$ grid points. This example is not a periodic problem, the characteristic-line-type moving grid will change the original computational region. In order to reach the desired computational region at the terminal time, we would need use a larger computational domain initially. Therefore, we choose the initial computational domain as $x\in[-2.0,2.0]$ and plot the solution at the final time in the domain $x\in[-1.0,1.0]$. From the figure, we can clearly observe that our scheme can numerically converge to the correct viscosity solution, and the characteristic-line-type moving grid produces more accurate results on meshes with the same number of grid points.

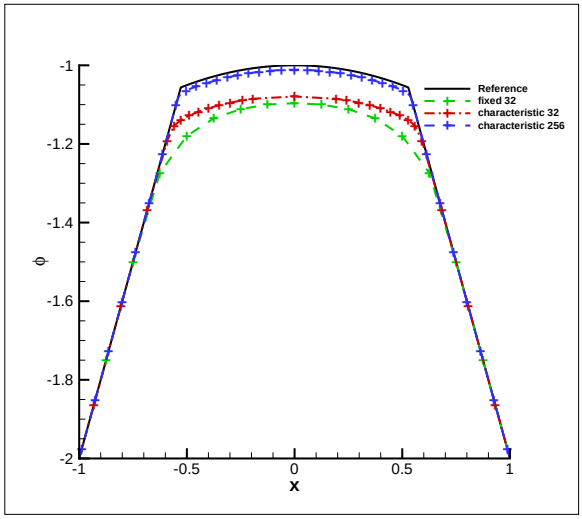


Figure 6: ALE-WENO solution at $t=1.0$ in Example 3.5.

Example 3.6. We solve the simplest two-dimensional linear equation

$$\begin{cases} \phi_t+\phi_x+\phi_y=0, & (x,y)\in[0,1]^2, \\ \phi(x,y,0)=\sin(2\pi(x+y)), \end{cases} \tag{3.10}$$

with periodic boundary condition. We can see the results obtained by our scheme (2.23) on different types of meshes in Tables 17-20. Similar patterns as in the one-dimensional case can be observed, namely the scheme obtains the designed third order accuracy for all types of meshes, and the results from the characteristic-line-type moving grids (we again use an approximate characteristic velocity of $(0.9,0.9)$ for the mesh movement to avoid solving the trivial equation $\frac{d\phi}{dt}=0$) are more accurate than those from fixed uniform meshes for the same number of mesh points.

Table 17: Errors and numerical orders of accuracy at $t=1$ for Example 3.6 on the fixed uniform meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	5.64E-02		6.21E-02		9.29E-02	
64	7.13E-03	2.98	9.87E-03	2.65	1.92E-02	2.27
128	5.34E-04	3.74	8.88E-04	3.47	2.32E-03	3.05
256	2.52E-05	4.41	3.57E-05	4.64	1.09E-04	4.41
512	1.74E-06	3.85	1.79E-06	4.32	2.27E-06	5.59

Table 18: Errors and numerical orders of accuracy at $t=1$ for Example 3.6 on the smoothly moving meshes (3.2) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	7.28E-02		7.77E-02		1.14E-01	
64	9.78E-03	2.90	1.28E-02	2.61	2.52E-02	2.18
128	8.37E-04	3.55	1.30E-03	3.29	3.62E-03	2.80
256	4.93E-05	4.09	7.27E-05	4.16	2.54E-04	3.83
512	3.36E-06	3.88	3.57E-06	4.35	8.51E-06	4.90

Table 19: Errors and numerical orders of accuracy at $t=1$ for Example 3.6 on the randomly moving meshes (3.4) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	6.01E-02		6.60E-02		9.85E-02	
64	7.34E-03	3.03	1.02E-02	2.69	1.98E-02	2.31
128	5.38E-04	3.77	9.12E-04	3.49	2.39E-03	3.05
256	2.41E-05	4.48	3.61E-05	4.66	1.14E-04	4.39
512	1.56E-06	3.94	1.61E-06	4.49	2.22E-06	5.69

Table 20: Errors and numerical orders of accuracy at $t=1$ for Example 3.6 on the characteristic-line-type moving grid with $\omega(x,y)=(0.9,0.9)$ and with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	4.87E-03		6.63E-03		1.20E-02	
64	7.21E-04	2.76	1.26E-03	2.40	3.14E-03	1.94
128	6.47E-05	3.48	1.38E-04	3.19	4.79E-04	2.71
256	2.62E-06	4.63	4.30E-06	5.00	1.67E-05	4.84
512	1.65E-07	3.98	1.70E-07	4.66	2.39E-07	6.13

Example 3.7. We solve the two-dimensional linear equation with variable coefficients

$$\begin{cases} \phi_t+\sin(\frac{x+y}{2})(\phi_x+\phi_y)=0, \quad (x,y)\in[0,4\pi]^2, \\ \phi(x,y,0)=\sin(\frac{x+y}{2}), \end{cases} \tag{3.11}$$

with periodic boundary condition. We list the errors and numerical orders of accuracy simulated by the ALE-WENO scheme (2.23) in Tables 21-24. We can clearly see that we have achieved the designed third order accuracy. Also, it is more efficient to use the characteristic-line-type grid motion because its error is smaller than that on the fixed uniform grid with the same number of grid points. In this case, the computational domain is relatively large, hence for coarse meshes h is quite large. In order to reduce the error in the time direction, when Δt determined by (2.24) is larger than h, we set it to be h.

Table 21: Errors and numerical orders of accuracy at $t=1$ for Example 3.7 on the fixed uniform meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.77E-03		4.47E-03		9.91E-03	
64	2.18E-04	3.67	4.12E-04	3.44	1.74E-03	2.51
128	1.91E-05	3.51	3.88E-05	3.41	1.87E-04	3.22
256	2.16E-06	3.15	3.55E-06	3.45	1.02E-05	4.19
512	2.55E-07	3.08	4.14E-07	3.10	1.28E-06	2.99

Table 22: Errors and numerical orders of accuracy at $t=1$ for Example 3.7 on the smoothly moving meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	5.67E-03		1.31E-02		7.82E-02	
64	9.80E-04	2.53	3.16E-03	2.06	2.15E-02	1.86
128	1.48E-04	2.72	5.79E-04	2.45	4.77E-03	2.17
256	1.47E-05	3.34	5.34E-05	3.44	4.84E-04	3.30
512	1.26E-06	3.54	3.50E-06	3.93	2.80E-05	4.11

Table 23: Errors and numerical orders of accuracy at $t=1$ for Example 3.7 on the randomly moving meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	7.72E-03		1.45E-02		5.23E-02	
64	5.77E-04	3.74	1.65E-03	3.14	8.86E-03	2.56
128	4.12E-05	3.81	1.35E-04	3.60	9.62E-04	3.20
256	2.80E-06	3.88	5.43E-06	4.64	3.33E-05	4.85
512	2.73E-07	3.36	4.61E-07	3.56	1.52E-06	4.45

Table 24: Errors and numerical orders of accuracy at $t=1$ for Example 3.7 on the characteristic-line-type moving grid with $\omega(x,y)=(\sin(\frac{x+y}{2}),\sin(\frac{x+y}{2}))$ and with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	6.45E-05		1.23E-04		6.31E-04	
64	3.51E-06	4.20	5.56E-06	4.46	3.25E-05	4.28
128	4.00E-07	3.13	4.65E-07	3.58	9.90E-07	5.03
256	4.96E-08	3.01	5.74E-08	3.02	1.32E-07	2.90
512	6.15E-09	3.01	7.17E-09	3.00	1.71E-08	2.96

Example 3.8. We solve the two-dimensional linear equation with variable coefficients

$$\begin{cases} \phi_t - y\phi_x + x\phi_y = 0, & (x,y)\in[-1,1]^2, \\ \phi(x,y,0) = \begin{cases} 0.2, & r\le 0.1, \\ 0.0, & r\ge 0.3, \\ 0.3-r, & \text{others}, \end{cases} \end{cases} \tag{3.12}$$

where $r=\sqrt{(x-0.4)^2+(y-0.4)^2}$. We use the ALE-WENO scheme (2.23) to compute to $t=1.0$ and we plot the results on different types of moving grids in Figs. 7-11. We also plot the line cut along $x=-0.13$ for the comparison of results on 32×32 grid points, obtained from the characteristic-line-type moving grid and the fixed uniform grid against the exact reference solution, in Fig. 11. We clearly observe much better resolution using the characteristic-line-type moving grid than using the fixed uniform grid with the same number of grid points.

Example 3.9. We solve the two dimensional Burgers equation

$$\begin{cases} \phi_t + \frac{1}{2}(\phi_x+\phi_y+1)^2 = 0, & (x,y)\in[-2,2]^2, \\ \phi(x,y,0) = -\cos(\pi\frac{x+y}{2}), \end{cases} \tag{3.13}$$

with periodic boundary conditions. At the time $t=\frac{0.5}{\pi^2}$, the solution is still smooth. We list the errors and numerical orders of accuracy simulated by the ALE-WENO scheme (2.23)

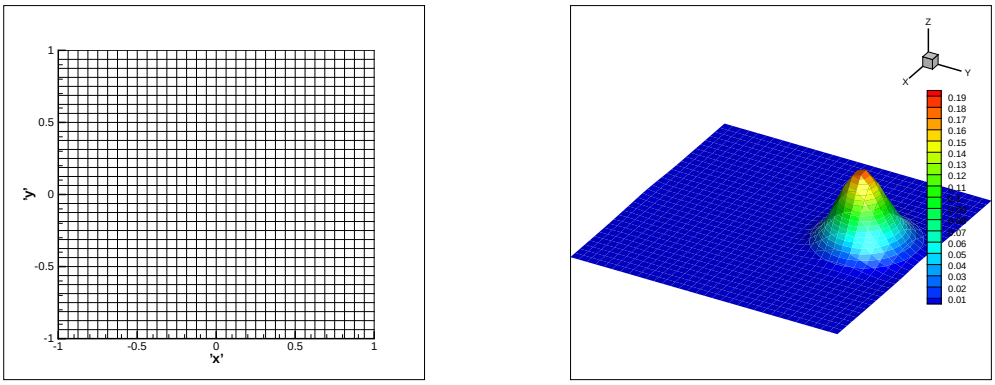


Figure 7: The mesh and contours of ϕ at $t=1.0$ for Example 3.8 on the fixed uniform mesh, 32×32 grid points.

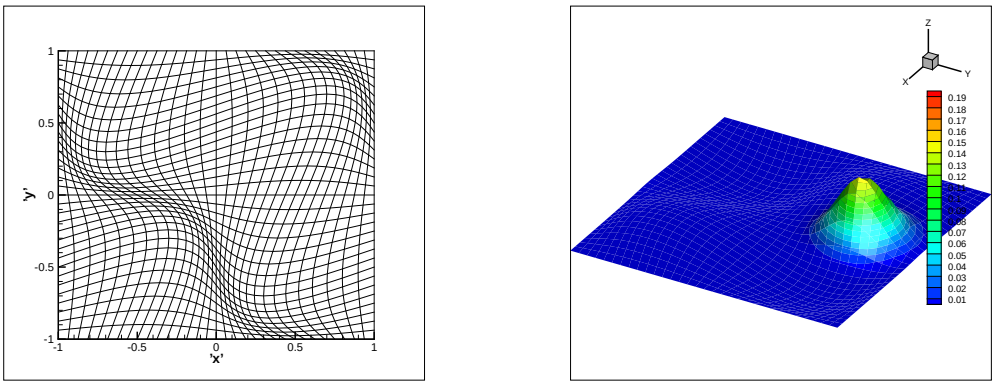


Figure 8: The mesh and contours of ϕ at $t=1.0$ for Example 3.8 on the smoothly moving mesh (3.2), 32×32 grid points.

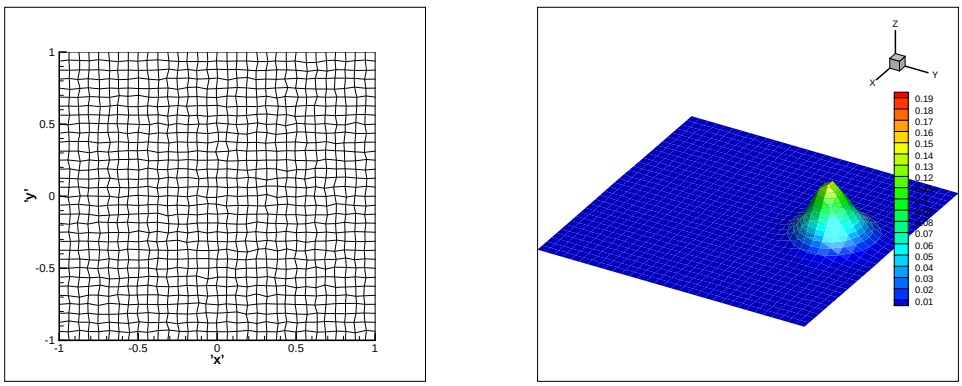


Figure 9: The mesh and contour of ϕ at $t=1.0$ for Example 3.8 on the randomly moving mesh, 32×32 grid points.

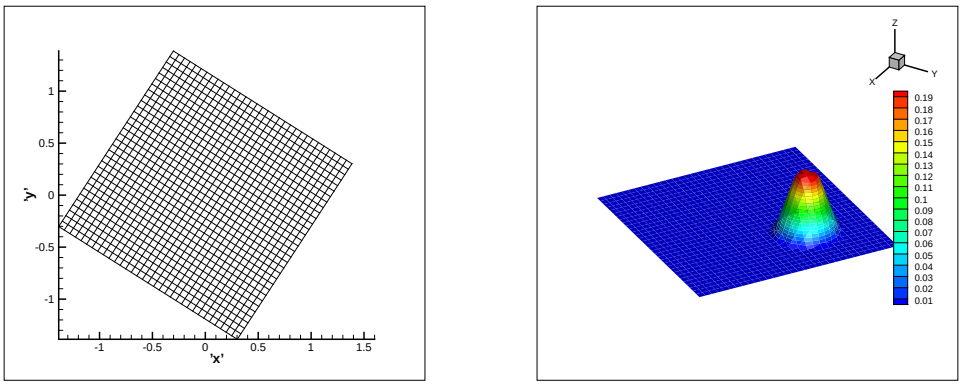


Figure 10: The mesh and contour of ϕ at $t=1.0$ for Example 3.8 on the characteristic-line-type moving meshes with $\omega(x,y)=(-y,x)$, 32×32 grid points.

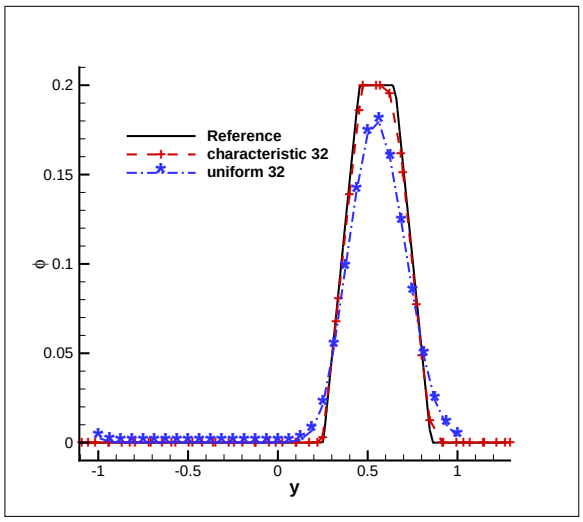


Figure 11: ALE-WENO solution at $t=1.0$ in Example 3.8 cut along $x=-0.13$ for the comparison of results from the characteristic-line-type moving grid and the fixed uniform grid against the exact reference solution, 32×32 grid points.

in Tables 25-28. It can again be observed that our ALE-WENO scheme (2.23) on all types of grid motions can achieve the designed third order accuracy. Also from the error tables, we can see that, at least for the L_∞ norm on finer grids, we have smaller errors by the moving grid along the characteristic lines than by the fixed uniform mesh with the same number of grid points.

Next, we compute the equation to $t=\frac{1.5}{\pi^2}$ by our scheme (2.23), when the solution has already developed corner singularities. Since the moving speed of the mesh follows the characteristic line, the mesh might intersect when the solution is not smooth. When the mesh is seriously distorted, we apply a smoothing filter to smooth out the mesh (e.g. [26]). To be more specific, when the local mesh size $h\leq0.2h_0$ (where h_0 is the initial uniform

Table 25: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 on the fixed uniform meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.47E-03		3.55E-03		7.99E-03	
64	2.61E-04	3.24	4.50E-04	2.98	1.28E-03	2.64
128	1.09E-05	4.59	1.55E-05	4.86	4.72E-05	4.76
256	6.74E-07	4.01	9.94E-07	3.97	3.75E-06	3.66
512	8.51E-08	2.99	1.23E-07	3.02	4.69E-07	3.00

Table 26: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 on the smoothly moving meshes (3.2) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.84E-03		5.43E-03		1.47E-02	
64	4.57E-04	3.07	7.33E-04	2.89	2.57E-03	2.51
128	2.96E-05	3.95	4.89E-05	3.90	1.94E-04	3.73
256	2.17E-06	3.77	3.90E-06	3.65	2.24E-05	3.11
512	2.59E-07	3.07	4.74E-07	3.04	2.82E-06	2.99

Table 27: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 on the randomly moving meshes (3.4) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.61E-03		3.76E-03		8.69E-03	
64	2.68E-04	3.29	4.62E-04	3.02	1.37E-03	2.67
128	1.08E-05	4.63	1.56E-05	4.89	4.84E-05	4.82
256	6.55E-07	4.05	9.93E-07	3.97	3.88E-06	3.64
512	8.42E-08	2.96	1.23E-07	3.02	4.80E-07	3.02

Table 28: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 on the characteristic-line-type moving grid with $\omega(x,y)=(\phi_x+\phi_y+1,\phi_x+\phi_y+1)$ and with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	3.25E-03		4.73E-03		1.03E-02	
64	3.58E-04	3.18	6.35E-04	2.90	1.68E-03	2.62
128	1.36E-05	4.72	1.99E-05	5.00	5.72E-05	4.87
256	7.88E-07	4.11	8.93E-07	4.48	1.60E-06	5.16
512	9.76E-08	3.01	1.03E-07	3.11	1.41E-07	3.51

mesh size) or when the mesh is intersecting, we use $x_{i,j}^{n+1}\leftarrow\frac{1}{4}x_{i-1,j}^{n+1}+\frac{1}{2}x_{i,j}^{n+1}+\frac{1}{4}x_{i+1,j}^{n+1}$ and $y_{i,j}^{n+1}\leftarrow\frac{1}{4}y_{i,j-1}^{n+1}+\frac{1}{2}y_{i,j}^{n+1}+\frac{1}{4}y_{i,j+1}^{n+1}$ to obtain the new mesh at time level $(n+1)$. This filtering

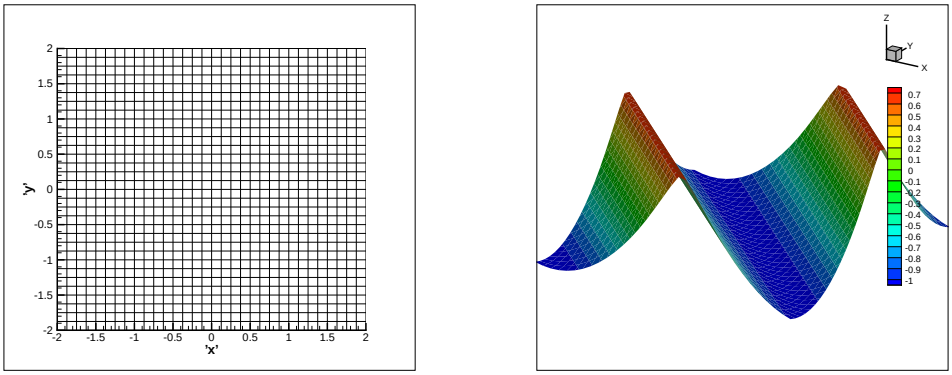


Figure 12: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.9 on the fixed uniform grid, 32×32 grid points.

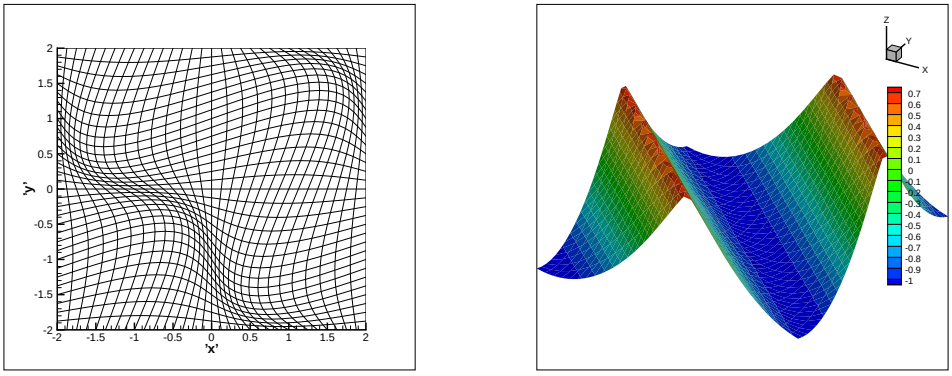


Figure 13: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.9 on the smoothly moving grid (3.2), 32×32 grid points.

procedure is repeated until the mesh at the $(n+1)$-th time level is satisfactory. We plot here the solutions on the fixed uniform grid and on the three types of moving meshes in Figs. 12-15. We also plot the line cut along $x=0$ for the comparison of results on 32×32 grid points, obtained from the characteristic-line-type moving grid and the fixed uniform grid against the "exact" reference solution, in Fig. 16. It can be observed that the scheme again provides non-oscillatory solutions with good resolution for all types of grid motions. We observe once again that, in the case of the moving grid along the characteristic lines, the mesh becomes dense near the corner singularities, helping us to capture these singularities better than the fixed uniform mesh.

One thing we can observe from Tables 25 and 28 is: the advantage of the characteristically moving meshes over fixed uniform meshes, in terms of smaller errors on the meshes with the same number of grid points, is less apparent in this two-dimensional example than in the one-dimensional Example 3.3. To further explore this, we use the same equation (3.13), and choose the initial condition as $\phi(x,y,0)=-0.1\cos(\pi\frac{x+y}{2})$. In this case, the solution will stay smooth for much longer time. If we run to the same time $t=\frac{0.5}{\pi^2}$, and

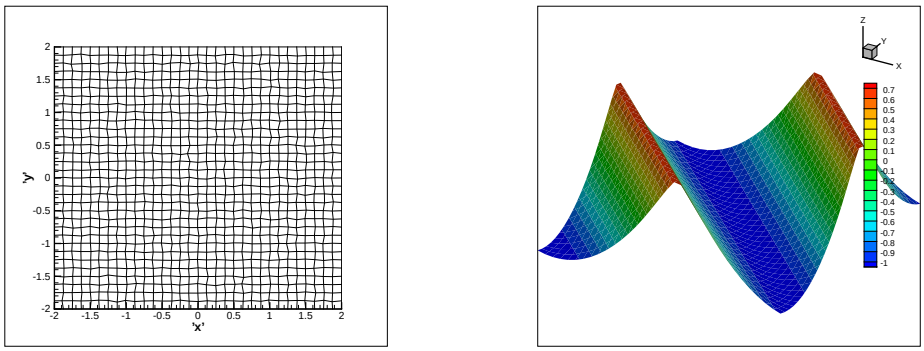


Figure 14: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.9 on the randomly moving grid, 32×32 grid points.

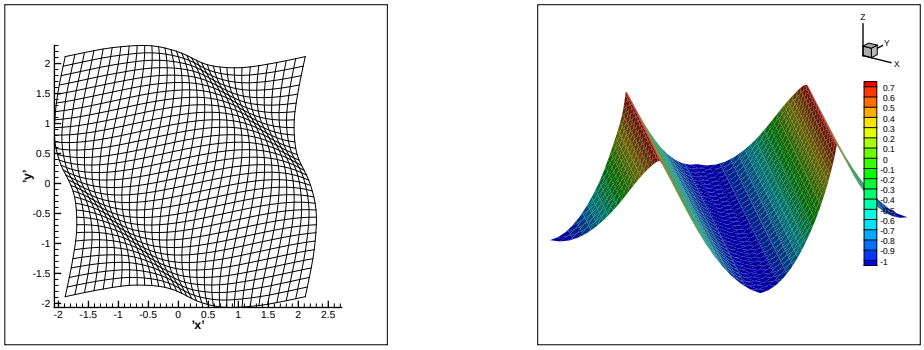


Figure 15: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.9 on the characteristic-line-type moving grid with $\omega(x,y)=(\phi_x+\phi_y+1,\phi_x+\phi_y+1)$, 32×32 grid points.

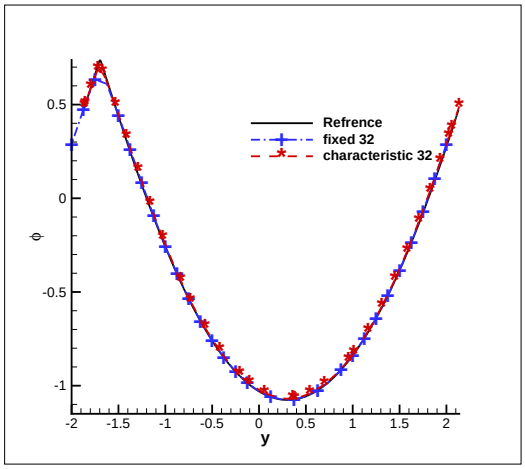


Figure 16: ALE-WENO solution at $t=\frac{1.5}{\pi^2}$ in Example 3.9 cut along $x=0$ for the comparison of results from the characteristic-line-type moving grid and the fixed uniform grid against the "exact" reference solution, 32×32 grid points.

Table 29: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 with the initial condition $\phi(x,y,0)=-0.1\cos(\pi\frac{x+y}{2})$ on the fixed uniform meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.44E-05		2.81E-05		5.79E-05	
64	1.29E-06	4.24	1.38E-06	4.35	1.87E-06	4.95
128	1.41E-07	3.20	1.55E-07	3.15	2.35E-07	3.00
256	1.74E-08	3.01	1.93E-08	3.01	2.94E-08	3.00
512	2.18E-09	3.00	2.41E-09	3.00	3.68E-09	3.00

Table 30: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 with the initial condition $\phi(x,y,0)=-0.1\cos(\pi\frac{x+y}{2})$ on the smoothly moving meshes (3.2) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	7.36E-04		1.61E-03		8.70E-03	
64	4.68E-05	3.97	9.82E-05	4.03	5.35E-04	4.02
128	3.25E-06	3.85	6.68E-06	3.88	3.58E-05	3.90
256	2.37E-07	3.78	4.71E-07	3.83	2.44E-06	3.87
512	1.93E-08	3.62	3.62E-08	3.70	1.74E-07	3.81

Table 31: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 with the initial condition $\phi(x,y,0)=-0.1\cos(\pi\frac{x+y}{2})$ on the randomly moving meshes (3.4) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.69E-05		3.11E-05		7.16E-05	
64	1.32E-06	4.36	1.41E-06	4.47	2.29E-06	4.97
128	1.39E-07	3.24	1.55E-07	3.18	2.80E-07	3.03
256	1.71E-08	3.03	1.91E-08	3.02	3.79E-08	2.89
512	2.13E-09	3.00	2.38E-09	3.00	4.93E-09	2.94

list the errors and numerical orders of accuracy simulated by the ALE-WENO scheme (2.23) in Tables 29-32, we again observe that our ALE-WENO scheme (2.23) on all types of grid motions can achieve the designed third order accuracy. Also from the error tables, we can see that we now have smaller errors by the moving grid along the characteristic lines than by the fixed uniform mesh with the same number of grid points in all norms and for all mesh sizes. This indicates that the moving meshes along the characteristic lines would be of advantage also in two dimensions for reducing errors, at least for very smooth solutions.

Table 32: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.9 with the initial condition $\phi(x,y,0)=-0.1\cos(\pi\frac{x+y}{2})$ on the characteristic-line-type moving grid with $\omega(x,y)=(\phi_x+\phi_y+1,\phi_x+\phi_y+1)$ and with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	9.77E-06		1.22E-05		2.32E-05	
64	4.51E-07	4.44	4.74E-07	4.68	6.00E-07	5.27
128	4.76E-08	3.25	5.10E-08	3.22	6.61E-08	3.18
256	5.90E-09	3.01	6.33E-09	3.01	8.21E-09	3.01
512	7.38E-10	3.00	7.91E-10	3.00	1.03E-09	3.00

Example 3.10. We solve the two dimensional nonlinear equation

$$\begin{cases} \phi_t-\cos(\phi_x+\phi_y+1)=0, & (x,y)\in[-2,2]^2, \\ \phi(x,y,0)=-\cos(\pi\frac{x+y}{2}), \end{cases} \tag{3.14}$$

with periodic boundary condition. We compute the equation up to the time $t=\frac{0.5}{\pi^2}$. The errors and numerical orders of accuracy are shown in Tables 33-36. We can clearly see similar performance as in the previous Example 3.9 for the errors and orders of accuracy, and the superiority of the characteristic-line-type moving mesh.

Table 33: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.10 on the fixed uniform meshes with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	4.46E-04		6.62E-04		1.52E-03	
64	5.40E-05	3.05	9.30E-05	2.83	2.87E-04	2.41
128	2.72E-06	4.31	4.20E-06	4.47	2.06E-05	3.80
256	2.22E-07	3.62	4.14E-07	3.34	2.69E-06	2.94
512	2.77E-08	3.00	5.18E-08	3.00	3.39E-07	2.98

Table 34: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.10 on the smoothly moving meshes (3.2) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.13E-02		4.66E-02		2.49E-01	
64	1.47E-03	3.86	3.09E-03	3.91	1.58E-02	3.97
128	9.26E-05	3.99	2.01E-04	3.94	1.18E-03	3.75
256	6.35E-06	3.87	1.40E-05	3.84	8.86E-05	3.73
512	5.13E-07	3.63	1.19E-06	3.56	1.02E-05	3.11

Table 35: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.10 on the randomly moving meshes (3.4) with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	6.27E-04		9.18E-04		2.45E-03	
64	6.49E-05	3.27	1.12E-04	3.04	3.71E-04	2.72
128	3.00E-06	4.44	4.81E-06	4.54	2.58E-05	3.85
256	2.32E-07	3.69	4.64E-07	3.37	3.12E-06	3.04
512	2.80E-08	3.05	5.66E-08	3.04	3.74E-07	3.06

Table 36: Errors and numerical orders of accuracy at $t=\frac{0.5}{\pi^2}$ for Example 3.10 on the characteristic-line-type moving grid with $\omega(x,y)=(\sin(\phi_x+\phi_y+1),\sin(\phi_x+\phi_y+1))$ and with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	9.33E-04		1.39E-03		2.84E-03	
64	1.08E-04	3.12	1.96E-04	2.82	5.64E-04	2.33
128	4.80E-06	4.49	6.49E-06	4.92	1.80E-05	4.97
256	3.43E-07	3.81	4.19E-07	3.95	1.01E-06	4.15
512	4.17E-08	3.04	5.15E-08	3.02	1.27E-07	3.00

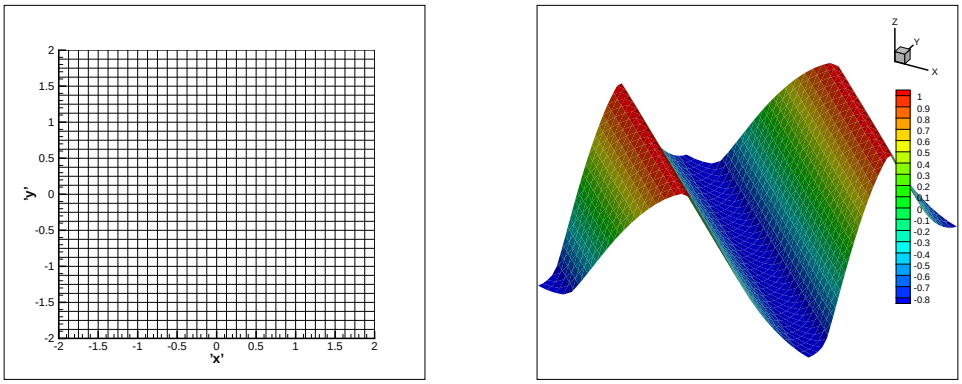


Figure 17: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.10 on the fixed uniform grid, 32×32 grid points.

The mesh and the contours of ϕ obtained by the scheme (2.23) at $t=\frac{1.5}{\pi^2}$, when the solution is no longer smooth, are shown in Figs. 17-20. Similar to Example 3.9, when the moving speed of the mesh follows the characteristic line, the mesh might intersect after the solution is no longer smooth. The same smoothing filter is then again used to smooth out the mesh in this case. We also plot the line cut along $x=0$ for the comparison of results on 32×32 grid points, obtained from the characteristic-line-type moving grid and the fixed uniform grid against the "exact" reference solution, in Fig. 21. We can observe that the performance of this example is similar to Example 3.9 in capturing corners.

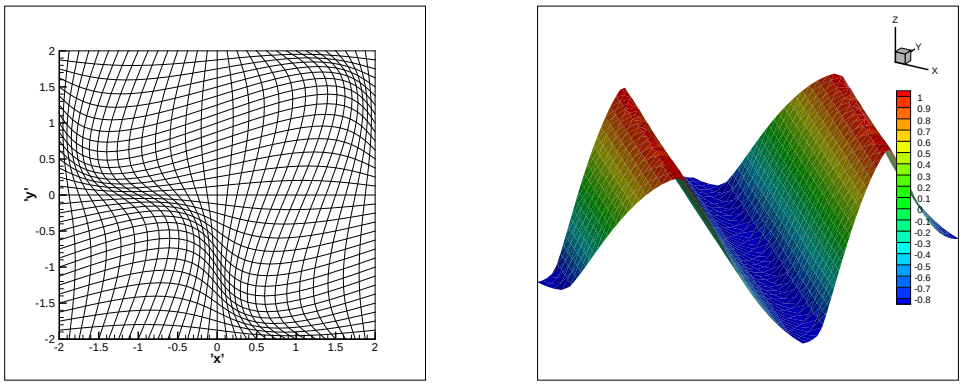


Figure 18: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.10 on the smoothly moving grid (3.2), 32×32 grid points.

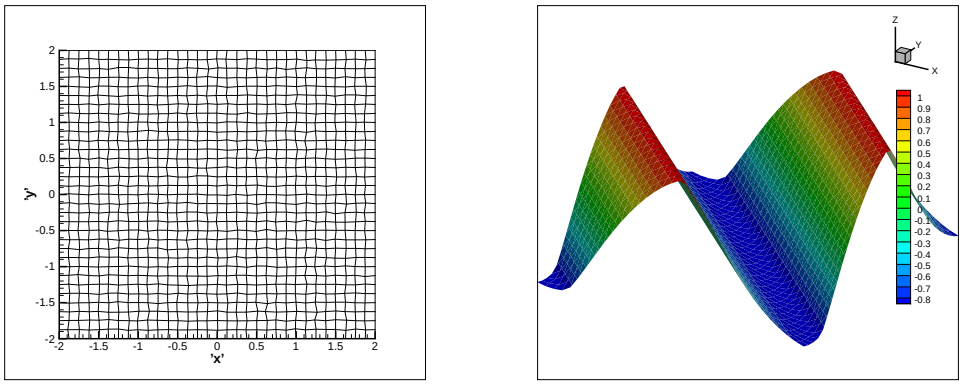


Figure 19: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.10 on the randomly moving grid, 32×32 grid points.

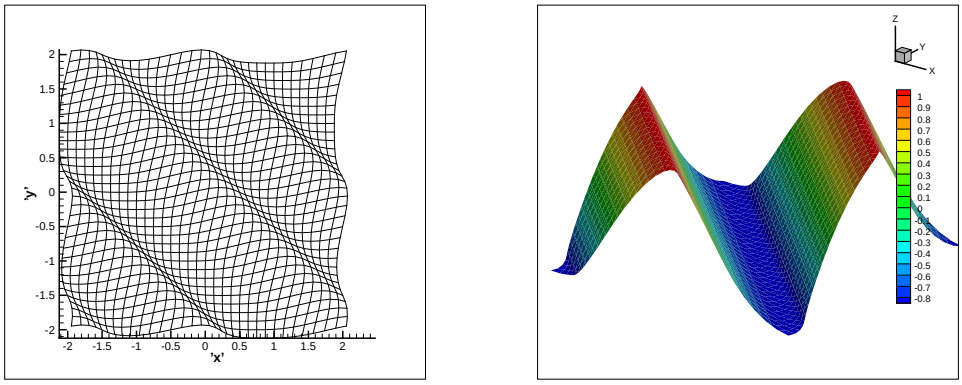


Figure 20: The mesh and contours of ϕ at $t=\frac{1.5}{\pi^2}$ for Example 3.10 on the characteristic-line-type moving grid with $\omega(x,y)=(\sin(\phi_x+\phi_y+1),\sin(\phi_x+\phi_y+1))$, 32×32 grid points.

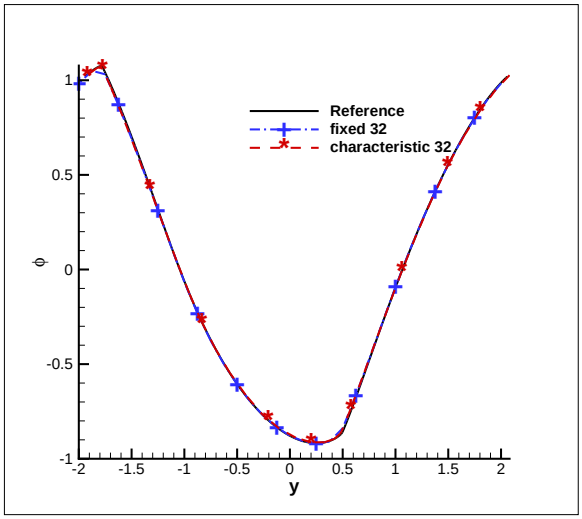


Figure 21: ALE-WENO solution at $t=\frac{1.5}{\pi^2}$ in Example 3.10 cut along $x=0$ for the comparison of results from the characteristic-line-type moving grid and the fixed uniform grid against the "exact" reference solution, 32×32 grid points.

Example 3.11. We solve the two dimensional Eikonal equation

$$\begin{cases} \phi_t+\sqrt{\phi_x^2+\phi_y^2+1}=0, & (x,y)\in[0,1]^2, \\ \phi(x,y,0)=\frac{1}{4}(\cos(2\pi x)-1)(\cos(2\pi y)-1)-1, \end{cases} \tag{3.15}$$

and compute the problem to the time $t=0.6$ by using the scheme (2.23). The numerical results are shown in Figs. 22-25. From this example, it can be observed that the schemes provide non-oscillatory solutions with good resolution for all types of grid motions. Obviously, we can easily observe that in the case of the characteristic-line type moving mesh,

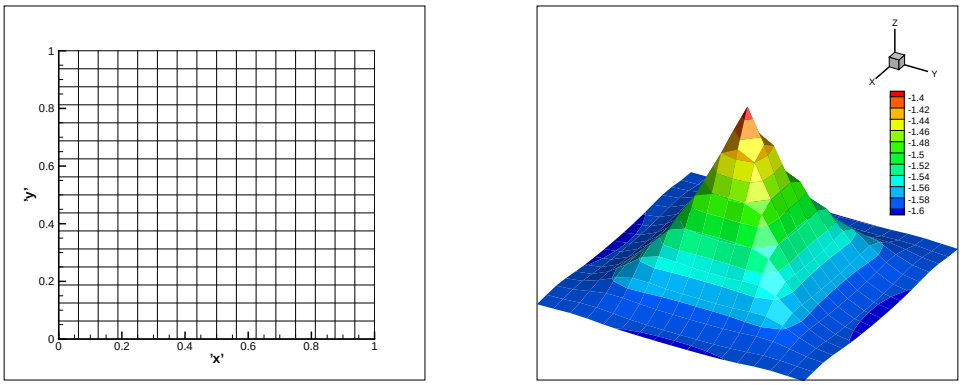


Figure 22: The mesh and contours of ϕ at $t=0.6$ for Example 3.11 on the fixed uniform mesh, 16×16 grid points.

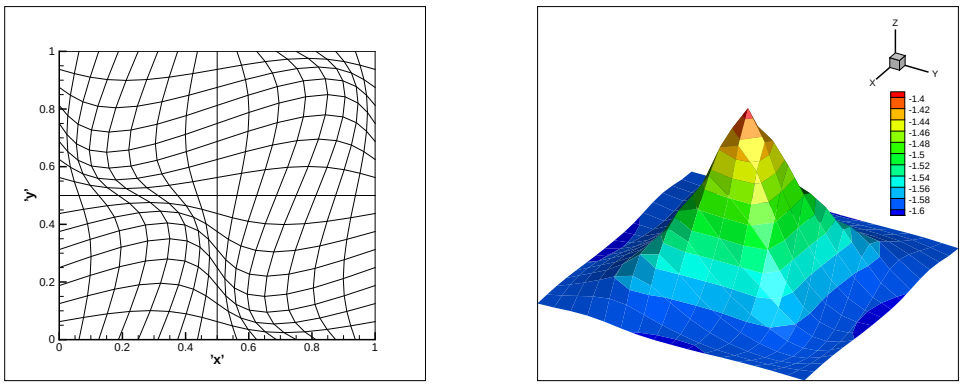


Figure 23: The mesh and contours of ϕ at $t=0.6$ for Example 3.11 on the smoothly moving mesh (3.2), 16×16 grid points.

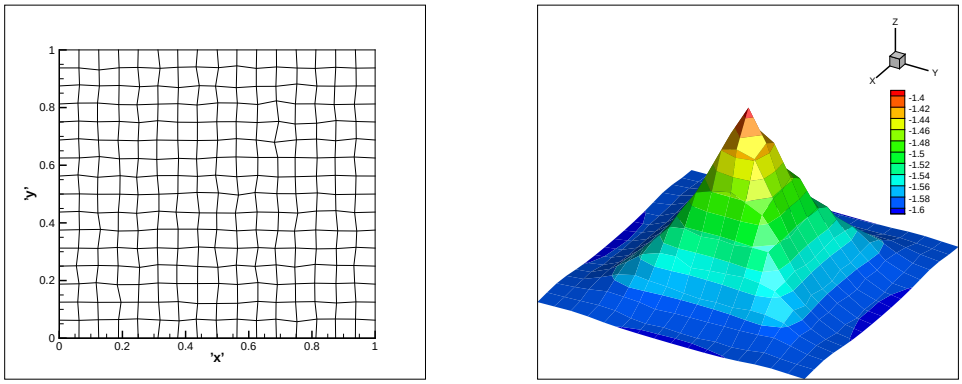


Figure 24: The mesh and contour of ϕ at $t=0.6$ for Example 3.11 on the randomly moving mesh, 16×16 grid points.

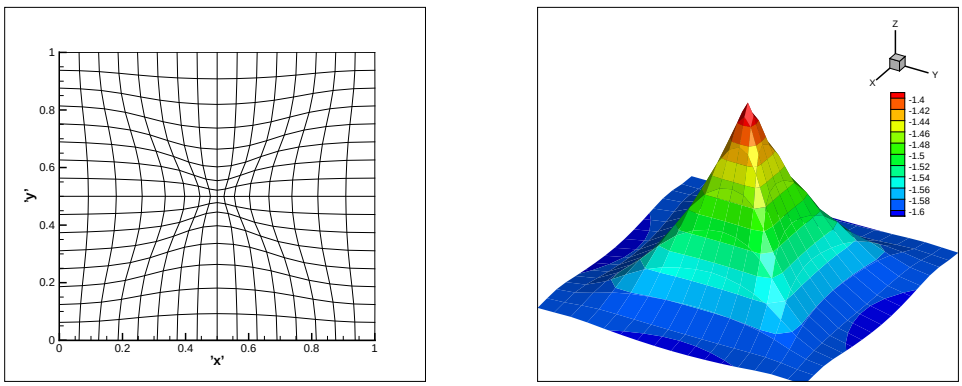


Figure 25: The mesh and contour of ϕ at $t=0.6$ for Example 3.11 on the characteristic-line-type moving meshes with $\omega(x,y)=(\frac{\phi_x}{\sqrt{\phi_x^2+\phi_y^2+1}},\frac{\phi_y}{\sqrt{\phi_x^2+\phi_y^2+1}})$, 16×16 grid points.

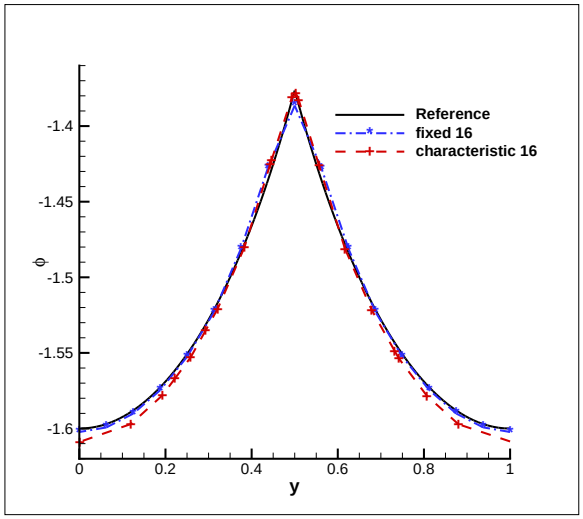


Figure 26: ALE-WENO solution at $t=0.6$ in Example 3.11 cut along $x=0.5$ for the comparison of results from the characteristic-line-type moving grid and the fixed uniform grid against the "exact" reference solution, 16×16 grid points.

the grid points are squeezed toward the center. So we have used the filter in the same way as that in Examples 3.9 and 3.10. We also plot the line cut along $x=0.5$ for the comparison of results on 16×16 grid points, obtained from the characteristic-line-type moving grid and the fixed uniform grid against the "exact" reference solution, in Fig. 26. We observe once again that, in the case of the moving grid along the characteristic lines, the mesh becomes dense near the corner singularities, helping us to capturing these singularities better than the fixed uniform mesh.

Example 3.12. We solve the problem

$$\begin{cases} \phi_t+\frac{1}{4}(\phi_x^2-1)(\phi_x^2-4)=0, \quad (x,y)\in[-2,2]\times[-2,2], \\ \phi(x,y,0)=-2|x|, \end{cases} \tag{3.16}$$

which is a one dimensional problem solved in two dimensions [28]. The Dirichlet and periodic boundary conditions are applied in the x and y directions respectively. We compute the solution until the time $t=1.0$. The results obtained by using our ALE-WENO scheme (2.23) are plotted in Figs. 27-29.

This is a rather demanding test case. Many schemes can not obtain satisfactory results, some of them may even fail to converge to the correct viscosity solution. We plot the solutions along the central cut line $y=0$. When comparing the results from the fixed uniform mesh with that on the moving mesh along the characteristic lines using the same 32×32 grid points, we can see that the one with the characteristic-line-type moving mesh has better resolution. The result of the characteristic-line-type moving mesh with a more refined grid of 256×256 grid points shows clearly the numerical convergence towards the viscosity solution.

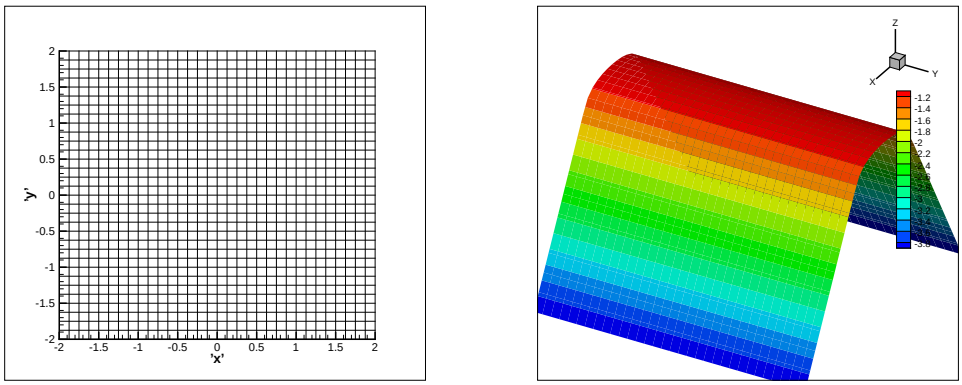


Figure 27: The mesh and contours of ϕ at $t=1$ for Example 3.12 on the fixed uniform mesh, 32×32 grid points.

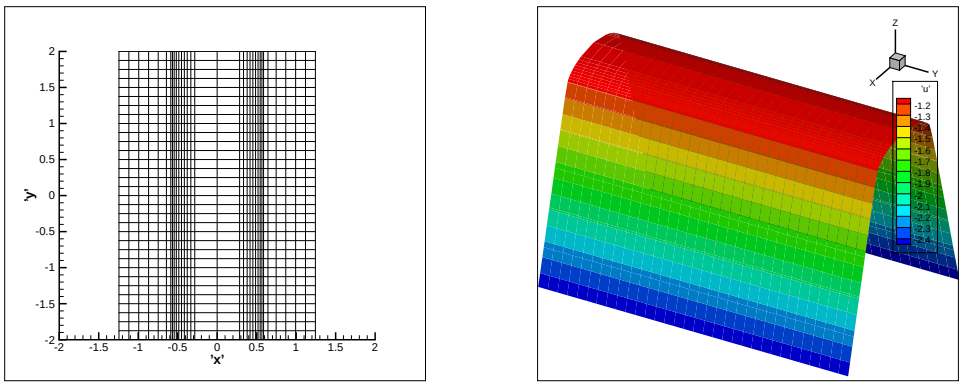


Figure 28: The mesh and contour of ϕ at $t=1$ for Example 3.12 on the characteristic-line-type moving meshes with $\omega(x,y)=(\phi_x^3+\frac{5}{2}\phi_x,0)$, 32×32 grid points.

Example 3.13. We solve the problem from nonlinear solid mechanics

$$\begin{cases} \phi_t+H(x,y,\phi,\phi_x,\phi_y)=0, & (x,y)\in[0,2]^2, \\ \phi(x,y,0)=0, \end{cases} \tag{3.17}$$

with the Hamiltonian

$$H=\phi-\frac{G}{2}[e^y\cos hx+e^{-y}-2]-e^{-y}\tanh\left(\frac{x}{2}\right)\frac{\partial\phi}{\partial x}-[1-e^{-y}]\frac{\partial\phi}{\partial y}+\frac{e^{-y}}{4G\cosh^2(\frac{x}{2})}\left(\frac{\partial\phi}{\partial x}\right)^2,$$

where G denotes the initial shear modulus. This example comes from [15]. Its exact solution is

$$\phi(x,y,t)=2Ge^y\sinh^2\left(\frac{x}{2}\right)\tanh\left(\frac{t}{2}\right)+\frac{G}{2}(e^y-1)\ln(e^{-y}+e^t-e^{t-y}).$$

In this example, we choose $G=1$. Notice that this problem is defined in the whole plane, hence we must truncate the computational domain to be a finite one. In order to obtain

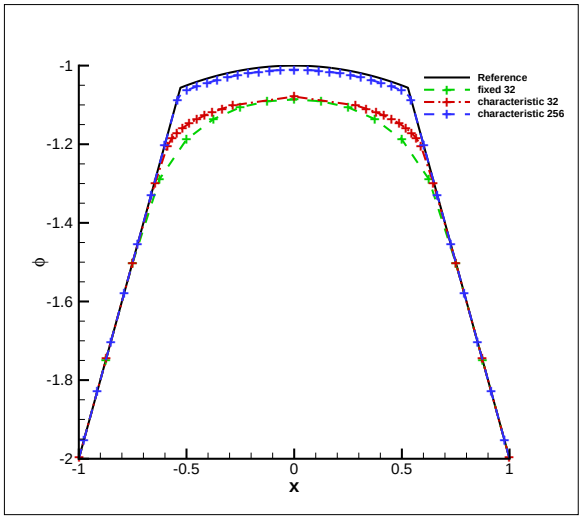


Figure 29: ALE-WENO solution at $t=1$ in Example 3.12 cut along $y=0$. Comparison of results from the fixed uniform mesh and from the characteristic-line-type moving mesh both with 32×32 and 256×256 grid points, against the "exact" reference solution.

boundary conditions for the truncated domain, we use the method of characteristics to solve a system of ordinary differential equations, by the same Runge-Kutta method, to obtain the necessary numerical boundary conditions. This procedure can be used if the singularities of the solution do not reach the boundary of the computational domain (that is, the forward going characteristics along the boundary of the computational domain do not intersect with each other), as is the case for this example. Once the numerical boundary values are obtained in this fashion, we use our ALE-WENO scheme (2.23) to solve the partial differential equations inside the computational domain. For this example, we will use two types of grid motions. The first one is initially a uniform mesh, for which the boundary points move by characteristics and the internal mesh points are just obtained by straight-line connections of the corresponding boundary points. This is still a moving mesh since the boundary of the computational domain is moving along characteristics. The other type is the characteristic-line-type moving mesh, for which in some sense the grid points inside the computational domain have compatible movements as the boundary points. During the characteristic-line-type mesh movement, when the time step Δt determined by the stability condition is larger than h, we set it to h. The solution is computed up to the time $t=0.8$, and we list the errors and numerical orders of accuracy simulating by the ALE-WENO scheme (2.23) in Tables 37-38. We can see that we have obtained the designed third order accuracy, and the errors for the characteristic type mesh movement are smaller in magnitudes than those with uniform meshes for the same number of grid points. We also plot the meshes and the solutions corresponding to both types of mesh movements in Figs. 30-31.

Table 37: Errors and numerical orders of accuracy at $t=0.8$ for Example 3.13 on the moving "uniform" grid with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	1.51E-05		2.57E-05		1.48E-04	
64	1.12E-06	3.74	1.91E-06	3.75	1.31E-05	3.49
128	1.02E-07	3.46	1.67E-07	3.51	1.34E-06	3.29
256	1.08E-08	3.24	1.71E-08	3.29	1.51E-07	3.14
512	1.33E-09	3.02	2.13E-09	3.00	1.81E-08	3.07

Table 38: Errors and numerical orders of accuracy at $t=0.8$ for Example 3.13 on the characteristic-line-type moving grid with $\omega(x,y)=(-e^{-y}\tanh\left(\frac{x}{2}\right)+\frac{e^{-y}}{2G\cosh^2(\frac{x}{2})}\phi_x,-(1-e^{-y}))$ and with $N\times N$ grid points.

N	L_1 error	order	L_2 error	order	L_∞ error	order
32	2.33E-06		3.13E-06		1.01E-05	
64	3.58E-07	2.71	4.88E-07	2.68	1.73E-06	2.56
128	4.98E-08	2.85	6.84E-08	2.83	2.58E-07	2.74
256	6.57E-09	2.92	9.06E-09	2.92	3.54E-08	2.87
512	8.44E-10	2.96	1.16E-09	2.96	4.61E-09	2.94

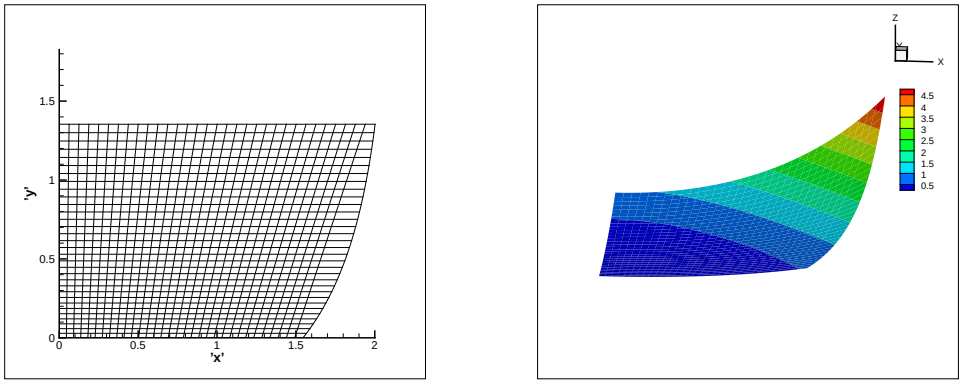


Figure 30: The mesh and contours of ϕ at $t=0.8$ for Example 3.13 on the moving "uniform" meshes, 32×32 grid points.

4 Concluding remarks

In this paper, we have developed and tested a new type of high order multi-resolution WENO scheme in the ALE framework on moving meshes to solve the Hamilton-Jacobi equations. Our algorithm can achieve high order accuracy in the smooth regions and can avoid spurious oscillations near the corner singularities under very mild requirement on the mesh movement function, just boundedness and Lipschitz continuity will suffice. We have tested three types of grid motions, namely smoothly moving meshes, randomly

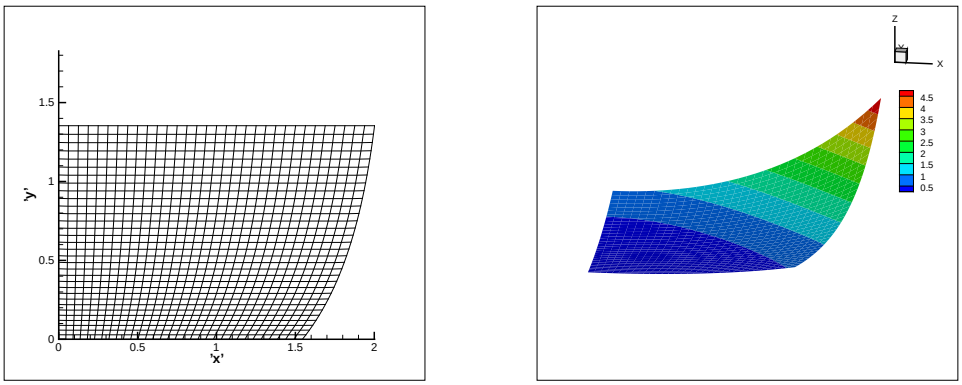


Figure 31: The mesh and contours of ϕ at $t=0.8$ for Example 3.13 on the characteristic-line-type moving meshes with $\omega(x,y)=(-e^{-y}\tanh\left(\frac{x}{2}\right)+\frac{e^{-y}}{2G\cosh^2(\frac{x}{2})}\phi_x,-(1-e^{-y}))$, 32×32 grid points.

moving meshes, and meshes moving with characteristic velocities, to verify the robustness and accuracy of our algorithm. Through ample numerical examples both in one and two dimensions, we have shown that our algorithm on the moving mesh is stable and high order accurate. We can also observe that the results obtained by this algorithm on the grid moving along the characteristic line directions yield better results than those on the fixed grid with the same number of mesh points, which demonstrates the efficiency of our high order scheme on the moving mesh if the mesh movement is chosen appropriately. In the future, we will investigate similar type of multi-resolution WENO finite volume schemes in the ALE framework to solve hyperbolic conservation laws including compressible Euler equations, and eventually we will develop a combined ALE-WENO solver to simulate multi-material flows.

Acknowledgments

The research of J. Cheng is supported in part by NSFC grants 11871111 and U1630247. The research of Y. Xia is supported in part by NSFC grant 11871449. The research of C.-W. Shu is supported in part by NSF grant DMS-1719410.

References

[1] R. Abgrall, Numerical discretization of the first-order Hamilton-Jacobi equation on triangular meshes, Commun. Pure Appl. Math., 49 (1996) 1339-1373.

[2] R. Borges, M. Carmona, B. Costa and W.S. Don, An improved weighted essentially non-oscillatory scheme for hyperbolic conservation laws, J. Comput. Phys., 227 (2008) 3191-3211.

[3] W. Boscheri and M. Dumbser, Arbitrary-Lagrangian-Eulerian discontinuous Galerkin schemes with a posteriori subcell finite volume limiting on moving unstructured meshes, J. Comput. Phys., 346 (2017) 449-479.

[4] S. Bryson and D. Levy, High-order semi-discrete central-upwind schemes for multi-dimensional Hamilton-Jacobi equations, J. Comput. Phys., 189 (2003) 63-87.
[5] R. Calderer and A. Masud, A multiscale stabilized ALE formulation for impressible flows with moving boundaries, Comput. Mech., 46 (2010) 185-197.
[6] J. Cheng and C.-W. Shu, A high order accurate conservative remapping method on staggered meshes, Appl. Numer. Math., 58 (2008) 1042-1060.
[7] M.G. Crandall and P.-L. Lions, Viscosity solutions of Hamilton-Jacobi equations, Trans. Amer. Math. Soc., 277 (1983) 1-42.
[8] M.G. Crandall and P.-L. Lions, Two approximations of solutions of Hamilton-Jacobi equations, Math. Comp., 43 (1984) 1-19.
[9] M.Dumbser, W. Boscheri, M. Semplice and G. Russo, Central WENO schemes for hyperbolic conservation laws on fixed and moving unstructured meshes, SIAM J. Sci. Comput., 39 (2017) 2564-2591.
[10] S. Gottlieb, C.-W. Shu and E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM Rev., 43 (2001) 89-112.
[11] C. W. Hirt, A. A. Amsden and J. L. Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, J. Comput. Phys., 14 (1974) 227-253.
[12] C. Hu and C.-W. Shu, A discontinuous Galerkin finite element method for Hamilton-Jacobi equations, SIAM J. Sci. Comput., 21 (1999) 666-690.
[13] G.S. Jiang and D. Peng, Weighted ENO schemes for Hamilton-Jacobi equations, SIAM J. Sci. Comput., 21 (2000) 2126-2143.
[14] C. Klingenberg, G. Schnucke and Y. Xia, An arbitrary Lagrangian-Eulerian local discontinuous Galerkin method for Hamilton-Jacobi equations, J. Sci. Comput., 73 (2017) 906-942.
[15] V. Lefevre, A. Garnica and O. Lopez-Pamies, A WENO finite-difference scheme for a new class of Hamilton-Jacobi equations in nonlinear solid mechanics, Comput. Methods Appl. Mech. Eng., 349 (2019) 17-44.
[16] D. Levy, S. Nayak, C.-W. Shu and Y.-T. Zhang, Central WENO schemes for Hamilton-Jacobi equations on triangular meshes, SIAM J. Sci. Comput., 28 (2006) 2229-2247.
[17] F. Li and C.-W. Shu, Reinterpretation and simplified implementation of a discontinuous Galerkin method for Hamilton-Jacobi equations, Appl. Math. Lett., 18 (2005) 1204-1209.
[18] J.A. Mackenzie and A. Nicola, A discontinuous Galerkin moving mesh method for Hamilton-Jacobi equations, SIAM J. Sci. Comput., 29 (2007) 2258-2282.
[19] V.-T. Nguyen, An arbitrary Lagrangian-Eulerian discontinuous Galerkin method for simulations of flows over variable geometries, J. Fluids Struct., 26 (2010) 312-329.
[20] S. Osher and R. Fedkiw, Level Sets and Dynamic Implicit Surfaces, Springer-Verlag, New York, 2002.
[21] S. Osher and J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulation, J. Comput. Phys., 79 (1988) 12-49.
[22] S. Osher and C.-W. Shu, High-order essentially non-oscillatory schemes for Hamilton-Jacobi equations, SIAM J. Numer. Anal., 28 (1991) 907-922.
[23] J. Qiu and C.-W. Shu, Hermite WENO schemes for Hamilton-Jacobi equations, J. Comput. Phys., 204 (2005) 82-99.
[24] J.A. Sethian, Curvature and the evolution of fronts, Commun. Math. Phys., 101 (1985) 487-499.
[25] C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes, J. Comput. Phys., 77 (1988) 439-471.
[26] H. Tang, T. Tang and P. Zhang, An adaptive mesh redistribution method for nonlinear

Hamilton-Jacobi equations in two- and three-dimensions, J. Comput. Phys., 188 (2003) 543-572.

[27] X. Yang, W. Huang and J. Qiu, A moving mesh WENO method for one-dimensional conservation laws, SIAM J. Sci. Comput., 34 (2012) 2317-2343.

[28] Y.-T. Zhang and C.-W. Shu, High order WENO schemes for Hamilton-Jacobi equations on triangular meshes, SIAM J. Sci. Comput., 24 (2003) 1005-1030.

[29] F. Zheng, C.-W. Shu and J. Qiu, High order finite difference Hermite WENO schemes for the Hamilton-Jacobi equations on unstructured meshes, Comput. Fluids, 183 (2019) 53-65.

[30] J. Zhu and J. Qiu, A new type of high-order WENO scheme for Hamilton-Jacobi equations on triangular meshes, J. Comput. Phys., 204 (2005) 82-99.

[31] J. Zhu and C.-W. Shu, A new type of multi-resolution WENO schemes with increasingly higher order of accuracy, J. Comput. Phys., 375 (2018) 659-683.