A high order positivity-preserving polynomial projection remapping method

Nuo Lei[1], Juan Cheng[2] and Chi-Wang Shu[3]

Abstract

In this paper, we develop a high-order positivity-preserving polynomial projection remapping method based on the L^2 projection for the discontinuous Galerkin (DG) scheme. Combined with the Lagrangian type DG scheme and the rezoning strategies, we present an indirect arbitrary Lagrangian-Eulerian discontinuous Galerkin (ALE-DG) method. By clipping precisely the intersections between the old distorted mesh and the new rezoned mesh, our remapping method is high-order accurate and has no limitation for the mesh movements, so it is suitable for the large deformable problems. A multi-resolution weighted essentially non-oscillatory (WENO) limiter is adopted to overcome numerical oscillations and it can keep the original high-order accuracy in the smooth region. This WENO limiter combines several different degrees of polynomials which are the local L^2 projections of the original polynomial with nonlinear weights calculated by their smoothness, therefore, it is highly parallel efficient. A positivity-preserving limiter is also added for the physical variables in computational fluid dynamics without losing the original high-order accuracy and conservation. The properties of positivity-preserving, non-oscillation and high-order accuracy of the remapping method will be shown by a variety of numerical experiments on one, two and three dimensional unstructured meshes. The performance of the ALE-DG scheme with rezoning and remapping is also tested for the Euler system in one and two dimensions.

Keywords: Polynomial projection remapping; Discontinuous Galerkin method; High-order; Positivity-preserving; Multi-resolution WENO limiter Arbitrary Lagrangian-Eulerian (ALE).

[1]Graduate School, China Academy of Engineering Physics, Beijing 100088, China. E-mail: leinuo19@gscaep.ac.cn.

[2]Corresponding author. Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, Beijing 100088, China and HEDPS, Center for Applied Physics and Technology, and College of Engineering, Peking University, Beijing 100871, China. E-mail: cheng_juan@iapcm.ac.cn. Research is supported in part by NSFC grants 12031001 and 11871111, and CAEP Foundation No. CX20200026.

[3]Division of Applied Mathematics, Brown University, Providence, RI 02912. E-mail: chi-wang_shu@brown.edu. Research is supported in part by AFOSR grant FA9550-20-1-0055 and NSF grant DMS-2010107.

1 Introduction

There are two major techniques in computational fluid dynamics: the Lagrangian framework with the mesh moving with the fluid velocity, and the Eulerian framework with a fixed mesh, which can easily extend to higher-order accuracy. However, the moving mesh may be twisted in the Lagrangian framework, making the schemes unstable, whereas the Eulerian framework requires a finer mesh for higher resolution near shocks and especially near contact discontinuities. The arbitrary Lagrangian-Eulerian (ALE) framework incorporates the advantages of both frameworks above, and the indirect ALE framework builds in the following three steps.

1. The Lagrangian step: solving the hydrodynamic equations and moving the mesh vertices with the fluid motion;

2. The rezoning step: adjusting the mesh for better mesh quality;

3. The remapping step: transferring numerical information between the two meshes.

The computational mesh in the arbitrary Lagrangian-Eulerian method can move with the fluid as in the Lagrangian method. However, when the distorted mesh causes numerical instability, the rezoning and remapping steps are used to continue the calculation in another mesh with better mesh quality. The ALE framework is widely applied to computational fluid dynamics, based on the finite volume (FV) method [8, 6, 5, 2, 13], or the Runge-Kutta discontinuous Galerkin (RK-DG) method [9, 10, 7].

In the ALE framework based on the FV method, the remapping procedure transfers cell averages from the old mesh to the new mesh, and this procedure should be high-order accurate, essentially non-oscillatory, positive and conservative, as, for example, what we have done in [13, 14]. Based on the discontinuous function space, the DG method [3, 4] is widely adopted for solving hyperbolic conservation laws, since it is flexible for complex mesh geometries and unstructured meshes. In the meantime, this method is highly parallel efficient,

because the elements only communicate with their immediate neighbors. Differently from the ALE-FV method, the remapping procedure in the ALE-DG method is more complicated, since it needs to transfer high-order polynomials to another set of high-order polynomials defined on the new rezoned mesh, while maintaining the good performances. Up to now, most of the remapping methods are applied for the finite volume method, and there have been fewer discussions on remapping methods for the DG method. In this paper, we focus on the remapping step in the ALE framework coupled with the discontinuous Galerkin method.

As we know, one needs to rezone the computational mesh when the mesh is distorted in the ALE framework, and there are many studies about the rezoning strategies, such as the reference Jacobian rezoning [11], high-order linear mesh relaxation [1], or using the Voronoi tessellation method [18]. In [25], the author developed an adaptive mesh topology optimization technique, to improve mesh quality with mesh refinements, edge collapse operation and mesh regularization. We will not focus on this aspect in our study, and hence we may use different rezoning strategies for each of our numerical test problems.

The remapping algorithm has two popular approaches: the fluxed-based method and the intersection-based method. By describing the information exchanges between the old and new mesh cells as a transport equation [5, 19, 12, 17], the flux-based remapping method is faster and easier to apply in the ALE framework. But this method demands that the connection and the number of cells should not change, and the mesh motions should not be too drastic. The solution-updating algorithm in the moving mesh method with the finite element approach also solves a transport equation to convert finite element solution between moving meshes [15, 16], which can be regarded as a flux-based method.

The intersection-based remapping approach is more flexible and rigorous since it picks out exactly the intersections between the old distorted mesh and the new rezoned mesh and calculates the contributions of the old cells to the new cells [6, 2, 20, 13, 14]. This algorithm does not require the connectivity on the new mesh to be the same as the old mesh, and it has no restriction on the movements of the mesh vertices, both of which limit the fluxed-based

remapping algorithm. Meanwhile, the clipping error is close to machine zero and can be ignored. Zhang [27] developed a conservative intersection-based remapping method based on L^2 projection for the one-dimensional moving mesh method. But the applications for the two or three dimensional cases remain to be seen. It is fairly straightforward to detect the overlaps between two intervals in one dimension, but finding the intersection between two triangular or tetrahedral cells in two or three dimensions is costly, especially when compared to the flux-based remapping technique. However, in the Lagrangian type DG method based on the unstructured mesh, the rezoning step will change the mesh connectivity if a large deformable mesh appears, and only the intersection-based remapping method can handle this.

To extend the area of our algorithm's application for the large deformable problems on unstructured mesh, we prefer the intersection-based remapping approach and it can be described as follows. Assume that we have two sets of meshes, and the numerical solution is a piecewise high order polynomial defined on the old mesh. This piecewise polynomial must be transferred to the new mesh, and the new piecewise polynomial must also have the same high order accuracy.

Besides that, we would like to develop an indirect ALE-DG scheme, by combining the moving mesh DG scheme introduced in [9, 10, 7], via a Lagrangian type mesh movement, with the rezoning step and our polynomial projection remapping approach. We will use this Lagrangian type DG scheme to solve the fluid dynamics first, since this scheme can capture contact discontinuities and flow interfaces automatically and sharply with low numerical dissipation. When the computational mesh undergoes distortion or large deformation, which leads to numerical instability or extremely small time step, we will try to rezone the mesh and then use our intersection-based remapping algorithm to make the scheme more stable.

When dealing with high gradient or discontinuous solutions, the high-order piecewise polynomial in the discontinuous Galerkin method may be oscillatory, which should be avoided during the computation. The multi-resolution weighted essentially non-oscillatory (WENO)

technique, which establishes smoothness indicators for a series of polynomials of different degrees and assigns different nonlinear weights to them, is a popular solution. In the smooth region, the higher-order polynomials have more weights so that the new modified polynomial can maintain high-order accuracy. On the other hand, in the non-smooth region, the lower order polynomials play a larger role in the new modified polynomial, making it essentially non-oscillatory. Recently, based on a sequence of local L^2 projection polynomials in the troubled cell, Zhu, Qiu and Shu [28, 29] proposed a new multi-resolution WENO limiter for the discontinuous Galerkin method. In comparison with the traditional WENO limiters, this new WENO limiter is more flexible with any positive linear weights as long as they sum up to 1, and can be easily extended to higher order schemes. This limiter is especially suitable for the moving mesh methods. Besides, since it mainly uses information in the troubled cell itself, with information from immediate neighboring cells used only for the smoothness indicator of the lowest degree polynomial, this limiter is efficient and can be executed in parallel mode. Therefore, we will use this new multi-resolution WENO limiter for the new polynomials, to obtain modified polynomials which are essentially non-oscillatory and highly accurate.

Since the ALE-DG framework is usually applied for fluid flow problems, the physical quantities involved should preserve their physical properties, such as being conservative and positive (non-negative). Our polynomial projection remapping method is automatically conservative which will be explained in the next section. But it is not easy to maintain high-order accuracy when one needs to preserve also the positivity. Zhang and Shu proposed a widely used positivity-preserving framework [26], see also [24], which is based on the positivity of cell averages and includes a simple positivity-preserving scaling limiter, for finite volume and discontinuous Galerkin schemes. By compressing the high order polynomial towards its positive cell average, this limiter makes the negative minimum of the polynomial in the target domain greater than 0, without destroying its original high order accuracy. This positivity-preserving technique has been successfully used for high order conservative remap-

ping method in two and three dimensions [13, 14], and a variety of numerical experiments have confirmed its high efficiency, so we will introduce it into our remapping procedure.

In this paper, we develop a high order polynomial projection remapping method with the local multi-resolution WENO limiter and the positivity-preserving limiter for the ALE-DG framework. In Section 2, we describe our remapping algorithm step by step. In Section 3, we design a series of numerical experiments in one, two and three dimensions to verify the excellent properties of our remapping algorithm, such as high order accuracy, essentially non-oscillatory performance, and positivity-preserving. Afterwards, in conjunction with the Lagrangian type DG scheme, we use this new ALE-DG scheme to solve some benchmarks of the one and two dimensional Euler system and compare with the same order Eulerian DG scheme and the Lagrangian type DG scheme in Section 4. Finally, concluding remarks are given in Section 5.

2 The polynomial projection remapping algorithm

2.1 Basic concepts

Let us start with the one-dimensional polynomial projection remapping algorithm. In the discontinuous Galerkin method, the numerical solution $u_h \in \mathcal{V}^m$ is a piecewise polynomial

$$\mathcal{V}^m = \{w(x) : w(x)|_{I_i} \in \mathcal{P}^m,\ 1 \le i \le N\},$$

where $I_i = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}],\ i = 1, \cdots, N$ are cells of the computational domain $\Omega := \bigcup_{i=1}^{N} I_i$. In each cell I_i, u_h can be written as

$$u_h|_{I_i} = \sum_{l=0}^{m} u_l^i \varphi_l(x),$$

where $u_l^i \in \mathbb{R}$ are coefficients and $\varphi_l(x)$ are basis functions, e.g.

$$\varphi_0(x) = 1, \quad \varphi_1(x) = x, \quad \varphi_2(x) = x^2, \quad \varphi_3(x) = x^3, \quad \cdots$$

In the arbitrary Lagrangian-Eulerian discontinuous Galerkin method, sometimes we need to modify the computational mesh after the Lagrangian step to maintain good mesh quality.

Now, assume that we have a new rezoned mesh $\Omega := \bigcup_{i=1}^{\tilde{N}} \tilde{I}_i$ which satisfies

$$|I_i| > 0, \quad |\tilde{I}_i| > 0, \quad \left|I_i \bigcap I_j\right| = 0, \quad \left|\tilde{I}_i \bigcap \tilde{I}_j\right| = 0, \quad \forall i \neq j,$$

where $|\cdot|$ means the size of the cell. Then we need to convert the numerical solution u_h based on the old mesh $\{I_i\}_{i=1}^N$ to the new rezoned mesh $\{\tilde{I}_i\}_{i=1}^{\tilde{N}}$ with the new discrete polynomial space

$$\tilde{\mathcal{V}}^m = \{w(x) : w(x)|_{\tilde{I}_i} \in \mathcal{P}^m,\ 1 \leq i \leq \tilde{N}\}.$$

That means we need to find a new piecewise polynomial $\tilde{u}_h \in \tilde{\mathcal{V}}^m$, which is the L^2 projection of u_h on $\tilde{\mathcal{V}}^m$

$$(\tilde{u}_h, w_h)_{\tilde{I}_j} = (u_h, w_h)_{\tilde{I}_j}, \quad \forall w_h \in \tilde{\mathcal{V}}^m, \tag{2.1}$$

where $(u, v)_{\tilde{I}_j} := \int_{\tilde{I}_j} u(x)v(x)dx$. It is not easy to compute $(u_h, w_h)_{\tilde{I}_j}$, since u_h is a piecewise polynomial defined on $\{I_i\}_{i=1}^N$ and one needs to calculate the intersection between the new cell $\tilde{I}_j$ and the old mesh $\{I_i\}_{i=1}^N$,

$$(u_h, w_h)_{\tilde{I}_j} = \sum_{i=1}^{N} (u_h, w_h)_{I_i \bigcap \tilde{I}_j} = \sum_{i=1}^{N} \sum_{l=0}^{m} u_l^i (\varphi_l(x), w_h)_{I_i \bigcap \tilde{I}_j}.$$

In practice, we take w_h from the basis functions $\varphi_s(x),\ s = 0, 1, \cdots, m$, so we can rewrite (2.1) as

$$(\tilde{u}_h, \varphi_s(x))_{\tilde{I}_j} = \sum_{i=1}^{N} \sum_{l=0}^{m} u_l^i (\varphi_l(x), \varphi_s(x))_{I_i \bigcap \tilde{I}_j}, \quad s = 0, 1, \cdots, m. \tag{2.2}$$

Notice that, if we take $\varphi_0(x) = 1$ in (2.2), then we have

$$\begin{aligned} \int_{\tilde{I}_j} \tilde{u}_h(x)dx &= \sum_{i=1}^{N} \int_{I_i \bigcap \tilde{I}_j} u_h(x)dx \\ &= \int_{\tilde{I}_j} u_h(x)dx \end{aligned} \tag{2.3}$$

which means our remapping algorithm is conservative.

Assume $\tilde{u}_h|_{\tilde{I}_j} = \sum_{l=0}^m \tilde{u}_l^j \varphi_l(x)$. The new coefficients $\tilde{u}_l^j$ satisfy

$$\sum_{l=0}^{m} \tilde{u}_l^j (\varphi_l(x), \varphi_s(x))_{\tilde{I}_j} = \sum_{i=1}^{N} \sum_{l=0}^{m} u_l^i (\varphi_l(x), \varphi_s(x))_{I_i \bigcap \tilde{I}_j}, \quad s = 0, 1, \cdots m$$

which can be written as

$$\mathbf{M}^j \tilde{\mathbf{u}}^j = \mathbf{b}^j \tag{2.4}$$

where $\mathbf{M}^j = (M^j_{sl})^m_{s,l=0}$ is the mass matrix with $M^j_{sl} = (\varphi_l(x), \varphi_s(x))_{\tilde{I}_j}$ and $\tilde{\mathbf{u}}^j = (\tilde{u}^j_0, \cdots, \tilde{u}^j_m)^T$ are the coefficients which need to be determined. The right-hand side of (2.4) is defined as

$$\mathbf{b}^j = (b^j_0, \cdots, b^j_m)^T, \quad b^j_s = \sum_{i=1}^{N} \sum_{l=0}^{m} u^i_l \left(\varphi_l(x), \varphi_s(x)\right)_{I_i \bigcap \tilde{I}_j}, \quad s = 0, 1, \cdots, m.$$

So, the idea of the polynomial projection remapping algorithm is finding a new piecewise polynomial $\tilde{u}_h \in \tilde{\mathcal{V}}_m$ on the new rezoned mesh $\{\tilde{I}_i\}_{i=1}^{\tilde{N}}$ by solving the linear system (2.4). But the new high-order piecewise polynomial may generate overshoots and the minimum for the physically non-negative variables may less than 0, both of them should be avoided. After solving (2.4), we add a multi-resolution WENO limiter on the new polynomial to prevent numerical oscillations, especially when we design the high-order remapping procedure, and we add a positivity-preserving limiter to maintain positivity for the relevant physical quantities, such as density and internal energy. The above limiters should not destroy the original accuracy and the flowchart of the remapping algorithm is below:

1. Clipping: finding the intersection of $I_i \bigcap \tilde{I}_j, \ \forall\, i, j$;

2. Numerical integration: calculate the integration of the basis functions over the intersections $(\varphi_l(x), \varphi_s(x))_{I_i \bigcap \tilde{I}_j}$, then obtain the new polynomials $\tilde{u}_h$ by solving (2.4);

3. Multi-resolution WENO limiter: modify the high-order polynomials by the multi-resolution WENO limiter, in the so-called troubled cells;

4. Positivity-preserving limiter: modify the high-order polynomials by the positivity-preserving limiter.

Next, we will introduce our polynomial projection remapping algorithm in detail.

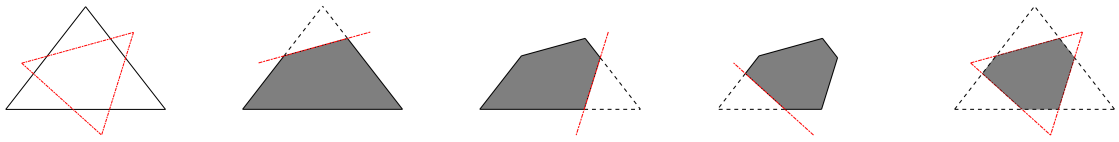
2.2 Clipping

Although the intersection-based remapping method needs to determine the intersections, it is much more flexible for large deformable problems on unstructured meshes and easier to achieve high-order accuracy since the clipping error can be ignored. Therefore, we prefer the intersection-based remapping method for the ALE-DG scheme on the unstructured mesh.

The clipping procedure for two one-dimensional intervals I_i, $\tilde{I}_j$ is very simple, so we concentrate on the multi-dimensional clipping procedure for the triangular cells and the tetrahedral cells.

In this paper, we use the Sutherland-Hodgman polygon clipping algorithm [13, 23] for the two-dimensional triangular cells. In this clipping algorithm, one needs to use the 'window' cell to clip against the 'target' cell. By setting visible and invisible sides for each edge in the 'window' cell, one can separate the 'target' cell with the 'window' cell's edges and pick the intersection parts in the visible sides. Repeating the above loop for each edge in the 'window' cell completes the algorithm. Figure 2.1 gives an example for the clipping algorithm with two triangular cells.

The clipping algorithm for the tetrahedral cells is similar to the 2D clipping algorithm, by setting visible and invisible faces and clipping the 'target' cell using each face of the 'window' cell in turn. The clipping error is close to machine zero and has no involvement with the mesh size, that helps our intersection-based remapping algorithm to achieve high-order precision.



(a) (b) (c) (d) (e)

Figure 2.1: The clipping procedure. The black triangle is the target cell, the red triangle is the window cell, the gray shaded polygon is the clipping result.

2.3 Numerical integration

Now, we need to calculate the integration for the product of two basis functions over the intersections $I_i \bigcap \tilde{I}_j$. For the one-dimensional case, the intersection is also an interval and it is easy to calculate the integration with suitable high-order quadrature rules by mapping the intersection interval to the reference unit $[-1, 1]$.

For the two-dimensional case, the intersection can be any polygon, therefore we will split it into several triangles and use high-order quadrature rules to calculate the numerical integration over these triangles. We locate the barycenter of the convex polygon and connect it to the vertices to divide the intersection into multiple smaller triangles. If the polygon is non-convex, one can also cut it into several triangles by connecting the vertices. The idea of the numerical integration step for the three-dimensional case is the same. By splitting complicated polyhedral cells into several small tetrahedral cells, one can do such integration over these shapes with suitable high-order quadrature rules.

In this paper, we adopt unstructured triangular meshes for the two-dimensional tests and unstructured tetrahedral meshes for the three-dimensional tests. The reference cell in one dimension is the interval $[-1, 1]$ and the two-dimensional triangular reference cell is made up with vertices $(0, 0)$, $(1, 0)$, $(0, 1)$. The three-dimensional tetrahedral reference cell is made up with vertices $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$. In Table 2.1, Table 2.2, Table 2.3 and Figure 2.2, we show the quadrature points and the weights on the reference cells for the third-order remapping schemes in one, two and three dimensions, respectively. These quadrature rules hold exactly for the product of two quadratic polynomials.

Consider the triangular cell $I = \{(x_1^I,\ y_1^I), (x_2^I,\ y_2^I), (x_3^I,\ y_3^I)\}$ and define matrix A_I as:

$$A_I := \begin{pmatrix} x_2^I - x_1^I & x_3^I - x_1^I \\ y_2^I - y_1^I & y_3^I - y_1^I \end{pmatrix},$$

then we can map the reference cell I_0 to the physical cell I as

$$\begin{pmatrix} x \\ y \end{pmatrix} := A_I \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} + \begin{pmatrix} x_1^I \\ y_1^I \end{pmatrix}, \quad (x, y) \in I, \quad \forall (\xi_1, \xi_2) \in I_0.$$

The integration over I can be written as:

$$\begin{aligned}(\varphi_l(x,y),\varphi_s(x,y))_I &= \int_I \varphi_l(x,y)\varphi_s(x,y)dxdy \\ &= \int_{I_0} \varphi_l(x(\xi_1,\xi_2),y(\xi_1,\xi_2))\varphi_s(x(\xi_1,\xi_2),y(\xi_1,\xi_2))2|I|d\xi_1 d\xi_2 \\ &= |I|\sum_{\alpha=1}^{L}\omega_\alpha\varphi_l(x_\alpha,y_\alpha)\varphi_s(x_\alpha,y_\alpha)\end{aligned} \tag{2.5}$$

where $\binom{x_\alpha}{y_\alpha} = A_I\binom{\xi_{1,\alpha}}{\xi_{2,\alpha}} + \binom{x_1^I}{y_1^I}$. ω_α and $(\xi_{1,\alpha},\xi_{2,\alpha})$ are the weights and the quadrature points in the reference cell I_0, respectively, which have been listed in Table 2.2. The integration over the tetrahedral cell is the same, so we omit it here.

Up to now, we can use the numerical integration method to calculate the mass matrix $\mathbf{M}^j$ and the right hand side vector $\mathbf{b}^j$ in (2.4). Then, we obtain our new piecewise polynomial $\tilde{u}_h$ by solving the linear system (2.4).

Table 2.1: Quadrature rule for the one-dimensional remapping scheme.

ω_α	$\xi_{1,\alpha}$
0.5688888888888889	0.0000000000000000
0.4786286704993665	-0.5384693101056831
0.4786286704993665	0.5384693101056831
0.2369268850561891	-0.9061798459386640
0.2369268850561891	0.9061798459386640

Table 2.2: Quadrature rule for the two-dimensional remapping scheme.

ω_α	$\xi_{1,\alpha}$	$\xi_{2,\alpha}$
0.205950504760887	0.124949503233232	0.437525248383384
0.205950504760887	0.437525248383384	0.124949503233232
0.205950504760887	0.437525248383384	0.437525248383384
0.063691414286223	0.797112651860071	0.165409927389841
0.063691414286223	0.797112651860071	0.037477420750088
0.063691414286223	0.165409927389841	0.797112651860071
0.063691414286223	0.165409927389841	0.037477420750088
0.063691414286223	0.037477420750088	0.797112651860071
0.063691414286223	0.037477420750088	0.165409927389841

Table 2.3: Quadrature rule for the three-dimensional remapping scheme.

ω_α	$\xi_{1,\alpha}$	$\xi_{2,\alpha}$	$\xi_{3,\alpha}$
0.0190476190476190	0.0000000000000000	0.5000000000000000	0.5000000000000000
0.0190476190476190	0.5000000000000000	0.0000000000000000	0.5000000000000000
0.0190476190476190	0.5000000000000000	0.5000000000000000	0.0000000000000000
0.0190476190476190	0.5000000000000000	0.0000000000000000	0.0000000000000000
0.0190476190476190	0.0000000000000000	0.5000000000000000	0.0000000000000000
0.0190476190476190	0.0000000000000000	0.0000000000000000	0.5000000000000000
0.0885898247429807	0.6984197043243866	0.1005267652252045	0.1005267652252045
0.0885898247429807	0.1005267652252045	0.1005267652252045	0.1005267652252045
0.0885898247429807	0.1005267652252045	0.1005267652252045	0.6984197043243866
0.0885898247429807	0.1005267652252045	0.6984197043243866	0.1005267652252045
0.1328387466855907	0.0568813795204234	0.3143728734931922	0.3143728734931922
0.1328387466855907	0.3143728734931922	0.3143728734931922	0.3143728734931922
0.1328387466855907	0.3143728734931922	0.3143728734931922	0.0568813795204234
0.1328387466855907	0.3143728734931922	0.0568813795204234	0.3143728734931922

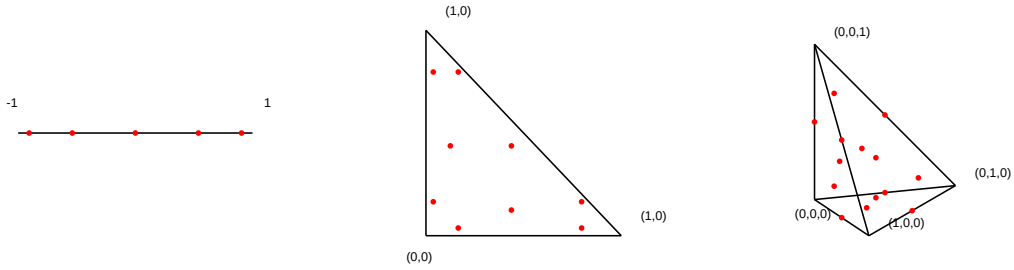


Figure 2.2: The quadrature points.

2.4 Multi-resolution WENO limiter

Near the discontinuity or in the large gradient regions, high-order polynomials may generate overshoots or oscillations which could make the scheme unstable, therefore we should pay more attention on how to prevent the numerical oscillations effectively. Differently from our previous remapping work [13, 14], where we reconstructed high-order polynomials based on the cell averages with the multi-resolution WENO procedure which can overcome the numerical oscillation, the new polynomials $\tilde{u}_h$ generated by this remapping algorithm may be oscillatory, since they are solved by the linear system (2.4), corresponding to the standard

L^2 projection without any limitation. To overcome this problem, we adopt a new type of multi-resolution WENO limiter [28, 29] for the DG scheme, which can retain high-order accuracy in the smooth region and can achieve essentially non-oscillatory performance in the discontinuous region.

In this part, we will follow the multi-resolution WENO procedure proposed in [30, 28] step by step to generate an essentially non-oscillatory polynomial. In order to save space, we take the one-dimensional third-order multi-resolution WENO limiter as an example to discuss the specific procedure and we refer to [30, 28] for higher-order schemes or multi-dimensional cases.

Suppose we need to modify a third-order polynomial $\tilde{u}_{\tilde{I}_j}(x) = \sum_{l=0}^{2} \tilde{u}_l^j \varphi_l(x)$, for all $j = 1, \cdots, \tilde{N}$.

- Step 1. Define low-order polynomials $q_0(x), q_1(x)$ on local lower-order polynomial spaces with the L^2 projection method

$$(q_s(x), \varphi_l(x))_{\tilde{I}_j} = \left(\tilde{u}_{\tilde{I}_j}(x), \varphi_l(x)\right)_{\tilde{I}_j}, \quad l = 0, 1, \cdots, s \tag{2.6}$$

  where $q_s(x) \in \mathcal{P}^s$ for $s = 0, 1$. Notice that, $q_s(x)$ is defined on the basis $q_s(x) = \sum_{k=0}^{s} q_k^s \varphi_k(x)$, so the integration in (2.6) can be written as

$$\sum_{k=0}^{s} q_k^s \left(\varphi_k(x), \varphi_l(x)\right)_{\tilde{I}_j} = \sum_{k=0}^{m} \tilde{u}_k^j \left(\varphi_k(x), \varphi_l(x)\right)_{\tilde{I}_j}, \quad l = 0, 1, \cdots, s \tag{2.7}$$

  which is convenient to solve. Besides that, we define the third-order polynomial as $q_2(x) := \tilde{u}_{\tilde{I}_j}(x)$.

- Step 2. Introduce linear weights and define

$$\begin{aligned} p_0(x) &= q_0(x), \\ p_1(x) &= \frac{1}{\gamma_{1,1}} q_1(x) - \frac{\gamma_{0,1}}{\gamma_{1,1}} p_0(x), \\ p_2(x) &= \frac{1}{\gamma_{2,2}} q_2(x) - \frac{\gamma_{0,2}}{\gamma_{2,2}} p_0(x) - \frac{\gamma_{1,2}}{\gamma_{2,2}} p_1(x), \end{aligned}$$

where $\gamma_{0,1} = \frac{1}{11},\ \gamma_{1,1} = \frac{10}{11}$ and $\gamma_{0,2} = \frac{1}{111},\ \gamma_{1,2} = \frac{10}{111},\ \gamma_{2,2} = \frac{100}{111}$. By combining $p_s(x)$ with the linear weights $\gamma_{s,2}$, we can achieve the optimal third-order accuracy

$$\sum_{s=0}^{2} \gamma_{s,2} p_s(x) = q_2(x) = \tilde{u}_{\tilde{I}_j}(x).$$

- Step 3. Define smoothness indicators to measure how smooth these functions are in the cell $\tilde{I}_j$,

$$\beta_1 := \int_{\tilde{I}_j} |\tilde{I}_j| \left(\frac{\partial p_1(x)}{\partial x} \right)^2 dx, \quad \beta_2 := \sum_{\alpha=1}^{2} \int_{\tilde{I}_j} |\tilde{I}_j|^{2\alpha-1} \left(\frac{\partial^\alpha p_2(x)}{\partial x^\alpha} \right)^2 dx,$$

and the smoothness indicator of the zero-order polynomial $p_0(x)$ should be defined in another way. Define

$$\zeta_0 = (\bar{\tilde{u}}_{\tilde{I}_j} - \bar{\tilde{u}}_{\tilde{I}_{j-1}})^2, \quad \zeta_1 = (\bar{\tilde{u}}_{\tilde{I}_{j+1}} - \bar{\tilde{u}}_{\tilde{I}_j})^2, \quad \bar{\eta}_0 = \begin{cases} 1 & \zeta_0 \geq \zeta_1 \\ 10 & \text{else} \end{cases}, \quad \bar{\eta}_1 = 11 - \bar{\eta}_0,$$

and

$$\eta_0 = \frac{\bar{\eta}_0}{\bar{\eta}_0 + \bar{\eta}_1},\ \eta_1 = 1 - \eta_0, \quad \sigma_0 = \eta_0 \left(1 + \frac{|\zeta_0 - \zeta_1|}{\zeta_0 + \varepsilon_0} \right),\ \sigma_1 = \eta_1 \left(1 + \frac{|\zeta_0 - \zeta_1|}{\zeta_1 + \varepsilon_0} \right),$$

where we take $\varepsilon_0 = 10^{-10}$. Then we have

$$\beta_0 := \frac{1}{\sigma^2} \left(\sigma_0 (\bar{\tilde{u}}_{\tilde{I}_j} - \bar{\tilde{u}}_{\tilde{I}_{j-1}}) + \sigma_1 (\bar{\tilde{u}}_{\tilde{I}_{j+1}} - \bar{\tilde{u}}_{\tilde{I}_j}) \right)^2, \quad \sigma = \sigma_0 + \sigma_1,$$

where $\bar{\tilde{u}}_{\tilde{I}_j}$ is the cell average on the cell $\tilde{I}_j$.

- Step 4. Define the nonlinear weights as

$$\omega_l = \frac{\bar{\omega}_l}{\bar{\omega}_0 + \bar{\omega}_1 + \bar{\omega}_2}, \quad \bar{\omega}_l = \gamma_{l,2} \left(1 + \frac{\tau}{\varepsilon_0 + \beta_l} \right), \quad l = 0, 1, 2$$

where $\tau = |\beta_2 - \beta_0| + |\beta_2 - \beta_1|$ and the final polynomial is given by

$$\tilde{u}^W_{\tilde{I}_j}(x) := \sum_{s=0}^{2} \omega_s p_s(x).$$

However, it is not necessary to apply this multi-resolution WENO limiter for every cell. Instead, we identify the troubled cells by a shock detection technique [21, 28, 29, 31], and only apply the WENO limiter on those cells.

To summarize, we simply employ the multi-resolution WENO limiter only on a few troubled cells to improve the efficiency of our method. Actually, in the following numerical tests, we do not always utilize this limitation because some of these tests do not need it. This limiter combines the polynomial's local L^2 projection on lower discrete polynomial spaces with nonlinear weights given by the polynomials' smoothness, and this limiter is parallel efficient, since there is little information exchange between neighbors (only for the smoothness indicator of the lowest degree polynomial). The nonlinear weights are close to the linear weights in the smooth region, and the modified polynomial $\tilde{u}^W_{\tilde{I}_j}(x)$ is close to the original high-order polynomial $\tilde{u}_{\tilde{I}_j}(x)$. In the discontinuous region, on the other hand, the low-order polynomials play a larger role in $\tilde{u}^W_{\tilde{I}_j}(x)$, which can prevent numerical oscillations.

2.5 Positivity-preserving limiter

Associated with fluid flow problems, the involved physical quantities in the ALE framework such as density and internal energy should preserve positivity. This requires our polynomial projection remapping procedure to maintain this property as well.

Assume that the input piecewise polynomial satisfies $u_h(x) > \varepsilon$ everywhere, or at least at the quadrature points in each $I_i \bigcap \tilde{I}_j$, and we obtain the new piecewise polynomial $\tilde{u}_h$ by solving the linear system (2.4) with or without the multi-resolution WENO limiter, where ε is a small positive number and we take $\varepsilon = 10^{-14}$. Then the new cell averages $\bar{\tilde{u}}_{\tilde{I}_j}$ are also greater than ε, since

$$\begin{aligned} |\tilde{I}_j|\bar{\tilde{u}}_{\tilde{I}_j} &= \left(\tilde{u}_{\tilde{I}_j}, 1\right)_{\tilde{I}_j} \\ &= (u_h, 1)_{\tilde{I}_j} \\ &= \sum_{i=1}^{N} \int_{I_i \bigcap \tilde{I}_j} u_h|_{I_i} dx \geq 0. \end{aligned}$$

Besides that, we need to preserve positivity for the new piecewise polynomial $\tilde{u}_h$, otherwise, the numerical solutions in the next Lagrangian DG step may be negative, which are contradicted with the laws of physics.

Referring to the work of Zhang and Shu [26], we compress the polynomial $\tilde{u}_{\tilde{I}_j}$ in each cell

$\tilde{I}_j$ towards its non-negative cell average $\bar{\tilde{u}}_{\tilde{I}_j}$ as,

$$\begin{aligned} \tilde{u}^P_{\tilde{I}_j}(x) = &\quad \theta\tilde{u}_{\tilde{I}_j}(x) + (1-\theta)\bar{\tilde{u}}_{\tilde{I}_j}, \\ \theta = &\quad \min\left\{1, \frac{|\bar{\tilde{u}}_{\tilde{I}_j} - \varepsilon|}{|\bar{\tilde{u}}_{\tilde{I}_j} - m_{\tilde{I}_j}|}\right\}, \quad m_{\tilde{I}_j} = \min_{x\in S_j} \tilde{u}_{\tilde{I}_j}(x), \quad \varepsilon = 10^{-14}. \end{aligned} \tag{2.8}$$

So the new polynomials are non-negative $\tilde{u}^P_{\tilde{I}_j}(x) > 0$ for all $x \in S_j$. If we need to remap $\tilde{u}^P_h$ on $\{\tilde{I}_j\}_{j=1}^{\tilde{N}}$ to a new mesh $\{\tilde{\tilde{I}}_l\}_{l=1}^{\tilde{\tilde{N}}}$ in the next remapping loop, S_j is the set of the quadrature points in each $\tilde{I}_j \bigcap \tilde{\tilde{I}}_l$, $\forall\, l$. If we need to do the Lagrangian DG step in the ALE-DG scheme with $\tilde{u}^P_h$, S_j is the set of the quadrature points in the cell $\tilde{I}_j$ and its boundaries. Notice that S_j is a finite set and we only need to find the minimum of $\tilde{u}_{\tilde{I}_j}(x)$ in S_j rather than the minimum in the entire cell $\tilde{I}_j$, which makes the implementation of this scaling limiter considerably more efficient.

When we solve the Euler system, we first utilize the above positivity-preserving limiter for the density ρ, then we use the momentum, total energy and the modified density to preserve positivity for the internal energy e, as detailed in [24]. This compressing strategy is also suitable for the multi-dimensional case.

Meanwhile, this positivity-preserving limiter is also conservative because it does not affect the cell averages and also maintains the original high-order accuracy as proved in [24, 26]. Notice that, we apply this limiter after the multi-resolution WENO limiter since it will not destroy the essentially non-oscillatory property by compressing the polynomial towards the cell average.

We have now completed the discussion of the polynomial projection remapping algorithm. Although we have only introduced the local multi-resolution WENO limiter and the positivity-preserving limiter for the one-dimensional situation in detail, extending to the multi-dimensional case is not difficult, and details may be found in the references [29, 26]. Following that, we apply our remapping method to several benchmarks in one, two, and three dimensions to validate its good properties such as conservation, high-order accuracy, positivity and essentially non-oscillatory performance. After that, we utilize the novel ALE-DG scheme consisting of this remapping algorithm and the Lagrangian type DG scheme to

solve the Euler system in one and two dimensions.

3 Numerical results

3.1 One-dimensional case

Suppose $\Omega = \{I_i^0\}_{i=1}^N,\ I_i^0 = [x_{i-\frac{1}{2}}^0, x_{i+\frac{1}{2}}^0]$ is the initial mesh and we design a randomly moving mesh strategy $\Omega = \{I_i^t\}_{i=1}^N,\ I_i^t = [x_{i-\frac{1}{2}}^t, x_{i+\frac{1}{2}}^t]$,

$$x_{i-\frac{1}{2}}^t = x_{i-\frac{1}{2}}^0 + c_R h r_{i-\frac{1}{2}}^t, \quad h = \min_i h_i, \quad h_i = x_{i+\frac{1}{2}}^0 - x_{i-\frac{1}{2}}^0, \quad i = 2, \cdots, N, \tag{3.1}$$

where $r_{i-\frac{1}{2}}^t \in [-1, 1]$ are random numbers and we take $c_R = 0.5$ in our numerical tests. After remapping T times on the randomly moving mesh, we require the final mesh to move back to the initial mesh in the accuracy tests.

3.1.1 Accuracy test

Now, we verify the high-order accuracy of our polynomial projection remapping method. Suppose the initial function is

$$u(x) = \cos^8(8\pi x) + 10^{-12}, \quad x \in [0, 1].$$

First, we calculate the initial L^2 projection $u_h^0(x)$ of $u(x)$ on the initial mesh and calculate the error $\|u_h^0 - u\|$. After that, we will remap $T = 10$ times on the randomly moving mesh (3.1) and move back to the initial mesh $\{I_i^0\}_{i=1}^N$. We denote the remapping results as $\tilde{u}_h^{10}$ without any limiters, $\tilde{u}_h^{P,10}$ with the positivity-preserving limiter and $\tilde{u}_h^{W,P,10}$ with the local multi-resolution WENO limiter and the positivity-preserving limiter.

Based on the different degrees of the DG space, we show the third-order and the fourth-order remapping results in Table 3.1 and Table 3.2 as examples, respectively. The last column 'PP' and 'WENO' in these tables represent the ratio of the cells being modified by the positivity-preserving limiter and the local multi-resolution WENO limiter. As one can see, our high-order polynomial projection remapping method achieves the designed high-order accuracy, regardless of the limiters being involved.

Table 3.1: One-dimensional third-order accuracy test: error and order of the polynomial projection remapping method on the randomly moving meshes with $T = 10$.

	$\|\|u_h^0 - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
80	6.3222E-04		1.3081E-03		6.4792E-03		-
160	7.4433E-05	3.09	1.6780E-04	2.96	9.7294E-04	2.74	-
320	9.1500E-06	3.02	2.1112E-05	2.99	1.2489E-04	2.96	-
640	1.1378E-06	3.01	2.6433E-06	3.00	1.5622E-05	3.00	-
	$\|\|\tilde{u}_h^{10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
80	2.0806E-03		4.0362E-03		2.3422E-02		-
160	2.1561E-04	3.27	4.3533E-04	3.21	3.0377E-03	2.95	-
320	2.7779E-05	2.96	6.1088E-05	2.83	5.3377E-04	2.51	-
640	3.2693E-06	3.09	7.1444E-06	3.10	5.7270E-05	3.22	-
	$\|\|\tilde{u}_h^{P,10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	PP(%)
80	2.2009E-03		4.1378E-03		2.3389E-02		6.63
160	2.1557E-04	3.35	4.3538E-04	3.25	3.0377E-03	2.94	3.44
320	2.7781E-05	2.96	6.1088E-05	2.83	5.3377E-04	2.51	2.03
640	3.2693E-06	3.09	7.1444E-06	3.10	5.7270E-05	3.22	1.03
	$\|\|\tilde{u}_h^{W,P,10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	WENO(%)
80	2.2007E-03		4.1374E-03		2.3389E-02		2.25
160	2.1557E-04	3.35	4.3537E-04	3.25	3.0373E-03	2.94	0.38
320	2.7781E-05	2.96	6.1088E-05	2.83	5.3377E-04	2.51	0.00
640	3.2693E-06	3.09	7.1444E-06	3.10	5.7270E-05	3.22	0.00

Table 3.2: One-dimensional fourth-order accuracy test: error and order of the polynomial projection remapping method on the randomly moving meshes with $T = 10$.

	$\|\|u_h^0 - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
80	7.4378E-05		1.5060E-04		6.8521E-04		-
160	4.5190E-06	4.04	9.6719E-06	3.96	5.8090E-05	3.56	-
320	2.8097E-07	4.01	6.0863E-07	3.99	3.8955E-06	3.90	-
640	1.7696E-08	3.99	3.8104E-08	4.00	2.4771E-07	3.98	-
	$\|\|\tilde{u}_h^{10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
80	4.1647E-04		8.0582E-04		4.0711E-03		-
160	2.7564E-05	3.92	5.5238E-05	3.87	3.2861E-04	3.63	-
320	1.6354E-06	4.08	3.4513E-06	4.00	2.8659E-05	3.52	-
640	1.0805E-07	3.92	2.2483E-07	3.94	1.8635E-06	3.94	-
	$\|\|\tilde{u}_h^{P,10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	PP(%)
80	5.6114E-04		9.4540E-04		4.0694E-03		5.88
160	2.8468E-05	4.30	5.5564E-05	4.09	3.2861E-04	3.63	3.06
320	1.6389E-06	4.12	3.4516E-06	4.01	2.8659E-05	3.52	1.53
640	1.0806E-07	3.92	2.2483E-07	3.94	1.8635E-06	3.94	5.78
	$\|\|\tilde{u}_h^{W,P,10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	WENO(%)
80	5.6114E-04		9.4540E-04		4.0694E-03		0.25
160	2.8468E-05	4.30	5.5564E-05	4.09	3.2861E-04	3.63	0.00
320	1.6389E-06	4.12	3.4516E-06	4.01	2.8659E-05	3.52	0.00
640	1.0806E-07	3.92	2.2483E-07	3.94	1.8635E-06	3.94	0.00

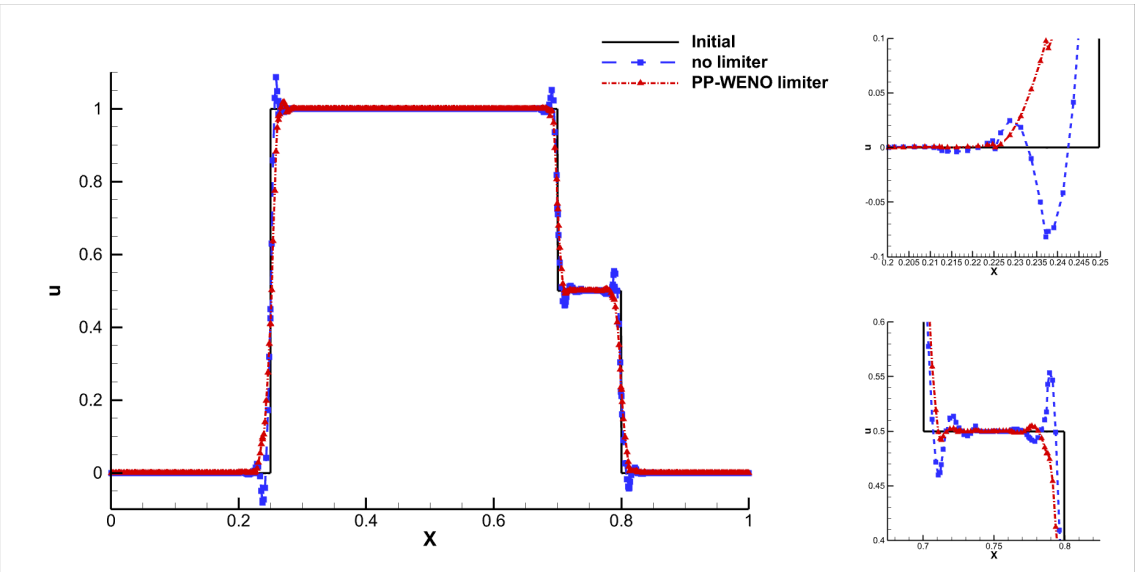


Figure 3.1: One-dimensional discontinuity test: third-order polynomial projection remapping method with 80 cells. The black solid lines are the initial function (3.2), the blue dash lines are the numerical results $\tilde{u}_h^{10}(x)$ without any limiter, the red dash dot lines are the numerical results with the two limiters $\tilde{u}_h^{W,P,10}(x)$. Top right: the zoomed-in subfigure at $x \in [0.2, 0.25]$; bottom right: the zoomed-in subfigure at $x \in [0.675, 0.825]$.

3.1.2 Discontinuity test

In this subsection, we consider a discontinuous function (3.2) to verify our polynomial projection remapping method is positive and essentially non-oscillatory

$$u(x) = \begin{cases} 10^{-12} & x \leq 0.25 \\ 1 & 0.25 < x \leq 0.7 \\ 0.5 & 0.7 < x \leq 0.8 \\ 10^{-12} & 0.8 < x \leq 1 \end{cases}, \quad 0 \leq x \leq 1. \tag{3.2}$$

Just as before, we remap on the randomly moving mesh for $T = 10$ times with $N = 80$ cells.

In Figure 3.1, there are about 2.13% cells which have been modified by the positivity-preserving limiter and about 1.75% cells which have been modified by the local multi-resolution WENO limiter. It is obvious that our positivity-preserving limiter can preserve positivity in the top right subfigure and the multi-resolution WENO limiter can prevent the numerical oscillation in the bottom right subfigure.

3.2 Two-dimensional case

Now, we move to the two-dimensional polynomial projection remapping method on the triangular mesh. Suppose (x_p, y_p) is the coordinate of an interior node of the computational domain Ω, and $h = \min_{I_i \in \Omega} h_i$ is the minimum of the diameter h_i, where h_i is the circumscribed sphere's diameter of the triangle I_i. We design a randomly moving mesh as:

$$(x_p^{t+1}, y_p^{t+1})_R = (x_p^0, y_p^0) + c_R h (r_{x_p}^t, r_{y_p}^t),$$

where $r_{x_p}^t, r_{y_p}^t \in [-1, 1]$ are random numbers and $c_R = 0.5$.

3.2.1 Accuracy test

We use the initial function

$$u(x, y) = \sin^8(2\pi x)\cos^8(2\pi y) + 10^{-12}, \quad -1 \le x, y \le 1,$$

to verify high-order accuracy of our remapping method. We remap $T = 10$ times on the randomly moving mesh and move back to the initial mesh and denote $u_h^0(x, y)$ as the L^2 projection of $u(x, y)$.

The initial mesh divides the computational domain uniformly into small squares with mesh size $h = \frac{2}{N_x}$, where $N_x = N_y$ are number of cells in each directions, then each square will be divided into two triangles with the same area. As one can see in Table 3.3, our polynomial projection remapping method achieves the designed third-order accuracy with the positivity-preserving limiter. Besides the positivity-preserving modification, we also use the local multi-resolution WENO limiter in this accuracy test but there are no cells which have been picked out by the shock detection technique.

3.2.2 Positivity-preserving test

Then, we verify our remapping method can preserve positivity for the cell averages. This time, the computational domain is a circle with radius 1 and center at $(0, 0)$, and the initial

Table 3.3: Two-dimensional third-order accuracy test: error and order of the polynomial projection remapping method on the randomly moving meshes with $T = 10$.

	$\|\|u_h^0 - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
3200	3.4041E-04		8.4090E-04		5.1226E-03		-
7200	1.0441E-04	2.91	2.5425E-04	2.95	1.6435E-03	2.80	-
12800	4.4169E-05	2.99	1.0801E-04	2.98	7.0773E-04	2.93	-
20000	2.2683E-05	2.99	5.5478E-05	2.99	3.7241E-04	2.88	-
	$\|\|\tilde{u}_h^{10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
3200	5.7979E-04		1.3293E-03		9.1929E-03		-
7200	1.8175E-04	2.86	4.2199E-04	2.83	3.1854E-03	2.61	-
12800	7.9977E-05	2.85	1.8771E-04	2.82	1.4200E-03	2.81	-
20000	4.0482E-05	3.05	9.5285E-05	3.04	7.3757E-04	2.94	-
	$\|\|\tilde{u}_h^{P,10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	PP(%)
3200	5.6341E-04		1.3256E-03		9.1929E-03		0.0391
7200	1.8117E-04	2.80	4.2198E-04	2.82	3.1854E-03	2.61	0.0556
12800	7.9899E-05	2.85	1.8771E-04	2.82	1.4200E-03	2.81	0.0820
20000	4.0474E-05	3.05	9.5285E-05	3.04	7.3757E-04	2.94	0.0900

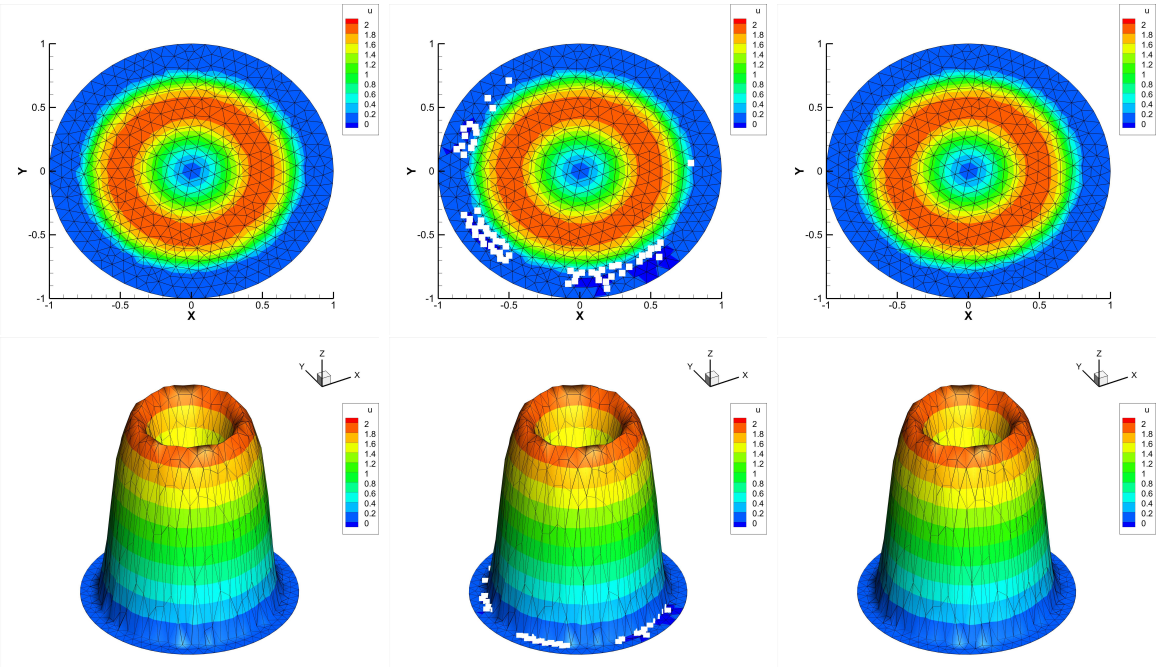


Figure 3.2: Two-dimensional positivity-preserving test: third-order polynomial projection remapping method with 1,016 triangular cells. Left: the initial function; middle: remapping results without any limiter $\tilde{u}_h^{10}(x,y)$; right: remapping results with the two limiters $\tilde{u}_h^{W,P,10}(x,y)$. White symbols represent the cells where the cell-averages are negative.

function is

$$u(x,y) = \begin{cases} 1 + \sin\left(2\pi(r - \frac{1}{4})\right) + 10^{-12} & r \le 0.75 \\ 10^{-12} & r > 0.75 \end{cases}, \quad r = \sqrt{x^2 + y^2},\ r \le 1. \tag{3.3}$$

We move the interior nodes randomly for $T = 10$ times and return to the initial triangular mesh. There are about 5.49% of the cells which have been modified by the positivity-preserving limiter and about 7.94% of the cells which have been modified by the multi-resolution WENO limiter. Near the discontinuity, there are many negative cell averages marked in white in Figure 3.2 without the positivity-preserving modification, but our remapping method preserves positivity well with the limiter.

3.2.3 Discontinuity test

We design a discontinuous initial function

$$u(x,y) = \begin{cases} 10^{-12} & x \le 0,\ y \le 0 \\ f_1(x,y) & x \le 0,\ y > 0 \\ f_2(x,y) & x > 0,\ y > 0 \\ f_3(x,y) & x > 0,\ y \le 0 \end{cases}, \quad -1 \le x, y \le 1, \tag{3.4}$$

where

$$f_1(x,y) = 10^{-12} + 10\max(0, 1 - 2.5R_1), \quad R_1 = \sqrt{(x+\frac{1}{2})^2 + (y-\frac{1}{2})^2}, \tag{3.5}$$

$$f_2(x,y) = \begin{cases} 10 & x > 0.1,\ y > 0.1 \\ 10^{-12} & \text{else} \end{cases}, \tag{3.6}$$

$$f_3(x,y) = \begin{cases} 10 & R_2 < 0.4 \\ 10^{-12} & R_2 \ge 0.4 \end{cases} \quad R_2 = \sqrt{(x-\frac{1}{2})^2 + (y+\frac{1}{2})^2}. \tag{3.7}$$

Remapping on the randomly moving mesh with $T = 10$ times, we show the values at 128 points at the cut line $x = y$ and $x = -y$ in Figure 3.4.

Overall, there are about 1.38% of the cells which have been modified by the positivity-preserving limiter and about 5.32% of the cells which have been modified by the multi-resolution WENO limiter. In Figure 3.3, our remapping results with these two limiters preserve positivity well and the WENO limiter can handle the overshoots near the discontinuity which can be seen in Figure 3.4.

3.3 Three-dimensional case

Consider the three-dimensional polynomial projection remapping method on the tetrahedral meshes. Suppose that (x_p, y_p, z_p) is the coordinate of an interior node of the computational domain Ω, and $h = \min_{I_i \in \Omega} h_i$ is the minimum of the diameter h_i, where h_i is the circumscribed sphere's diameter of the tetrahedral I_i. Here, the randomly moving mesh is defined as before:

$$(x_p^{t+1}, y_p^{t+1}, z_p^{t+1})_R = (x_p^0, y_p^0, z_p^0) + c_R h(r_{x_p}^t, r_{y_p}^t, r_{z_p}^t).$$

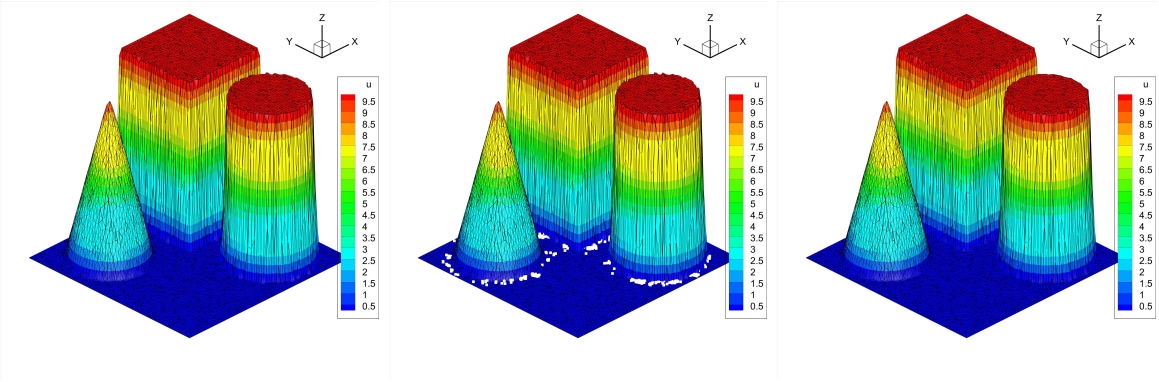
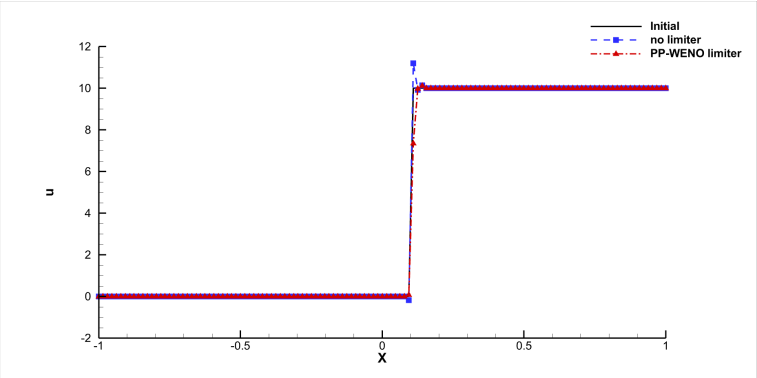
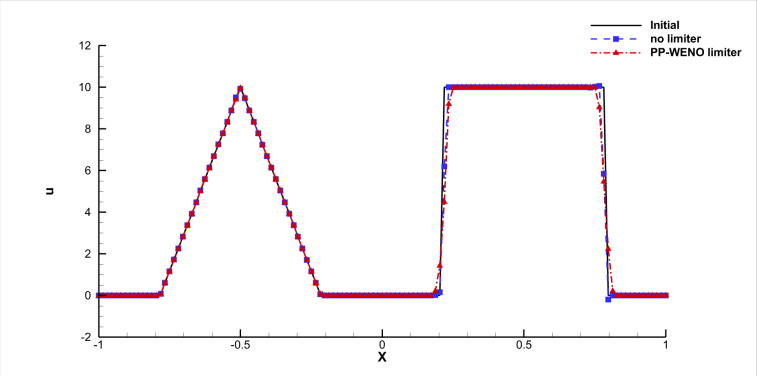


Figure 3.3: Two-dimensional discontinuity test: third-order polynomial projection remapping method with 14,120 triangular cells. Left: the initial function; middle: remapping results without any limiter $\tilde{u}_h^{10}(x,y)$; right: remapping results with two limiters $\tilde{u}_h^{W,P,10}(x,y)$. White symbols represent the cells where the cell-averages are negative.



(a) Cut line $x = y$



(b) Cut line $x = -y$

Figure 3.4: The function values at the cut lines $x = y$ and $x = -y$. The black solid lines: the initial function; the blue dash lines: the remapping results without any limiter $\tilde{u}_h^{10}(x,y)$; the red dash dot lines: the remapping results with two limiters $\tilde{u}_h^{W,P,10}(x,y)$.

3.3.1 Accuracy test

First, we use the following function to verify the high-order accuracy of our remapping method

$$u(x,y,z) = \cos^4(\pi x)\cos^4(\pi y)\cos^4(\pi z) + 10^{-12}, \quad 0 \le x,y,z \le 2.$$

The initial mesh divides the computational domain uniformly into small cubes with mesh size $h = \frac{2}{N_x}$, where $N_x = N_y = N_z$ are number of cells in each directions, then each cube will be divided into six tetrahedrons with the same volume. The remapping results with or without the positivity-preserving limiter are listed in Table 3.4, denoted as $\tilde{u}_h^{10}$ and $\tilde{u}_h^{P,10}$, respectively. Besides the positivity-preserving limiter, we also add the multi-resolution WENO limiter in this accuracy test but there are no cells which have been picked out by the shock detection technique. As one can see, our three-dimensional polynomial projection remapping method achieves the designed third-order accuracy, regardless whether the limiter is involved.

3.3.2 Positivity-preserving test

In this subsection, we verify the 3D remapping method can preserve positivity for the cell averages. We dig a ball of radius 1.4 centered at $(0,0,0)$ in the cube computational domain $[-2,2]\times[-2,2]\times[-2,2]$ and design a positive initial function

$$u(x,y,z) = \begin{cases} 10^{-12} & r \le 1.8 \\ r & r > 1.8 \end{cases}, \quad r = \sqrt{x^2+y^2+z^2}, \ -2 \le x,y,z \le 2. \tag{3.8}$$

Part of the computational domain in $[0,2]\times[0,2]\times[0,2]$ is shown in Figure 3.5. Here, we plot the values on the computational nodes (x_p, y_p, z_p) and we mark the cell which has negative cell average with white color. Near the discontinuity, there are many negative cell averages marked in white without the positivity-preserving limiter, and our remapping method preserves positivity well.

Table 3.4: Three-dimensional third-order accuracy test: error and order of the polynomial projection remapping method on the randomly moving meshes with $T = 10$.

	$\|\|u_h^0 - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
6000	1.2729E-03		2.3126E-03		1.7824E-02		-
20250	4.1717E-04	2.75	7.1604E-04	2.89	5.3933E-03	2.95	-
48000	1.7943E-04	2.93	3.0686E-04	2.95	2.2191E-03	3.09	-
93750	9.1861E-05	3.00	1.5826E-04	2.97	1.2293E-03	2.65	-
	$\|\|\tilde{u}_h^{10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	-
6000	1.7112E-03		2.9751E-03		2.6823E-02		-
20250	5.9499E-04	2.61	9.9972E-04	2.69	9.5736E-03	2.54	-
48000	2.6383E-04	2.83	4.4650E-04	2.80	4.3434E-03	2.75	-
93750	1.3698E-04	2.94	2.3409E-04	2.89	2.2625E-03	2.92	-
	$\|\|\tilde{u}_h^{P,10} - u\|\|$						
N	L^1 error	order	L^2 error	order	L^∞ error	order	PP(%)
6000	2.4407E-03		4.0926E-03		4.8257E-02		0.15
20250	5.8573E-04	3.52	9.9562E-04	3.49	9.5734E-03	3.99	0.04
48000	2.5952E-04	2.83	4.4593E-04	2.79	4.3434E-03	2.75	0.00
93750	1.3511E-04	2.93	2.3376E-04	2.89	2.2625E-03	2.92	0.00

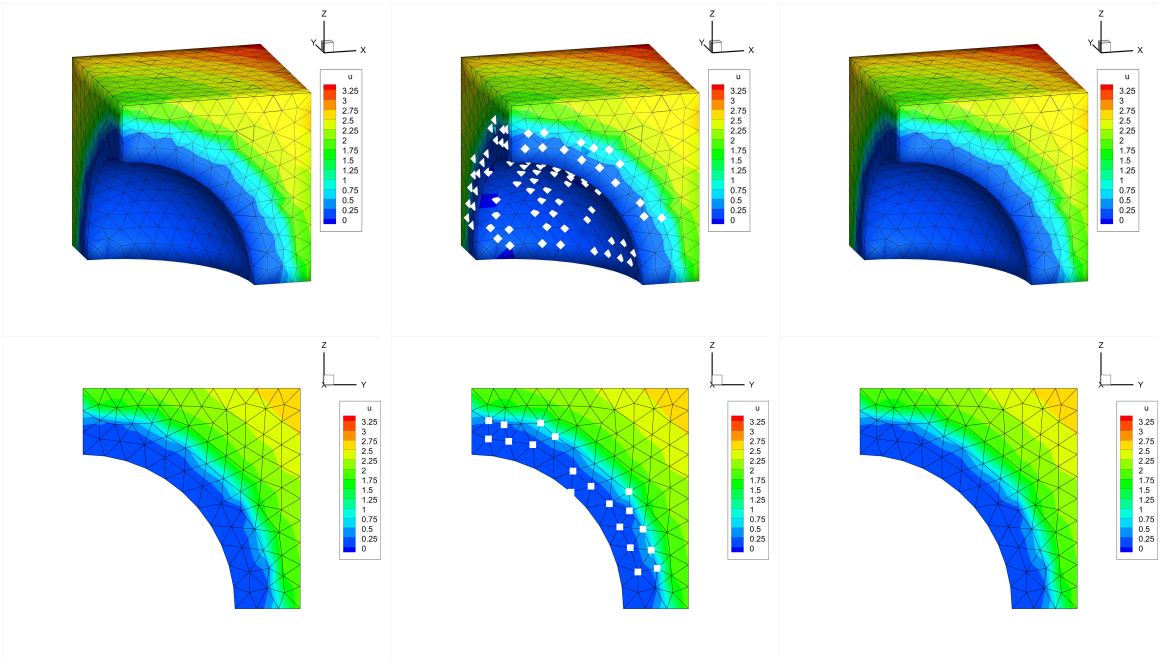


Figure 3.5: Three-dimensional positivity-preserving test: third-order polynomial projection remapping method with 3,938 tetrahedral cells. Left: the initial function; middle: remapping results without any limiter $\tilde{u}_h^{10}(x, y, z)$; right: remapping results with the two limiters $\tilde{u}_h^{W,P,10}(x, y, z)$. The bottom three subfigures are the 2D cut planes at $x = 0$ and white symbols represent the cells where the cell-averages are negative.

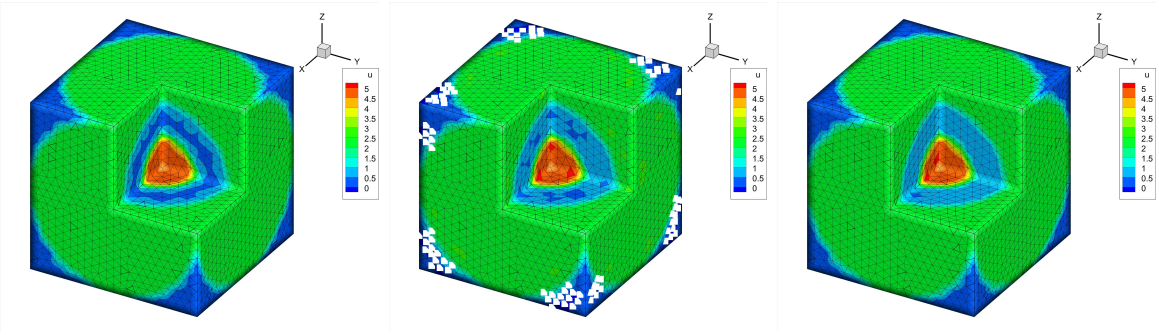


Figure 3.6: Three-dimensional discontinuity test: third-order polynomial projection remapping method with 37,160 tetrahedral cells. Left: the initial function; middle: remapping results without any limiter $\tilde{u}_h^{10}(x,y,z)$; right: remapping results with the two limiters $\tilde{u}_h^{W,P,10}(x,y,z)$. White symbols represent the cells where the cell-averages are negative.

3.3.3 Discontinuity test

In this subsection, we verify our 3D remapping method can handle the numerical oscillation with the limiters. The discontinuous initial function is

$$u(x,y,z) = \begin{cases} 5 & r \le 0.4 \\ 0.5 & 0.4 < r \le 0.8 \\ 2.5 & 0.8 < r \le 1.4 \\ 10^{-12} & r > 1.4 \end{cases}, \quad r = \sqrt{(x-1)^2+(y-1)^2+(z-1)^2}, \ 0 \le x,y,z \le 2. \tag{3.9}$$

There are about 0.60% of the cells which have been modified by the positivity-preserving limiter and about 4.63% of the cells which have been modified by the multi-resolution WENO limiter. In Figure 3.6, we show the whole computational domain without the part $x,y,z > 1$ and we can observe that the remapping results with these two limiters preserve positivity well. In Figure 3.7, we show the values of 40 points at the cut line $x = y = z$, and one can observe that the WENO limiter prevent the numerical oscillation well.

4 Numerical results with the ALE-DG scheme

Klingenberg, Schnucke and Xia [9, 10] and Fu, Schnucke and Xia [7] proposed a discontinuous Galerkin method on the moving mesh for conservation laws and Hamilton-Jacobi equations. The semi-discrete scheme is L^2 stable and achieves optimal accuracy with an upwind flux.

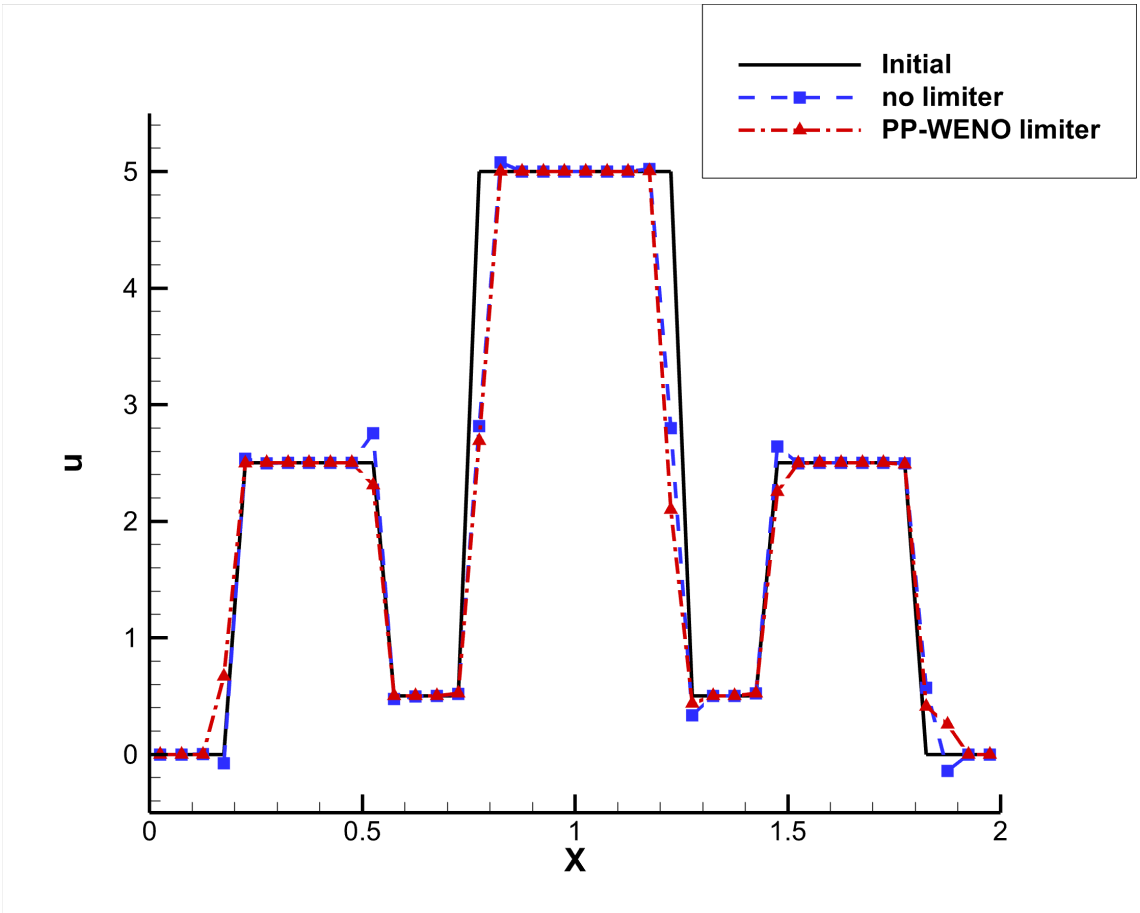


Figure 3.7: The function values at the cut line $x = y = z$. The black solid line: the initial function; the blue dash line: the remapping results without any limiter $\tilde{u}_h^{10}(x, y, z)$; the red dash dot line: the remapping results with the two limiters $\tilde{u}_h^{W,P,10}(x, y, z)$.

Combined with the total-variation-diminishing Runge-Kutta time discretization (TVD-RK) methods [22], the fully-discrete scheme satisfies the geometric conservation law, under the condition that the accuracy of the time discretization is greater than the spatial dimension.

When applied to fluid dynamics, if we take the velocities of the mesh movements as the fluid velocities as in the Lagrangian methods, this DG scheme (which we refer as the Lagrangian type DG scheme) may generate distorted meshes in the presence of large fluid deformations, just as the standard Lagrangian schemes. We therefore use an indirect ALE-DG scheme to overcome this difficulty. This ALE-DG scheme combines the Lagrangian type DG method, suitable rezoning strategy and our high order polynomial projection remapping method. In this section, we will compare the performances of the following three DG schemes, the Eulerian DG scheme on the fixed mesh, the Lagrangian type DG scheme on the moving mesh with the fluid velocities and the indirect ALE-DG scheme. The numerical results on these schemes will be denoted as ρ_E, ρ_L and ρ_A, respectively.

4.1 One-dimensional ALE-DG scheme with the high order polynomial projection remapping method

Here, we will first introduce the one-dimensional ALE-DG scheme briefly and then display our numerical results. Consider the following model problem:

$$\begin{array}{rcl} \partial_t u + \partial_x f(u) & = & 0, \quad (x,t) \in \Omega \times (0,T], \\ u(x,0) & = & u_0(x), \quad x \in \Omega. \end{array} \tag{4.10}$$

In the DG framework [9], we assume that there are given points $\{x^n_{j-\frac{1}{2}}\}_{j=1}^{N+1}$ and mesh velocities $\{\omega^n_{j-\frac{1}{2}}\}_{j=1}^{N+1}$ at the time level t^n. In our Lagrangian type DG scheme, we take the mesh velocity as the fluid velocity in the Euler system. Then we can define the new mesh $\{x^{n+1}_{j-\frac{1}{2}}\}_{j=1}^{N+1}$ at the time level t^{n+1} by

$$x^{n+1}_{j-\frac{1}{2}} := x^n_{j-\frac{1}{2}} + \omega^n_{j-\frac{1}{2}}(t^{n+1} - t^n),$$

which should satisfy $\Omega = \bigcup_{j=1}^{N} I_j^n = \bigcup_{j=1}^{N} I_j^{n+1}$ with $I_j^n = [x_{j-\frac{1}{2}}^n, x_{j+\frac{1}{2}}^n]$. Connect the point $x_{j-\frac{1}{2}}^n$ and $x_{j-\frac{1}{2}}^{n+1}$ by a straight line

$$x_{j-\frac{1}{2}}^n(t) := x_{j-\frac{1}{2}}^n + \omega_{j-\frac{1}{2}}^n (t - t_n), \quad \forall t \in [t_n, t_{n+1}],$$

and assume that all of the points in the cell I_j^n also move in the same way (along the straight line). Then we define the mesh velocity in I_j^n as

$$\omega^n(x,t) := \omega_{j+\frac{1}{2}}^n \frac{x - x_{j-\frac{1}{2}}^n}{\Delta_j^n(t)} + \omega_{j-\frac{1}{2}}^n \frac{x_{j+\frac{1}{2}}^n - x}{\Delta_j^n(t)}, \ x \in K_j^n(t),$$

where

$$K_j^n(t) = [x_{j-\frac{1}{2}}^n(t), x_{j+\frac{1}{2}}^n(t)], \quad \Delta_j^n(t) = x_{j+\frac{1}{2}}^n(t) - x_{j-\frac{1}{2}}^n(t).$$

Suppose $\{\varphi_l\}_{l=0}^m$ are the basis functions on the reference cell $[-1,1]$ and define

$$\hat{\varphi}_l^n(x,t) := \varphi_l \left(\frac{2\left(x - x_{j-\frac{1}{2}}^n\right)}{\Delta_j^n(t)} - 1 \right), \quad x \in K_j^n(t),$$

on the discrete space

$$\mathcal{V}_h(t) := \left\{ v_h \in L^2(\Omega) | v_h(x,t) \in \mathcal{P}^m \right\}.$$

Then, by the integration-by-parts method we obtain the DG scheme on the moving mesh: Find a function $u_h \in \mathcal{V}_h(t)$ such that

$$\begin{aligned} \frac{d}{dt}(u_h, v_h)_{K_j(t)} &= (g(\omega, u_h), \partial_x v_h)_{K_j(t)} \\ &\quad - \hat{g}\left(\omega_{j+\frac{1}{2}}, u_{h,j+\frac{1}{2}}^-, u_{h,j+\frac{1}{2}}^+\right) v_{h,j+\frac{1}{2}}^- + \hat{g}\left(\omega_{j-\frac{1}{2}}, u_{h,j-\frac{1}{2}}^-, u_{h,j-\frac{1}{2}}^+\right) v_{h,j-\frac{1}{2}}^+ \end{aligned} \tag{4.11}$$

for all $v_h = \sum_{l=0}^m v_l \hat{\varphi}_l(x,t) \in \mathcal{V}_h(t)$ and cells. Notice that $g(\omega, u_h) := f(u_h) - \omega u_h$ and $\hat{g}\left(\omega_{j+\frac{1}{2}}, u_{h,j+\frac{1}{2}}^-, u_{h,j+\frac{1}{2}}^+\right)$ is the numerical flux which should satisfy consistency, monotonicity and Lipschitz continuity.

Based on this Lagrangian type DG method and our high order polynomial projection remapping method, we give the flowchart of our indirect ALE-DG scheme. Suppose we know the mesh $\{x_{j-\frac{1}{2}}^n\}_{j=1}^{N+1}$ and piecewise polynomial u_h^n at $t = t^n$,

1. Calculate the mesh velocity $\{\omega_{j-\frac{1}{2}}^n\}_{j=1}^{N+1}$ as the Lagrangian method,

2. Calculate the time step τ which satisfies the CFL condition,

3. Solve the DG scheme (4.11) with the TVD-RK time discretization to get u_h^{n+1} on the new mesh $\{x_{j-\frac{1}{2}}^{n+1}\}_{j=1}^{N+1}$,

4. When the mesh is distorted, rezone the mesh $\{x_{j-\frac{1}{2}}^{n+1}\}_{j=1}^{N+1}$ into $\{\tilde{x}_{j-\frac{1}{2}}^{n+1}\}_{j=1}^{N+1}$,

5. After the rezoning step, remap u_h^{n+1} to the new rezoned mesh and obtain $\tilde{u}_h^{n+1}$,

then $\tilde{u}_h^{n+1}$ on the new rezoned mesh $\{\tilde{x}_{j-\frac{1}{2}}^{n+1}\}_{j=1}^{N+1}$ can enroll in the next loop.

4.1.1 Numerical tests for the one-dimensional Euler equation

We consider the Euler equation of gas dynamics:

$$\begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E+p) \end{pmatrix}_x = 0, \tag{4.12}$$

while $p = (\gamma - 1)(E - \frac{1}{2}\rho u^2)$ for the calorically ideal gas. Here, ρ is the density, u is the fluid velocity, E is the total energy, p is the pressure and γ is a constant that depends on the particular gas under consideration.

The time step satisfies

$$\tau \leq \frac{1}{2m+1} \min_j \frac{h_j}{|\lambda_j - \omega_j|}$$

where m is the order of the piecewise polynomial space and $|\lambda_j|$ is the maximum wave speed.

In the numerical tests, we adopt two sets of meshes,

- The fixed Eulerian mesh

$$x_{j+\frac{1}{2}}(t_n) = x_{j+\frac{1}{2}}(0) \tag{4.13}$$

- The Lagrangian type moving mesh

  Here, we assume that the mesh moves with the fluid as $x'(t) = u$

$$x_{j+\frac{1}{2}}(t_{n+1}) = x_{j+\frac{1}{2}}(t_n) + \tau \mu_{h,j+\frac{1}{2}}^n \tag{4.14}$$

where μ is the Roe average of the velocity u

$$\mu^n_{h,j+\frac{1}{2}} = \left(\frac{\sqrt{\rho^-}u^- + \sqrt{\rho^+}u^+}{\sqrt{\rho^-} + \sqrt{\rho^+}} \right)_{h,j+\frac{1}{2}},$$

and $\rho^{\pm}_{h,j+\frac{1}{2}}$ and $u^{\pm}_{h,j+\frac{1}{2}}$ are the left and right values of $\rho_h,\ u_h$ on the cell boundaries $x_{j+\frac{1}{2}}$, respectively.

Accuracy test. The initial condition is given as

$$(\rho_0, u_0, p_0) = \left(\frac{1 + 0.2\sin(x)}{2\sqrt{3}}, \sqrt{\gamma}\rho_0, \rho_0^{\gamma} \right), \quad x \in [0, 2\pi].$$

Suppose that $\rho(x,t),\ u(x,t),\ p(x,t)$ are the exact solutions, and if we take $\gamma = 3$, then we can verify that $2\sqrt{3}\rho(x,t)$ is the exact solution of the Burgers' equation:

$$v_t + \left(\frac{v^2}{2} \right)_x = 0, \quad v(x,0) = 1 + 0.2\sin(x)$$

and

$$u(x,t) = \sqrt{\gamma}\rho(x,t), \quad p(x,t) = \rho(x,t)^{\gamma}.$$

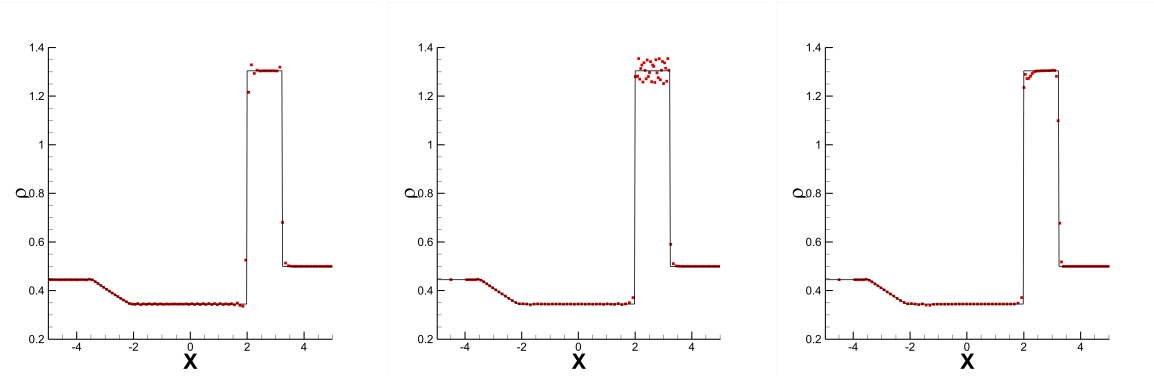
We demonstrate the numerical results for density at $t = 0.3$ calculated by the Eulerian type DG scheme ρ_E, the Lagrangian type DG scheme ρ_L and the ALE-DG scheme ρ_A. In the meantime, we show the projection error between the initial condition $\rho(x,0)$ and its projection ρ^0. We rezone the old mesh to the new uniform mesh and apply our remapping procedure in the ALE-DG scheme every 10 time steps. Table 4.5 shows the error on different sizes of the mesh $N = 32, 64, 128, 256$ and we can observe that the error for the Lagrangian type DG scheme $||\rho(x,T) - \rho_L||$ is a little smaller than the error for the Eulerian DG scheme $||\rho(x,T) - \rho_E||$, and there is almost no difference after applying the remapping procedure.

The Lax problem. Now, we consider the Lax problem for the Euler system with the initial condition

$$\begin{cases} (\rho, u, p) = (0.445, 0.698, 3.528), & x \in [-5, 0] \\ (\rho, u, p) = (0.5, 10^{-10}, 0.571), & x \in (0, 5] \end{cases} \quad \gamma = 1.4. \tag{4.15}$$

Table 4.5: Error at time $t = 0.3$ for the one-dimensional Euler system with three third-order DG schemes.

	initial projection error $\|\|\rho(x,0) - \rho^0\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
32	1.3369E-06		1.7768E-06		3.6271E-06	
64	1.6684E-07	3.00	2.2220E-07	3.00	4.5481E-07	3.00
128	2.0842E-08	3.00	2.7778E-08	3.00	5.6909E-08	3.00
256	2.6058E-09	3.00	3.4722E-09	3.00	7.1298E-09	3.00
	$\|\|\rho(x,T) - \rho_E\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
32	2.8338E-06		4.4243E-06		1.7132E-05	
64	3.5842E-07	2.98	5.5769E-07	2.99	2.2268E-06	2.94
128	4.5526E-08	2.98	7.0768E-08	2.98	2.8311E-07	2.98
256	5.7121E-09	2.99	8.8737E-09	3.00	3.5532E-08	2.99
	$\|\|\rho(x,T) - \rho_L\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
32	2.6026E-06		4.0057E-06		1.5937E-05	
64	3.2157E-07	3.02	4.8801E-07	3.04	1.8884E-06	3.08
128	4.0057E-08	3.00	6.0117E-08	3.02	2.1532E-07	3.13
256	5.2076E-09	2.94	7.7262E-09	2.96	2.4073E-08	3.16
	$\|\|\rho(x,T) - \rho_A\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
32	2.6026E-06		4.0057E-06		1.5937E-05	
64	3.2157E-07	3.02	4.8801E-07	3.04	1.8884E-06	3.08
128	4.0057E-08	3.00	6.0117E-08	3.02	2.1532E-07	3.13
256	5.2076E-09	2.94	7.7262E-09	2.96	2.4073E-08	3.16



(a) Eulerian DG, ρ_E (b) Lagrangian type DG, ρ_L (c) ALE-DG, ρ_A

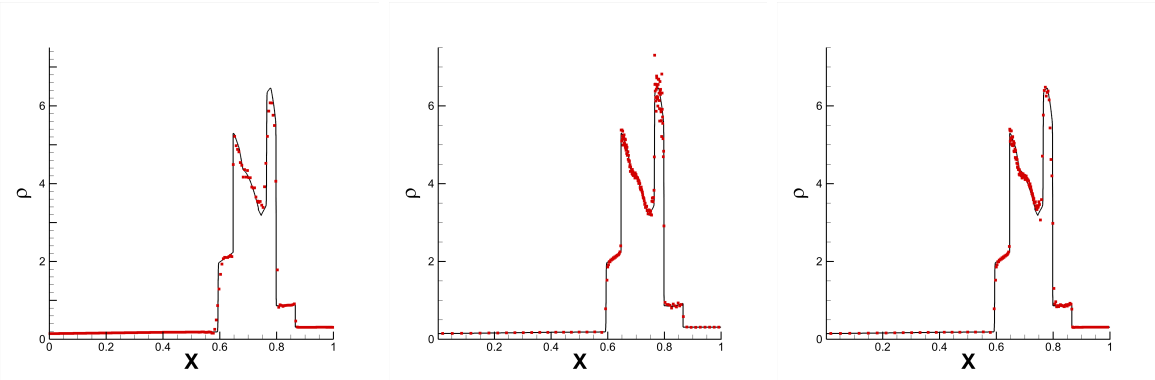
Figure 4.8: Comparison of the exact solution (black solid line) and three third-order DG solutions (red points) for the Lax problem with $N = 100$ at time $t = 1.3$.

We show the numerical results at time $t = 1.3$ with $N = 100$ cells in Figure 4.8. Notice that, all of the DG schemes need a positivity-preserving limiter [26]. We can observe from the middle subfigure of Figure 4.8 that there are almost no points at the contact discontinuity produced by the Lagrangian type DG scheme, which is the advantage of the Lagrangian method, but numerical oscillations appear near the contact discontinuity.

To control these overshoots and keep high resolution on the contact discontinuity, we utilize the local multi-resolution WENO limiter for the Lagrangian DG step and the remapping step in the ALE-DG scheme. Here, we perform the remapping and the rezoning step without moving the points at the front of the shock and the contact discontinuity, every 20 time steps when $t > 1.0$. The numerical solution ρ_A in the right subfigure of Figure 4.8 shows that the ALE-DG scheme makes a balance between the low numerical oscillations and the low numerical dissipation.

The blast wave problem. In this part, we consider the blast wave problem for the Euler system with

$$\rho = 1, \quad u = 1, \quad p = \begin{cases} 1000, & x \in [0, 0.1) \\ 0.01, & x \in [0.1, 0.9) \\ 100, & x \in [0.9, 1] \end{cases} \quad \gamma = 1.4, \quad x \in [0, 1]. \tag{4.16}$$



(a) Eulerian DG, ρ_E (b) Lagrangian type DG, ρ_L (c) ALE-DG, ρ_A

Figure 4.9: Comparison of the reference solution (black solid line) and three third-order DG solutions (red points) for the blast wave problem with $N = 200$ at time $t = 0.038$.

In Figure 4.9, we show the numerical results of the density for the above three DG schemes at time $t = 0.038$ with $N = 200$ cells. The black solid line is the numerical solution from [2] with 16,000 cells, which can be regarded as the reference solution.

This Lagrangian type DG scheme captures the contact discontinuity and the shock rigorously, but there are some overshoots near the contact discontinuity, see the middle subfigure of Figure 4.9. Besides that, we apply our remapping procedure and the rezoning method without moving the points at the front of the shock and contact discontinuities, every 50 time steps after $t > 0.03$ and the results of the ALE-DG scheme are displayed in the right of Figure 4.9, which can handle the overshoots very well and maintain the good performance of the Lagrangian type DG scheme that there are less transition points on the contact discontinuity.

4.2 Two-dimensional ALE-DG scheme with the high order polynomial projection remapping method

Let us apply our remapping method for the two-dimensional Lagrangian type DG scheme [7] to solve the fluid flow problems. We still begin with the brief description of the DG scheme for the following model problem:

$$\begin{aligned} \partial_t u + \partial_x f(u) + \partial_y g(u) &= 0, \quad (x, y, t) \in \Omega \times (0, T], \\ u(x, y, 0) &= u_0(x, y), \quad (x, y) \in \Omega. \end{aligned} \tag{4.17}$$

Assume the computational domain is divided into several triangles $\Omega = \bigcup_{i=1}^{N} I_i^n$ at the time level t^n and we know the mesh velocities $\boldsymbol{\omega}_l^n = (\omega_{1,l}^n, \omega_{2,l}^n)^T$ at the vertex $P_l^n = (x_l^n, y_l^n)$ where l denotes the index of the mesh vertices. Then the mesh at the time level t^{n+1} is defined as

$$P_l^{n+1} := P_l^n + \boldsymbol{\omega}_l^n (t^{n+1} - t^n),$$

and the new mesh satisfies $\Omega = \bigcup_{i=1}^{N} I_i^{n+1}$, where the new triangular cells I_i^{n+1} are made up with the new vertices P_l^{n+1}. Define time-dependent straight lines for all $t \in [t^n, t^{n+1}]$,

$$P_l(t) := P_l^n + \boldsymbol{\omega}_l^n (t - t^n),$$

and these vertices $P_l(t)$ make up with the time-dependent triangular cell $I(t)$.

By an integration-by-parts method, we obtain the DG scheme on the moving mesh: Find a function $u_h \in \mathcal{V}_h(t)$ such that for all $v_h \in \mathcal{V}_h(t)$ and all cells $I(t)$,

$$\begin{aligned} \frac{d}{dt}(u_h, v_h)_{I(t)} =& (\tilde{f}(\omega_1, u_h), \partial_x v_h)_{I(t)} + (\tilde{g}(\omega_2, u_h), \partial_y v_h)_{I(t)} \\ &- \left\langle \hat{\tilde{f}}(\omega_1, u_h^{\text{in}}, u_h^{\text{ex}}, n_{x,I(t)}), v_h^{\text{in}} \right\rangle_{\partial I(t)} \\ &- \left\langle \hat{\tilde{g}}(\omega_2, u_h^{\text{in}}, u_h^{\text{ex}}, n_{y,I(t)}), v_h^{\text{in}} \right\rangle_{\partial I(t)} \end{aligned} \tag{4.18}$$

where $\tilde{f}(\omega_1, u_h) := f(u_h) - \omega_1 u_h$, $\tilde{g}(\omega_2, u_h) := g(u_h) - \omega_2 u_h$, and

$$\mathcal{V}_h(t) := \left\{ v_h \in L^2(\Omega) | v_h(x, y, t) \in \mathcal{P}^m \right\}.$$

Notice that $\vec{n}_{I(t)} = (n_{x,I(t)}, n_{y,I(t)})^T$ is the outer normal vector for the cell boundary. The values of u_h on the cell boundary $L \in \partial I(t)$ with outer normal vector $\vec{n}_L$ are defined as

$$u_h^{\text{in}}(\boldsymbol{x})|_L := \lim_{\varepsilon \to 0^+} u_h(\boldsymbol{x} - \varepsilon \vec{n}_L), \quad u_h^{\text{ex}}(\boldsymbol{x})|_L := \lim_{\varepsilon \to 0^+} u_h(\boldsymbol{x} + \varepsilon \vec{n}_L).$$

In our numerical test, we use the Lax-Friedrichs flux

$$\begin{aligned} \hat{\tilde{f}}(\omega_1, u_h^{\text{in}}, u_h^{\text{ex}}, n_{x,I(t)}) &= \frac{n_{x,I(t)}}{2} \left(\tilde{f}(\omega_1, u_h^{\text{in}}) + \tilde{f}(\omega_1, u_h^{\text{ex}}) \right) - \frac{\lambda_{1,I(t)}}{2} \left(u_h^{\text{ex}} - u_h^{\text{in}} \right) \\ \lambda_{1,I(t)} &= \max \left\{ \left| n_{x,I(t)} \partial_u \tilde{f}(\omega_1, u) \right| : t \in [t^n, t^{n+1}] \right\} \\ \hat{\tilde{g}}(\omega_2, u_h^{\text{in}}, u_h^{\text{ex}}, n_{y,I(t)}) &= \frac{n_{y,I(t)}}{2} \left(\tilde{g}(\omega_2, u_h^{\text{in}}) + \tilde{g}(\omega_2, u_h^{\text{ex}}) \right) - \frac{\lambda_{2,I(t)}}{2} \left(u_h^{\text{ex}} - u_h^{\text{in}} \right) \\ \lambda_{2,I(t)} &= \max \left\{ \left| n_{y,I(t)} \partial_u \tilde{g}(\omega_2, u) \right| : t \in [t^n, t^{n+1}] \right\} \end{aligned} \tag{4.19}$$

which satisfies consistency, monotonicity and Lipschitz continuity. We refer to [7] for the details of this Lagrangian type DG scheme.

If we define the matrix $A_{I(t)}$ as

$$A_{I(t)} := \left(P_{l_2}(t) - P_{l_1}(t), P_{l_3}(t) - P_{l_1}(t)\right),$$

then we can map the 2D triangular reference cell I_0 to the physical cell $I(t)$ as

$$P(t) := A_{I(t)}\boldsymbol{\xi} + P_{l_1}(t), \quad P(t) \in I(t), \quad \forall \boldsymbol{\xi} \in I_0$$

where the physical cell $I(t)$ is made up with $P_{l_1}(t), P_{l_2}(t), P_{l_3}(t)$.

The above DG scheme (4.18) can also be constructed on the reference cell I_0,

$$\begin{aligned}\left(\frac{d}{dt}(J_{I(t)}u_h^*), v_h^*\right)_{I_0} &= \left(J_{K(I)}\tilde{\tilde{f}}(\omega_1, u_h^*), \partial_x v_h^*\right)_{I_0} + \left(J_{K(I)}\tilde{\tilde{g}}(\omega_2, u_h^*), \partial_y v_h^*\right)_{I_0} \\ &- \left\langle \hat{\tilde{f}}(\omega_1, u_h^{\text{in},*}, u_h^{\text{ex},*}, J_{I(t)}\tilde{n}_x), v_h^{\text{in},*}\right\rangle_{\partial I_0} \\ &- \left\langle \hat{\tilde{g}}(\omega_2, u_h^{\text{in},*}, u_h^{\text{ex},*}, J_{I(t)}\tilde{n}_y), v_h^{\text{in},*}\right\rangle_{\partial I_0}\end{aligned} \tag{4.20}$$

where $J_{I(t)} = \det(A_{I(t)}) = 2|I(t)|$ is the determinant of the Jacobian matrix, $\tilde{\boldsymbol{n}}(t) = A_{I(t)}^{-T}\boldsymbol{n}_{I_0}$, u_h^*, v_h^* are defined on the reference cell I_0 and

$$\begin{pmatrix}\tilde{\tilde{f}}(\omega_1, u_h^*) \\ \tilde{\tilde{g}}(\omega_2, u_h^*)\end{pmatrix} = A_{I(t)}^{-1}\begin{pmatrix}\tilde{f}(\omega_1, u_h^*) \\ \tilde{g}(\omega_2, u_h^*)\end{pmatrix} = A_{I(t)}^{-1}\begin{pmatrix}f(u_h^*) - \omega_1 u_h^* \\ g(u_h^*) - \omega_2 u_h^*\end{pmatrix}.$$

Just like before, we develop a two-dimensional indirect ALE-DG scheme with the Lagrangian type DG method, the rezoning step and our high-order polynomial projection remapping method. The flowchart is as the same as that in Section 4.1, so we omit it here.

4.2.1 Numerical tests for the two-dimensional Euler equation of gas dynamics

Consider the two-dimensional Euler system

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho \\ \rho u \\ \rho v \\ E\end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix}\rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p)\end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix}\rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p)\end{pmatrix} = 0 \tag{4.21}$$

where ρ is the density, u, v are velocities on the x, y directions, E is the total energy, $p = (\gamma - 1)\left(E - \frac{1}{2}\rho(u^2 + v^2)\right)$ is the pressure for calorically ideal gas and γ is a constant that

depends on the particular gas under consideration. In the following, we will set the mesh velocity in the Lagrangian type moving mesh as $\boldsymbol{\omega} = (u, v)^T$, where u, v are the velocities of the fluid flows. Notice that, in the following five tests, we do not apply the multi-resolution WENO limiter, since it is not necessary for these tests.

The accuracy test. We design an accuracy test for the 2D Euler system on $[0, 4\pi] \times [0, 4\pi]$. The initial condition is:

$$\rho_0(x, y) = \frac{1 + 0.2\sin(\frac{x+y}{2})}{\sqrt{6}},\ u_0(x, y) = v_0(x, y) = \sqrt{\frac{\gamma}{2}}\rho_0(x, y),\ p_0(x, y) = \rho_0(x, y)^\gamma.$$

Suppose that $\rho(x, y, t)$, $u(x, y, t)$, $v(x, y, t)$, $p(x, y, t)$ are the exact solutions, and if we take $\gamma = 3$, then we can verify that $\sqrt{6}\rho(x, y, t)$ is the exact solution of the 2D Burgers' equation:

$$u_t + \left(\frac{u^2}{2}\right)_x + \left(\frac{u^2}{2}\right)_y = 0, \quad \text{with} \quad u_0(x, y) = 1 + 0.2\sin(\frac{x+y}{2})$$

and

$$u(x, y, t) = v(x, y, t) = \sqrt{\frac{\gamma}{2}}\rho(x, y, t), \quad p(x, y, t) = \rho(x, y, t)^\gamma.$$

The initial mesh divides the computational domain uniformly into small squares with mesh size $h = \frac{4\pi}{N_x}$, where $N_x = N_y$ are number of cells in each directions, then each square will be divided into two triangles with the same area. $N = 2N_xN_y$ is the total number of the triangular cells.

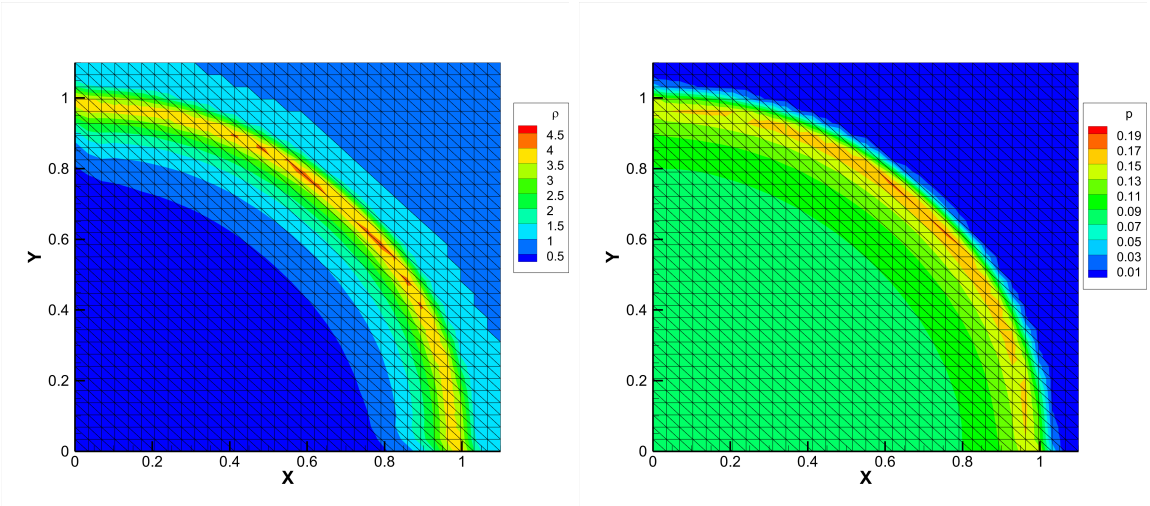
We show the numerical results of density obtained by the above three DG schemes at $t = 0.3$, and denote them as ρ_E, ρ_L and ρ_A in Table 4.6, respectively. In this test, we rezone the old mesh to the new uniform mesh and apply our remapping procedure in the ALE-DG scheme every 10 time steps. One can observe that all of these DG schemes have achieved the designed third-order accuracy.

The Sedov problem. Consider the Sedov problem with the initial condition as:

$$\begin{cases} \rho = 1, \\ u = 0, \\ v = 0, \end{cases} \quad (x, y) \in [0, 1.1] \times [0, 1.1], \tag{4.22}$$

Table 4.6: Error at time $t = 0.3$ for the two-dimensional Euler system with three third-order DG schemes.

	initial projection error $\|\|\rho(x,0) - \rho^0\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
200	2.6257E-05		3.0854E-05		5.7684E-05	
800	3.3675E-06	2.96	3.8656E-06	3.00	7.4948E-06	2.94
3200	4.2002E-07	3.00	4.8347E-07	3.00	9.4584E-07	2.99
7200	1.2440E-07	3.00	1.4327E-07	3.00	2.8075E-07	3.00
	$\|\|\rho(x,T) - \rho_E\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
200	3.4846E-05		4.5010E-05		1.2433E-04	
800	5.7684E-06	2.59	7.0419E-06	2.68	1.8629E-05	2.74
3200	5.9869E-07	3.27	7.7396E-07	3.19	2.8207E-06	2.72
7200	1.7409E-07	3.05	2.2913E-07	3.00	7.9754E-07	3.12
	$\|\|\rho(x,T) - \rho_L\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
200	3.4814E-05		4.4429E-05		1.2076E-04	
800	5.3049E-06	2.71	6.5904E-06	2.75	1.7712E-05	2.77
3200	5.6200E-07	3.24	7.2203E-07	3.19	2.7120E-06	2.71
7200	1.6017E-07	3.10	2.1291E-07	3.01	8.1529E-07	2.96
	$\|\|\rho(x,T) - \rho_A\|\|$					
N	L^1 error	order	L^2 error	order	L^∞ error	order
200	3.5181E-05		4.5255E-05		1.2388E-04	
800	5.8654E-06	2.58	7.1579E-06	2.66	1.9292E-05	2.68
3200	6.1853E-07	3.25	8.0111E-07	3.16	2.9961E-06	2.69
7200	1.7986E-07	3.05	2.3630E-07	3.01	8.4165E-07	3.13



(a) Density (b) Pressure

Figure 4.10: The Sedov problem of the Eulerian DG scheme.

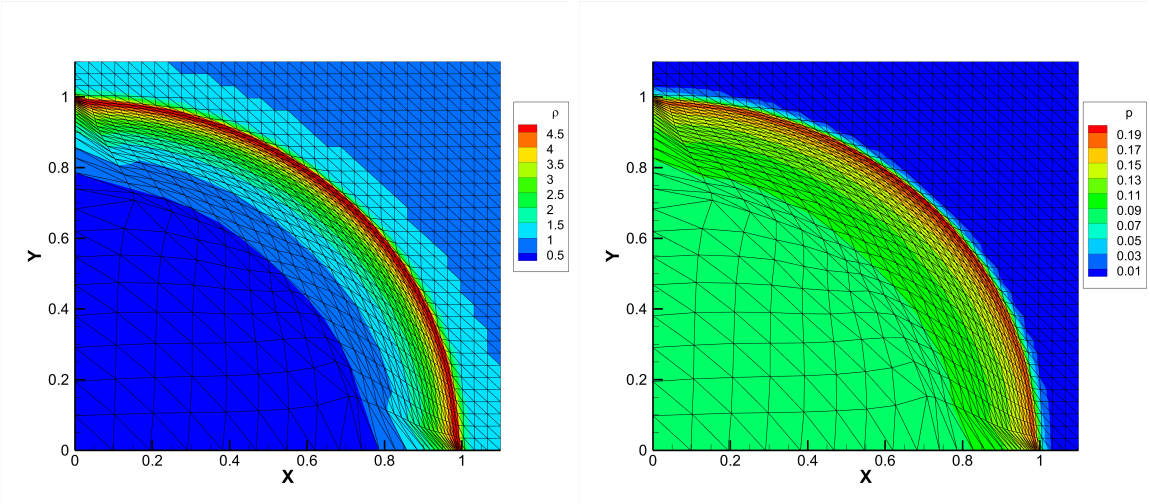
and the initial internal energy $e = 10^{-13}$ almost everywhere except for the only one cell I near the origin where we set $e = \frac{0.244816}{|I|}$. This problem is performed on the initially uniform mesh with 2,048 triangular cells. In Figure 4.10 and 4.11, we show the numerical results of density and pressure at time $t = 1$, calculated by the Eulerian DG scheme on the fixed mesh and the ALE-DG scheme.

For the Sedov problem, we utilize the rezoning procedure and the remapping procedure after $t > 0.5$ every 20 time steps, and we perform a simple smoothing operator on the inner points in the rezoning step.

For this case, the ALE-DG scheme captures the shock precisely and the mesh quality is well after adjusting the inner mesh. In Figure 4.12, we demonstrate the cut line at $x = y$ on these two DG schemes, and one can observe that the numerical diffusion for the ALE-DG scheme is much less than that for the Eulerian DG scheme.

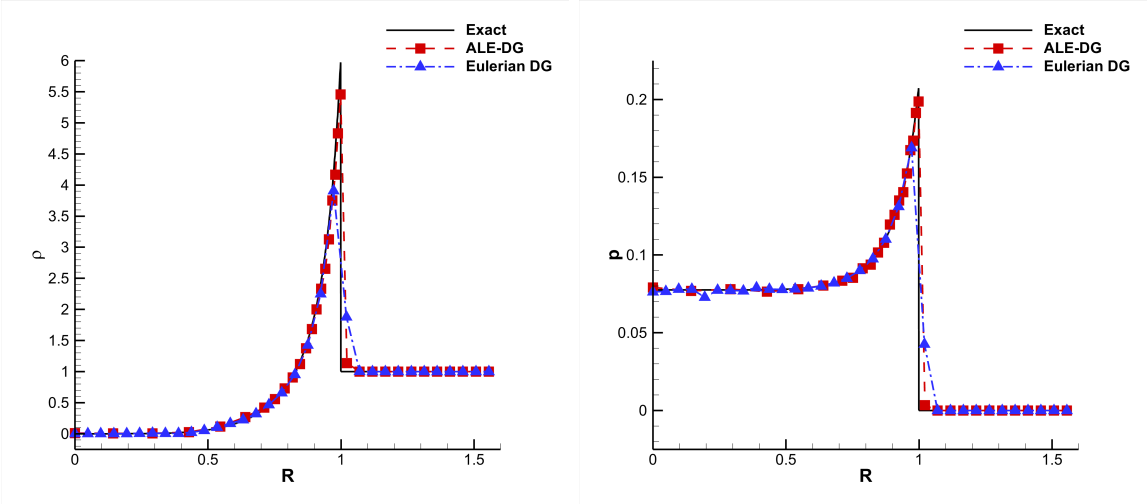
The Noh problem. Consider the Noh problem with the initial condition as:

$$\begin{cases} \rho = 1, \\ u_r = -1, \\ e = 10^{-13}, \end{cases} \quad (x, y) \in [0, 1] \times [0, 1], \tag{4.23}$$



(a) Density (b) Pressure

Figure 4.11: The Sedov problem of the ALE-DG scheme.



(a) Density (b) Pressure

Figure 4.12: The Sedov problem at the cut line $x = y$.

where u_r is the radial velocity and take $\gamma = \frac{5}{3}$. Reflective boundary conditions are considered for the left and below boundaries, besides that, free boundary conditions are considered for the right and top boundaries. In practice, the free boundary condition is set as the initial values. Since the initial internal energy is very close to 0 and the numerical results may be negative that makes the scheme unstable, the positivity-preserving limiter is essential in this test.

In Figure 4.13, we show the numerical results of density and pressure for the Eulerian DG scheme at time $t = 0.6$ with 2,048 triangular cells. In Figure 4.14, we show the results for the Lagrangian type DG scheme at $t = 0.058$ and we can observe that the mesh quality is very bad near the origin so we need to introduce the polynomial projection remapping procedure and the rezoning strategy.

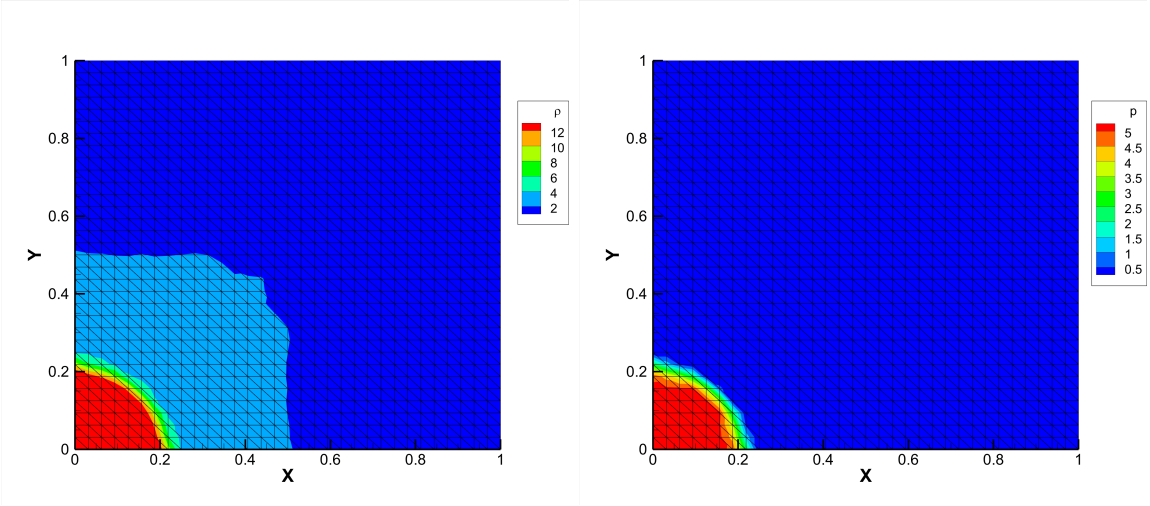
For the Noh problem, we perform the rezoning procedure and the remapping procedure after $t > 0$ every 10 time steps, and the rezoning strategy is as same as that in the Sedov problem. As one can see, the numerical results of the ALE-DG scheme are much better than the results on the fixed mesh, and the shock surface is sharper, in Figure 4.15. We demonstrate the cut line at $x = y$ in Figure 4.16, and one can observe that the ALE-DG scheme captures the shock well.

The Saltzman problem Consider the Saltzman problem with the initial condition as:

$$\begin{cases} \rho = 1, \\ u = 0, \\ v = 0, \\ e = 10^{-10}, \end{cases} \quad (x, y) \in [0, 1] \times [0, 0.1], \tag{4.24}$$

and take $\gamma = \frac{5}{3}$. Reflective boundary conditions are adopted for the right, up and below boundaries, besides that, the left boundary is a piston with velocity $u = 1$. Figure 4.17 shows the initial mesh on the computational domain $[0, 1] \times [0, 0.1]$ with 640 triangular cells.

For the cells I near the left boundary and its virtual neighbor cell J which has common



(a) Density (b) Pressure

Figure 4.13: The Noh problem of the Eulerian DG scheme at $t = 0.6$.

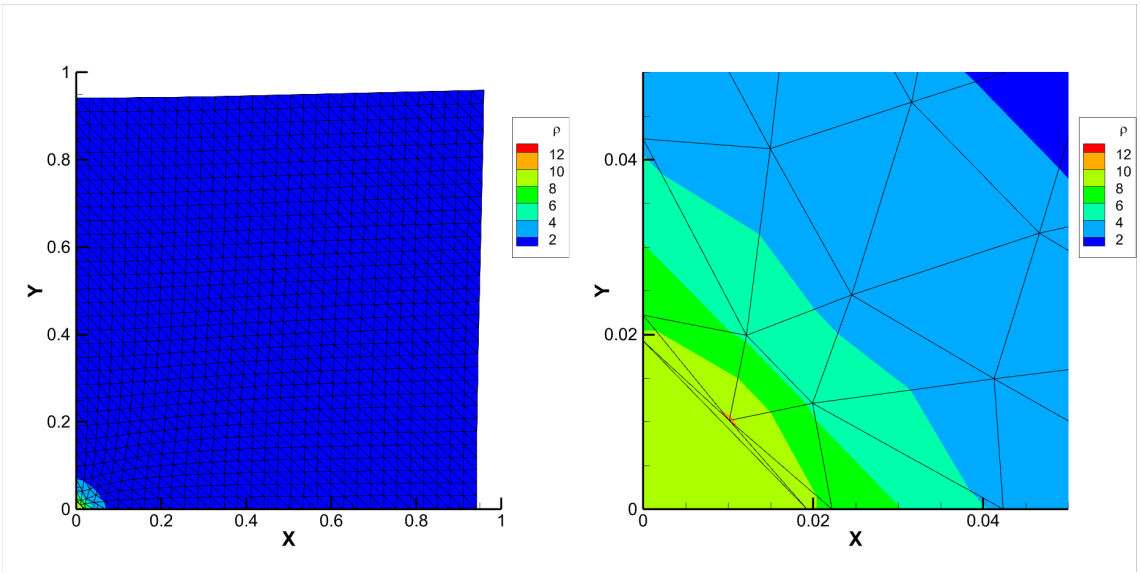
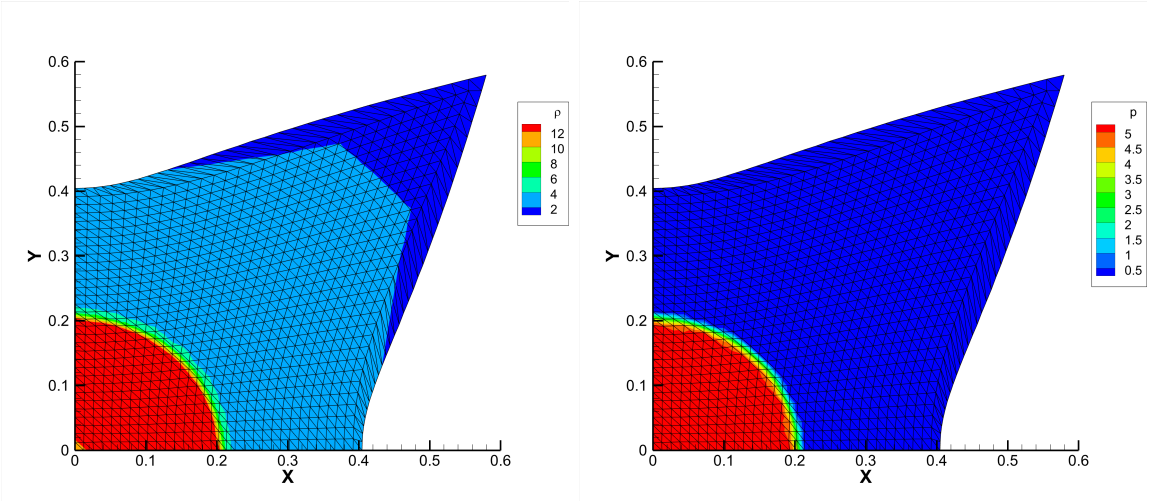
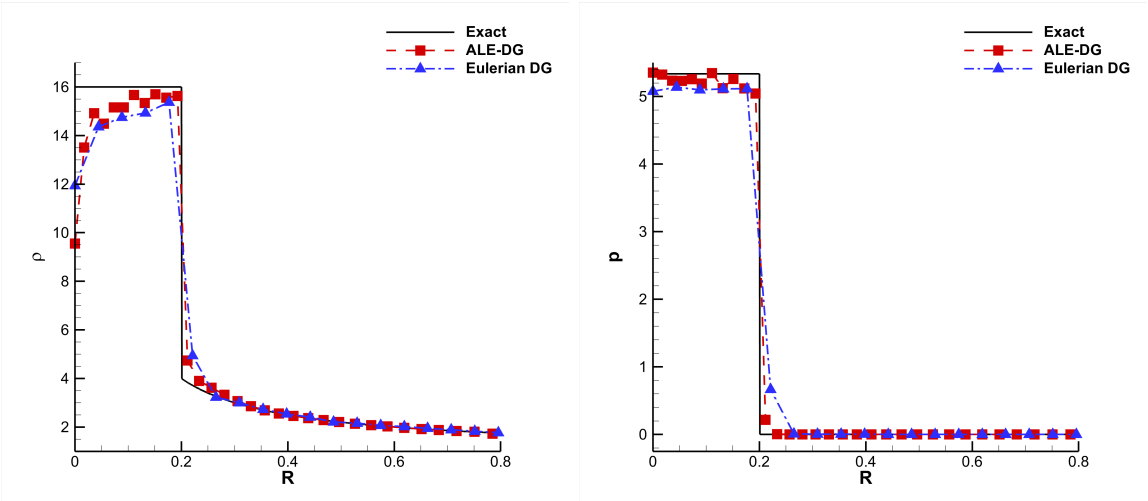


Figure 4.14: The Noh problem of the Lagrangian type DG scheme at $t = 0.058$. Right: the zoomed mesh near the origin.



(a) Density (b) Pressure

Figure 4.15: The Noh problem of the ALE-DG scheme at $t = 0.6$.



(a) Density (b) Pressure

Figure 4.16: The Noh problem at the cut line $x = y$.

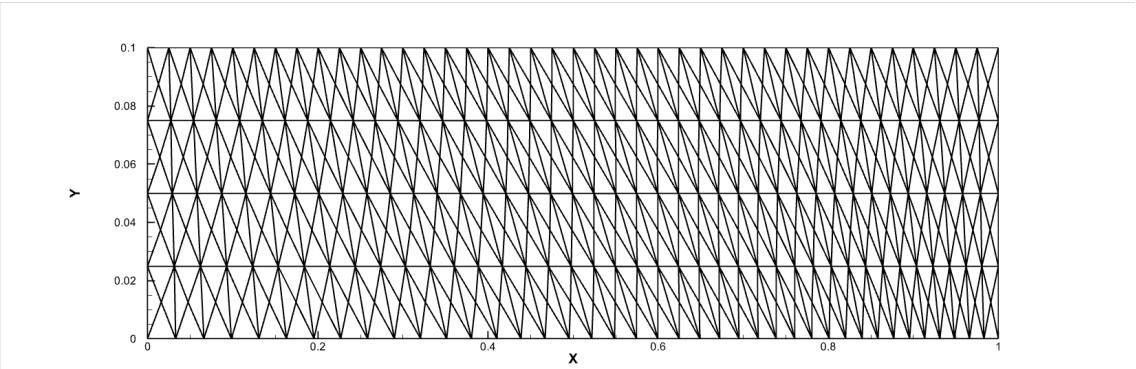


Figure 4.17: The initial Saltzman mesh.

edge with I on the left boundary, we take the values on the cell J as

$$\rho_J = \rho_I, \quad u_J = 2 - u_I, \quad v_J = v_I, \quad p_J = p_I.$$

For this Saltzman problem and the next Dukowicz problem, as the left boundary is moving, it is difficult for the Eulerian DG scheme to solve this kind of problem, thus we just use the Lagrangian type DG scheme and the indirect ALE-DG scheme to solve these problems.

We first try to use the Lagrangian type DG scheme to solve the Saltzman problem, but the triangular cells is squeezed and distorted soon (see Figure 4.18) and that stops the simulation. For the Saltzman problem, our rezoning method preserves the y-coordinates unchanged and modifies the inner point $P_l(x_l, y_l)$ in the x direction as

$$\tilde{x}_l := \frac{1}{4}\left(x_{l,1} + x_{l,2} + x_{l,3} + x_{l,4}\right),$$

where $x_{l,1}, x_{l,2}, x_{l,3}, x_{l,4}$ are the x-coordinates of the four neighbors of P_l. Therefore, we apply the rezoning method and our remapping procedure every 20 time steps to maintain the mesh quality, then we show the numerical results at $t = 0.6$ in Figure 4.19. The shock front in the ALE-DG scheme is clear and it is much more robust.

The Dukowicz problem. Last, consider the Dukowicz problem in [6]. The computational domain in the Dukowicz problem consists of two parts, the left one is a trapezoid with the

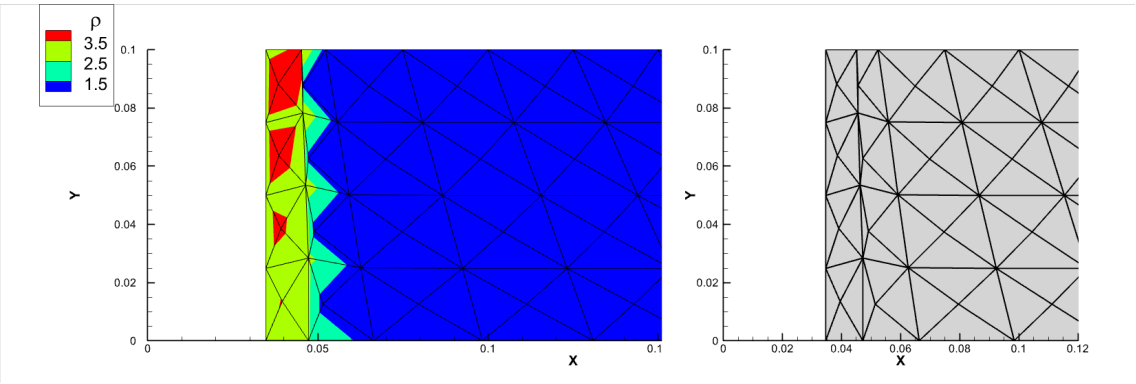


Figure 4.18: The Saltzman problem of the Lagrangian type DG scheme at $t = 0.034$. Right: the zoomed mesh near the left moving boundary.

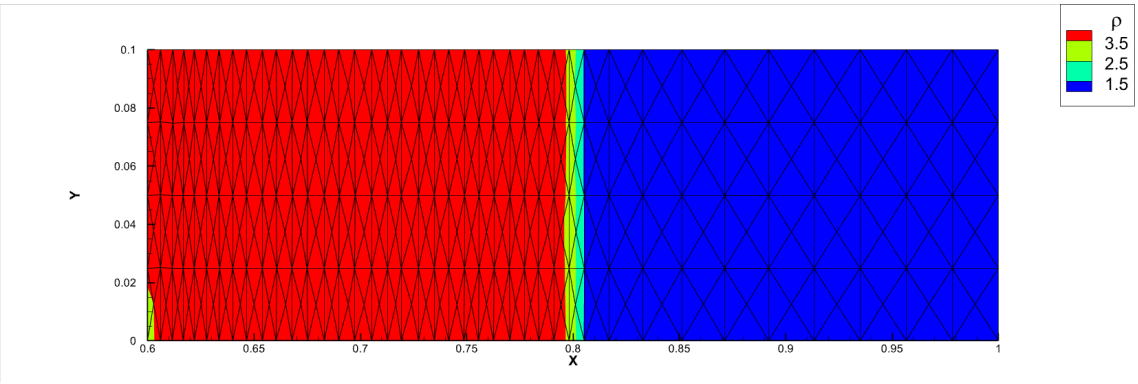


Figure 4.19: The Saltzman problem of the ALE-DG scheme at $t = 0.6$.

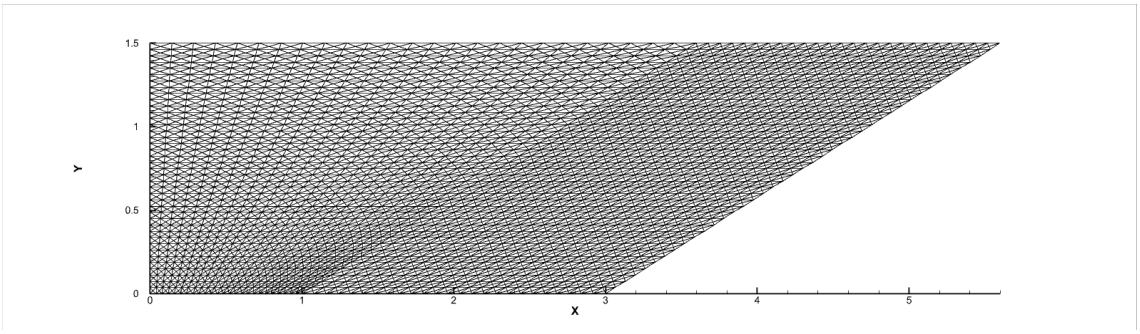


Figure 4.20: The initial Dukowicz mesh with 8,000 triangular cells.

vertical left boundary and the right boundary slanted at 60°. The right region is a slanted parallelogram and Figure 4.20 shows the initial computational mesh with $8,000$ triangular cells. The left region is made up with the trapezoid $(0,0)$, $(1,0)$, $(0,\frac{3}{2})$, $(1+\frac{3}{2}\sqrt{3},\frac{3}{2})$, and the right region is made up with the parallelogram $(1,0)$, $(3,0)$, $(1+\frac{3}{2}\sqrt{3},\frac{3}{2})$, $(3+\frac{3}{2}\sqrt{3},\frac{3}{2})$. The initial condition is given as

$$\begin{cases} \rho_L = 1, \\ u_L = 0, \\ v_L = 0, \\ p_L = 1, \end{cases} \quad \text{and,} \quad \begin{cases} \rho_R = 1.5, \\ u_R = 0, \\ v_R = 0, \\ p_R = 1, \end{cases} \tag{4.25}$$

where we take $\gamma = 1.4$ in the whole region.

Reflective boundary conditions are considered for the top, bottom and right boundaries, and the left boundary is a piston with velocity $u = 1.48$. The computational mesh is squeezed and distorted and that terminates the program in the Lagrangian type DG method, which can be seen in Figure 4.21. This time, our rezoning method preserves the y-coordinates unchanged and divide the mesh uniformly in the x direction. After adjusting the computational mesh and applying the remapping procedure every 20 time steps, we calculate to $t = 1.3$ with the indirect ALE-DG scheme and show the density contour in Figure 4.22. One can observe high resolution incident shock and transmitted shock interfaces in our ALE-DG scheme.

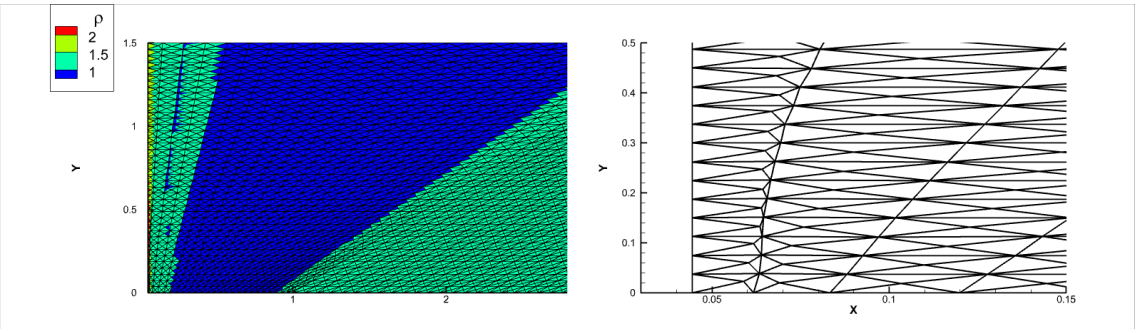


Figure 4.21: The Dukowicz problem of the Lagrangian type DG scheme at $t = 0.038$. Right: the zoomed mesh near the left moving boundary.

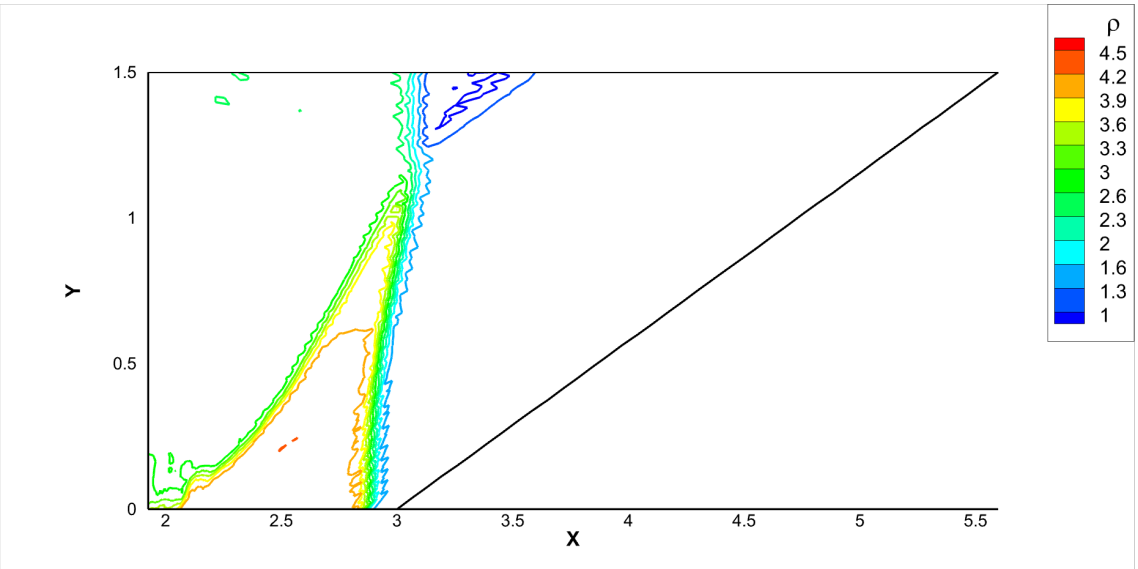


Figure 4.22: The Dukowicz problem of the ALE-DG scheme at $t = 1.3$.

5 Concluding remarks

In this paper, we develop a high-order accurate, essentially non-oscillatory, conservative and positivity-preserving polynomial projection remapping method in one, two and three dimensions to couple with the discontinuous Galerkin method for the Lagrangian type moving mesh, and establish an indirect ALE-DG framework. Since our remapping method is based on determining the intersections between the old and new meshes, it has a wider range of application. By adding the local multi-resolution WENO limiter, our remapping method can prevent the numerical oscillations generated by the high-order polynomials near the discontinuities. We also apply a positivity-preserving scaling limiter to ensure positivity without affecting the high order accuracy. We have designed a series of numerical tests in one, two and three dimensions to show that our remapping algorithm is high-order accurate, non-negative and essentially non-oscillatory. When used to solve the fluid dynamics, our remapping method is conservative for mass, momentum and total energy, and it can preserve positivity for density and internal energy. All of the above good properties have been verified by benchmark test problems for the Euler system. In future work, we will develop a three-dimensional indirect ALE-DG scheme as an application of this remapping method in three dimensions.

References

[1] M. Berndt, N.N. Carlson, *Using polynomial filtering for rezoning in ale*, Technical Report LA-UR 11-05015, Los Alamos National Laboratory, 2011.

[2] J. Cheng, C.-W. Shu, *A high order ENO conservative Lagrangian type scheme for the compressible Euler equations*, Journal of Computational Physics, 227 (2007), 1567-1596.

[3] B. Cockburn, C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework*, Mathematics of Com-

putation, 52 (1989), 411-435.

[4] B. Cockburn, C.-W. Shu, *Runge-Kutta discontinuous Galerkin methods for convection-dominated problems*, Journal of Scientific Computing, 16 (2001), 173-261.

[5] J. K. Dukowicz, J. R. Baumgardner, *Incremental remapping as a transport/advection algorithm*, Journal of Computational Physics, 160 (2000), 318-335.

[6] J. K. Dukowicz, J. W. Kodis, *Accurate conservative remapping (rezoning) for arbitrary Lagrangian-Eulerian computations*, SIAM Journal on Scientific and Statistical Computing, 8 (1987), 305-321.

[7] P. Fu, G. Schnucke, Y. Xia, *Arbitrary Lagrangian-Eulerian discontinuous Galerkin method for conservation laws on moving simplex meshes*, Mathematics of Computation, 88 (2019), 2221-2255.

[8] C. W. Hirt, A. Amsden, J. L. Cook, *An arbitrary Lagrangian-Eulerian computing method for all flow speeds*, Journal of Computational Physics, 14 (1974), 227-253.

[9] C. Klingenberg, G. Schnucke, Y. Xia, *Arbitrary Lagrangian-Eulerian discontinuous Galerkin method for conservation laws: Analysis and application in one dimension*, Mathematics of Computation, 86 (2017), 1203-1232.

[10] C. Klingenberg, G. Schnucke, Y. Xia, *An arbitrary Lagrangian-Eulerian local discontinuous Galerkin method for Hamilton-Jacobi equations*, Journal of Scientific Computing, 73 (2017), 906-942.

[11] P. Knupp, L.G. Margolin, M. Shashkov, *Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods*, Journal of Computational Physics, 176 (2002), 93-128.

[12] M. Kucharik, M. Shashkov, B. Wendroff, *An efficient linearity-and-bound-preserving remapping method*, Journal of Computational Physics, 188 (2003), 462-471.

[13] N. Lei, J. Cheng, C.-W. Shu, *A high order positivity-preserving conservative WENO remapping method on 2D quadrilateral meshes*, Computer Methods in Applied Mechanics and Engineering, 373 (2021), 113497.

[14] N. Lei, J. Cheng, C.-W. Shu, *A high order positivity-preserving conservative WENO remapping method on 3D tetrahedral meshes*, Computer Methods in Applied Mechanics and Engineering, 395 (2022), 115037.

[15] R. Li, T. Tang, P. Zhang, *Moving mesh methods in multiple dimensions based on harmonic maps*, Journal of Computational Physics, 170 (2001), 562-588.

[16] R. Li, T. Tang, P. Zhang, *A moving mesh finite element algorithm for singular problems in two and three space dimensions*, Journal of Computational Physics, 177 (2002), 365-393.

[17] K. Lipnikov, N. Morgan, *Conservative high-order discontinuous Galerkin remap scheme on curvilinear polyhedral meshes*, Journal of Computational Physics, 420 (2020), 109712.

[18] R. Loubere, P.-H. Maire, M. Shashkov, J. Breil, S. Galera, *ReALE: A reconnection-based arbitrary-Lagrangian-Eulerian method*, Journal of Computational Physics, 229 (2010), 4724-4761.

[19] L. G. Margolin, M. Shashkov, *Second-order sign-preserving conservative interpolation (remapping) on general grids*, Journal of Computational Physics, 184 (2003), 266-298.

[20] D. Powell, T. Abel, *An exact general remeshing scheme applied to physically conservative voxelization*, Journal of Computational Physics, 297 (2015), 340-356.

[21] C.-W. Shu, *TVB uniformly high-order schemes for conservation laws*, Mathematics of Computation, 49 (1987), 105-121.

[22] C.-W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, Journal of Computational Physics, 77 (1988), 439-471.

[23] I. Sutherland, G. W. Hodgman, *Reentrant polygon clipping*, Communications of the ACM, 17 (1974), 32-42.

[24] C. Wang, X. Zhang, C.-W. Shu, J. Ning, *Robust high order discontinuous Galerkin schemes for two-dimensional gaseous detonations*, Journal of Computational Physics, 231 (2012), 653-665.

[25] W. Wu, A-M. Zhang, M. Liu, *A cell-centered indirect Arbitrary-Lagrangian-Eulerian discontinuous Galerkin scheme on moving unstructured triangular meshes with topological adaptability*, Journal of Computational Physics, 438 (2021), 110368.

[26] X. Zhang, C.-W. Shu, *On maximum-principle-satisfying high order schemes for scalar conservation laws*, Journal of Computational Physics, 229 (2010), 3091-3120.

[27] Z. Zhang, *Moving mesh method with conservative interpolation based on L^2-projection*, Communications in Computational Physics, 1 (2006), 930-944.

[28] J. Zhu, J. Qiu, C.-W. Shu, *High-order Runge-Kutta discontinuous Galerkin methods with a new type of multi-resolution WENO limiters*, Journal of Computational Physics, 404 (2020), 109105.

[29] J. Zhu, J. Qiu, C.-W. Shu, *High-order Runge-Kutta discontinuous Galerkin methods with a new type of multi-resolution WENO limiters on triangular meshes*, Applied Numerical Mathematics, 153 (2020), 519-539.

[30] J. Zhu, C.-W. Shu, *A new type of multi-resolution WENO schemes with increasingly higher order of accuracy*, Journal of Computational Physics, 375 (2018), 659-683.

[31] J. Zhu, C.-W. Shu, J. Qiu, *High-order Runge-Kutta discontinuous Galerkin methods with a new type of multi-resolution WENO limiters on tetrahedral meshes*, Communications in Computational Physics, 29 (2021), 1030-1058.