

Moving Mesh Discontinuous Galerkin Method for Hyperbolic Conservation Laws

Ruo Li ^{*} Tao Tang [†]

October 4, 2004

Abstract

In this paper, a moving mesh discontinuous Galerkin (DG) method is developed to solve the nonlinear conservation laws. In the mesh adaptation part, two issues have received much attention. One is about the construction of the monitor function which is used to guide the mesh redistribution. In this study, a heuristic posteriori error estimator is used in constructing the monitor function. The second issue is concerned with the solution interpolation which is used to interpolate the numerical solution from the old mesh to the updated mesh. This is done by using a scheme that mimics the DG method for linear conservation laws. Appropriate limiters are used on seriously distorted meshes generated by the moving mesh approach to suppress the numerical oscillations. Numerical results are provided to show the efficiency of the proposed moving mesh DG method.

1 Introduction

It has been demonstrated that the discontinuous Galerkin (DG) method is a very powerful tool for solving partial differential equations (PDEs) such as nonlinear conservation laws (see, e.g., [10, 7]) and Maxwell equation (see, e.g., [8, 16]). Since the solutions to these problems may be discontinuous, it is more reasonable to approximate them in a discontinuous finite dimensional space. This is the primary reason that the DG method is very effective for solving certain classes of problems. Since there are no continuity requirement between the elements, the geometry of each element can be very flexible. Such character makes the DG method an attractive choice as the PDE solver for moving methods which may yield very irregular meshes. On the other hand, if an error indicator is provided then the combination of *hp* method and the DG method has been proved useful [4].

^{*}LMAM & School of Mathematical Sciences, Peking University, P.R. China, 100871, Email: rli@math.pku.edu.cn

[†]Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong & Institute of Computational Mathematics, Chinese Academy of Sciences, Beijing, China. Email: ttang@math.hkbu.edu.hk, ttang@lsec.cc.ac.cn

The moving mesh methods involve the solution of the underlying PDE in conjunction with a so-called moving mesh PDE for the mesh itself. The methods keep the total number of grid points unchanged, and can cluster more grid points to areas with singularities or large solution gradients, see, e.g., [5, 2]. As a result, it is very useful for time-dependent problems with localized singularities. The basic idea of moving mesh method is to construct a transformation from a logical domain (or called computational domain) to the physical domain. A fixed mesh is given on the logical domain, and the transformation is realized by solving moving mesh PDEs or minimization problems for a mesh functional, see, e.g., [1, 14]. In [15], such a transformation is constructed by using harmonic mappings, see also [12]. The numerical procedure proposed in [15] has now been extended to solve several classes of problems such as the incompressible Navier-Stokes equations [11] and elliptic optimal control problems [13]. One of the primary features of the numerical scheme is that the mesh redistribution part and the PDE evolution part are separated. As a result, the whole moving mesh algorithm can be packed in a black box which requires the following inputs: the current numerical solution of the underlying PDEs, the algorithm for solving the mesh PDEs, and an interpolation algorithm. Such a black box has been implemented in the adaptive finite element package *AFEPack* which is available at <http://circus.math.pku.edu.cn/AFEPack>.

In this work, the DG method will be used to solve the hyperbolic conservation laws. Hence, in viewing about the three required inputs in the *AFEPack*, we only need to consider two issues, namely the monitor function used in the mesh PDEs and the interpolation algorithm. The interpolation algorithm will be developed based on the general principle proposed in [15]. To provide an effective monitor function, our basic concern is that the mesh redistribution can not only catch the strong discontinuities (such as shock) but also resolve some weak singularities (such as rarefaction waves). To achieve this goal, the commonly used gradient-based monitor will not be used. Instead, we will propose a cutoff technique which uses a threshold to avoid clustering too many elements in the area with strong discontinuity. With the use of the cutoff technique, the small (weak) structures are enlarged so that they can be well resolved with locally finer meshes.

The paper is organized as follows. In section 2, the general idea of DG method for conservation law is described. The moving mesh methods will be briefly described. In section 3, several important issues relevant to the moving mesh DG methods will be discussed. Numerical results are given in section 4 to show the effectiveness of the new developed method.

2 Preliminaries

2.1 DG for hyperbolic conservation laws

In this section, we follow [9, 6] to briefly discuss the DG method for hyperbolic conservation laws. Consider the nonlinear system of conservation laws

$$\frac{\partial u}{\partial t} + \nabla \cdot f(u) = 0, \quad x \in \Omega, \quad t > 0, \quad (2.1)$$

where Ω is an open domain in \mathbf{R}^n , $u: \mathbf{R}^n \rightarrow \mathbf{R}^m$, and $f: \mathbf{R}^m \rightarrow \mathbf{R}^m$. With a given triangulation $T_h = \{K | \cup \bar{K} = \Omega\}$ on Ω , the finite dimension space used to approximate the solution u is chosen as $V_h = \{v_h | v_h|_K \in P^k(K)\}$. A weak formulation of (2.1) on a single element is given by

$$\int_K \frac{\partial u_h}{\partial t} v_h dx + \sum_{e \in \partial K} \int_e f(u_h) \cdot \vec{n} v_h ds - \int_K f(u_h) \cdot \nabla v_h dx = 0, \quad \forall v_h \in V_h \quad (2.2)$$

where e is the boundary of the element K . Replacing the boundary integration using the numerical flux $f(u_h) \rightarrow h_{e,K}(u_h)$ gives

$$\int_K \frac{\partial u_h}{\partial t} v_h dx + \sum_{e \in \partial K} \int_e h_{e,K}(u_h) \cdot \vec{n}_{e,K} v_h ds - \int_K f(u_h) \cdot \nabla v_h dx = 0. \quad (2.3)$$

Thus, the relationship between the degree of freedoms from the neighboring elements is built to reflect the convection property of the underlying PDEs. There are many forms of numerical flux advocated, among them the Godunov flux is of minimal numerical diffusion and the Lax-Friedrich flux is the most convenient for coding. The Lax-Friedrich flux is formulated as

$$h_{e,K}(u_h) := \frac{1}{2} \left[f(u_h^+) + f(u_h^-) - \alpha (u_h^+ - u_h^-) \cdot \vec{n}_{e,K} \right], \quad (2.4)$$

where α is the maximal eigenvalue of $\partial f / \partial u$. If the maximal is on the edge e (the whole domain Ω), then (2.4) is called local (global) L-F flux. A limiter strategy is adopted following [9, 6].

The TVD Runge-Kutta scheme is adopted in temporal discretization to guarantee numerical stability. The difference of the TVD Runge-Kutta scheme from the general Runge-Kutta scheme is that special composition coefficients are used and the time step is chosen small enough to make the resulting scheme TVD.

2.2 Moving Mesh Method

We follow [15, 14] to describe our moving mesh approach, which is divided into two independent parts, namely mesh redistribution and PDE evolution.

Denote the physical domain by Ω and the logical domain as Ω_c . The mesh transformation from the physical domain to the logical domain

$$\xi : x \mapsto \xi, \Omega \rightarrow \Omega_c$$

can be obtained by solving the following elliptic system

$$\nabla_x (m \nabla_x \xi) = 0 \quad (2.5)$$

where m is called monitor function. To obtain the coordinate of the physical mesh grid, the inverse of $\xi(x)$ should be obtained which is a transformation from the logical domain to the physical domain. Though the governing PDEs for the inverse transformation can

be derived from (2.5), the resulting system is highly nonlinear and much more complicated than (2.5). To overcome this difficulty, an iterative procedure was proposed in [15].

We now discuss the method to interpolate the numerical solution from the old mesh to the new mesh. This will be done by assuming that the information about the old mesh, the new mesh and the numerical solution on the old mesh is all known. More precisely, assume the given PDEs have the following general form

$$u_t = L(u). \quad \text{in } \Omega \quad (2.6)$$

Assume the solution u is approximated by u_h in a finite dimensional space V_h . It is required that the updated numerical solution u_h^{new} on the new mesh in the new finite dimensional space V_h^{new} has the following orthogonality property

$$u_h - u_h^{\text{new}} \perp V_h. \quad (2.7)$$

This requirement can be understood from the following point of view. Assume a Galerkin spatial discretization and a forward Euler temporal discretization is used for (2.6), which gives

$$\left(u_h^{(n+1)} - u_h^{(n)}, v_h^{(n)} \right) = \frac{1}{\Delta t} \left\langle L(u_h^{(n)}), v_h^{(n)} \right\rangle, \quad \forall v_h^{(n)} \in V_h. \quad (2.8)$$

After the mesh is redistributed by solving (2.5), an approximate solution on the new mesh should satisfy

$$\left(u_h^{(n+1),\text{new}} - u_h^{(n)}, v_h^{(n)} \right) = \frac{1}{\Delta t} \left\langle L(u_h^{(n)}), v_h^{(n)} \right\rangle, \quad \forall v_h^{(n)} \in V_h. \quad (2.9)$$

The orthogonal property (2.7) is then obtained by subtracting (2.9) from (2.8), which provides a general interpolation formulation, see [11] for an application to incompressible flow simulations.

3 Moving mesh DG method

At each time step, the moving mesh DG method first solves the given PDE using the DG method and then redistributes the meshes based on the known PDE solutions. Since the main ideas of the DG method and the moving mesh method have been described in the last section, we only need to emphasize several important issues which are very important in obtaining high quality moving mesh solutions.

3.1 Monitor function

Let us consider the monitor function for the piecewise linear approximation. For a smooth solution, the jump of the numerical solution across the edge of elements is about $\mathcal{O}(h^2)$. Thus, it can be deduced that the normal jump of the gradient for the numerical solution across the edge of elements is of order 1. Since the formal rate of convergence for the approximation solution is 2, the following quantity should be of order $\mathcal{O}(1)$:

$$\eta_K := \sum_{e \in \partial K} \frac{1}{|K|} \int_e \left\{ \llbracket u_h \rrbracket^2 h_e^{-1} + \llbracket \nabla u_h \cdot \vec{n}_{e,K} \rrbracket^2 h_e \right\} ds, \quad (3.1)$$

where $[[\cdot]]$ denotes the jump along the element edges.

The quantity η_K is a reasonable indicator for the errors of the numerical approximation, which is found very large along the shocks while relatively small in the regions with weaker singularities (such as rarefaction wave). Moreover, this quantity is exactly zero for the constant and linear part of the solution. However, using η_K directly will cluster too many points in the shock regions, which is obviously unnecessary. To prevent this from happening, a threshold $\bar{\eta}$ is used in the monitor function. More precisely, our monitor function is of the form

$$m|_K = \sqrt{\bar{\eta} + \alpha \min(\bar{\eta}, \eta_K)} \quad (3.2)$$

where $\alpha > 0$ is a user-defined parameter. The use of α is to control the ratio of the maximal and minimal element sizes. In general, α is small if the underlying solution is smooth (say $\alpha = 1$) and is large otherwise (say $\alpha = 200$). In our computations, The cut-off function $\bar{\eta}$ can be chosen as a constant or the average of the error function η :

$$\bar{\eta} = \frac{1}{|\Omega|} \int_{\Omega} \eta dx. \quad (3.3)$$

A similar averaging idea was also used in [3, 20].

The monitor (3.2) is smoothed using the method provided in [15]. The number of smoothing steps is set to be the largest integer not greater than $\sqrt{\#\text{ nodes}/5}$ according to our numerical experience.

3.2 Conservative interpolation

To interpolate the numerical solution from the old mesh to the new mesh, the technique proposed in [15] will be employed. Roughly speaking, the interpolation is implemented by solving a linear convection PDE. The difference here is that this linear convection PDE will be solved by using the DG scheme so that the solution conservation is preserved. Denote the velocity field by $\vec{\delta x}$ constructed using node-motion, which is piecewise linear and global continuous. Then the linear convection PDE is of the form, see [15]

$$\frac{\partial u}{\partial \tau} - \vec{\delta x} \cdot \nabla u = 0. \quad (3.4)$$

where u is the solution of the given PDE. The DG discretization to (3.4) is given by

$$\begin{aligned} & \int_K \frac{\partial u_h}{\partial \tau} v_h dx = \int_K \vec{\delta x} \cdot \nabla u_h v_h dx \\ &= \int_K \left\{ \nabla \cdot (u_h \vec{\delta x}) v_h - \nabla \cdot \vec{\delta x} u_h v_h \right\} dx \\ &= \sum_{e \in \partial K} \int_e u_h v_h \vec{\delta x} \cdot \vec{n}_{e,K} ds - \int_K u_h \vec{\delta x} \cdot \nabla v_h dx - \nabla \cdot \vec{\delta x} \int_K u_h v_h dx. \end{aligned} \quad (3.5)$$

Since the Lax-Friedrichs flux is the same as the upwind scheme for linear convection equations, the integration of the numerical flux in (3.5) is replaced by the upwind flux.

3.3 Degeneration of limiter

Let us first describe the limiters proposed in [9, 6] on triangle mesh. Assume the neighbors of K_0 are K_1 , K_2 and K_3 , see Fig. 3.3(a). The numerical solution u_h is piecewise linear on each element. Let C_i , $0 \leq i \leq 3$, be the barycenter of K_i , and B_j , $1 \leq j \leq 3$ be the midpoint of the common edge of K_0 and K_j (see Fig. 3.3(a), where only B_2 is labeled). The mean value of the numerical solution on element K_i , $0 \leq i \leq 3$ is denoted by \bar{u}_i . Then u_h on K_0 is determined by its values at B_j , $1 \leq j \leq 3$. Assume there are two nonnegative coefficients β_1 and β_2 such that

$$\overrightarrow{C_0 B_2} \triangleq \beta_1 \overrightarrow{C_0 C_2} + \beta_2 \overrightarrow{C_0 C_3}, \quad (3.6)$$

where β_1 and β_2 depend only on the element geometry. Denote

$$\Delta \tilde{u}_h(B_2) = \beta_1(\bar{u}_2 - \bar{u}_0) + \beta_2(\bar{u}_3 - \bar{u}_0).$$

We can compare $\Delta u_h(B_2) \triangleq u_h(B_2)|_{K_0} - \bar{u}_0$ and $\Delta \tilde{u}_h(B_2)$. If $\Delta u_h(B_2) > \nu \Delta \tilde{u}_h(B_2)$, where ν is a problem dependent parameter, then the solution on K_0 should be limited. In our implementation, ν is set as 1.5.

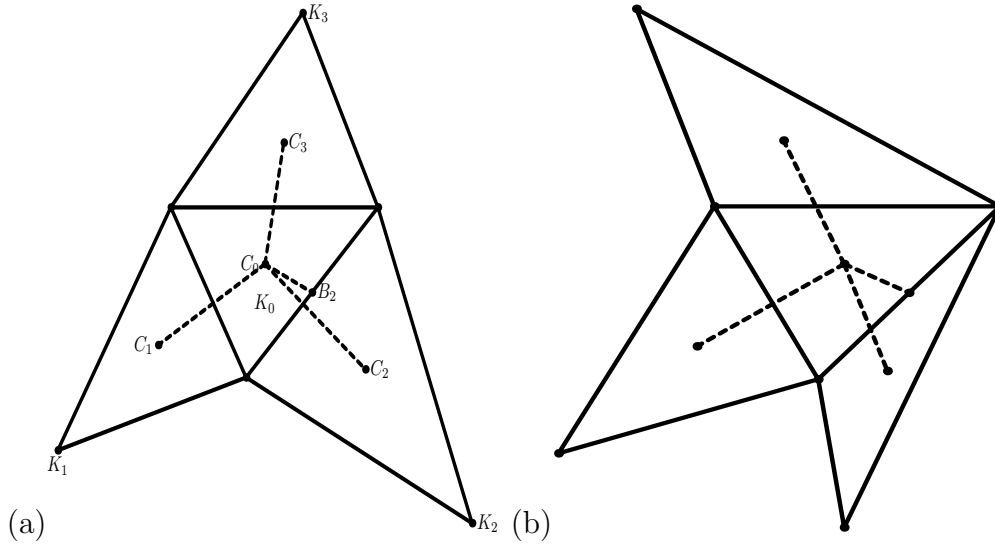


Figure 1: (a): The limiter proposed in [9, 6]; (b): an example of a not seriously distorted element, for which the limiter proposed in [9, 6] is not applicable.

This limiter is applicable only if the nonnegative coefficient β_1 and β_2 exist. However, this is not always true as demonstrated in Fig. 3.3(b). For meshes generated by the moving mesh methods, the situations similar to that in Fig. 3.3(b) have been observed. To fix this problem, the flatten limiters are used, namely, we set the numerical solution on K_0 as the constant mean value u_0 .

4 Numerical Results

4.1 Convergence order

Example 4.1 *To test the convergence order of our moving mesh DG method, we first consider a scalar linear equation whose solution is smooth:*

$$u_t + \vec{v} \cdot \nabla u_x = 0, \quad x \in (0, 1)^2 \quad (4.1)$$

with a constant convection speed $\vec{v} = (1, 1)$. The initial condition is chosen as $u_0(x, y) = \sin 2\pi x \sin 2\pi y$.

The exact solution of Example 4.1 is $u(x, y; t) = u_0(x - t, y - t)$. The monitor function used for this example is (3.2) with $\alpha = 1$. The resulting meshes using 10×10 , 20×20 and 40×40 elements are displayed in Fig. 4.1 and the corresponding errors are plotted in Fig. 2 from which a second-order rate of convergence is observed.

Example 4.2 *The second example contains a variable convection speed:*

$$u_t + \vec{v} \cdot \nabla u_x = 0, \quad x \in (0, 1)^2, \quad (4.2)$$

with $\vec{v} = (1/2 - y, x - 1/2)$. The initial and boundary conditions are chosen such that the exact solution is given by

$$u(x, y; t) = \tanh 50 \left[(x - 1/2) \cos(t) + (y - 1/2) \sin(t) \right].$$

Unlike Example 4.1, the solution of this problem has large solution gradients as seen in Fig. 4. In Fig. 4, the moving meshes at $t = 1.2$ are obtained by using 20×20 and 40×40 elements. In Fig. 5, numerical errors obtained by using the two sets of elements are presented. It is seen that the order of convergence is higher than 2, e.g., at $t = 1.2$, the error with the finer mesh is about 1/7 of that with the coarse mesh. The order enhancement for the moving mesh method to problems with large gradients has been observed for many other computations, see, e.g., [18].

In both of the above examples, the cutoff threshold is chosen as the average of the error indicator, i.e., (3.3).

4.2 Euler Equation

Consider the Euler equations

$$\frac{\partial U}{\partial t} + \frac{\partial E(U)}{\partial x} + \frac{\partial F(U)}{\partial y} = 0, \quad (4.3)$$

where

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ e \end{pmatrix}, \quad E(U) = \begin{pmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ u(p + e) \end{pmatrix}, \quad F(U) = \begin{pmatrix} \rho v \\ \rho uv \\ p + \rho v^2 \\ v(p + e) \end{pmatrix}.$$

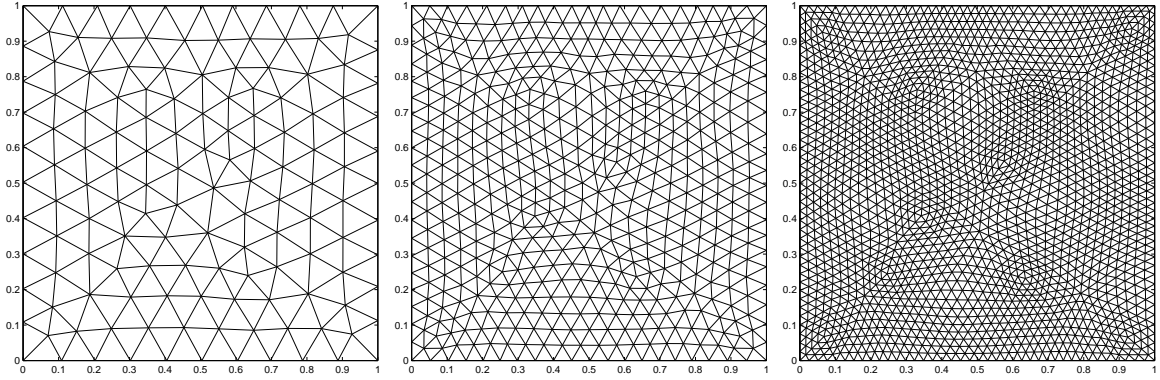


Figure 2: Example 4.1: meshes obtained with 10×10 , 20×20 , 40×40 elements.

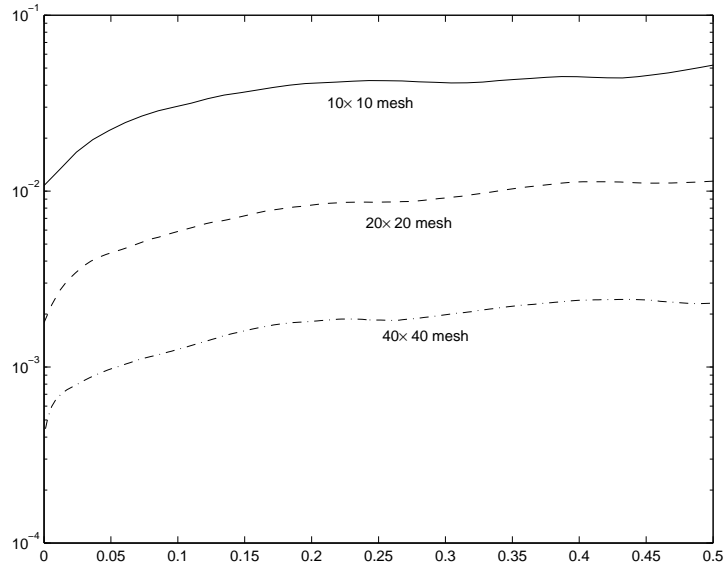


Figure 3: Example 4.1: numerical errors obtained using three different meshes.

The state relation is given by

$$e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho(u^2 + v^2). \quad (4.4)$$

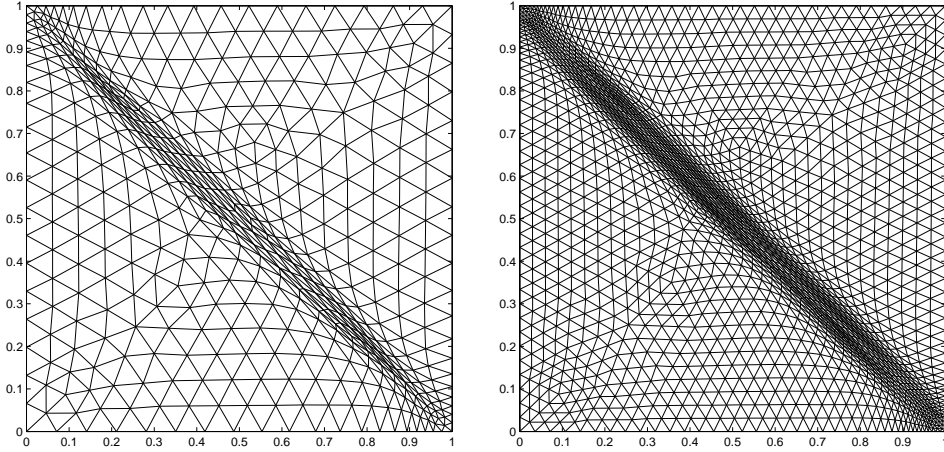


Figure 4: Example 4.2: moving meshes using 20×20 and 40×40 elements.

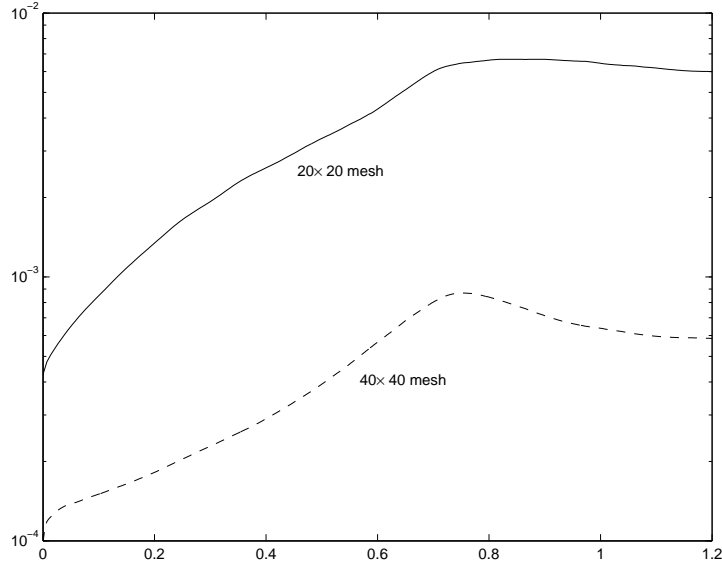


Figure 5: Example 4.2: errors obtained using 20×20 and 40×40 elements.

The problem is discretized on triangular meshes using second order DG scheme. On each triangle, the numerical solution is linear. The discretized system is

$$\begin{aligned} \frac{d}{dt} \int_K \rho_h w_h dx + \sum_{e \in \partial K} \int_e h_{e,K}^1 w_h d\Gamma - \int_K \left\{ \rho_h u_h \frac{\partial w_h}{\partial x} + \rho_h v_h \frac{\partial w_h}{\partial y} \right\} dx &= 0, \\ \frac{d}{dt} \int_K \rho_h u_h w_h dx + \sum_{e \in \partial K} \int_e h_{e,K}^2 w_h d\Gamma - \int_K \left\{ (p_h + \rho_h u_h^2) \frac{\partial w_h}{\partial x} + \rho_h u_h v_h \frac{\partial w_h}{\partial y} \right\} dx &= 0, \\ \frac{d}{dt} \int_K \rho_h v_h w_h dx + \sum_{e \in \partial K} \int_e h_{e,K}^3 w_h d\Gamma - \int_K \left\{ \rho_h u_h v_h \frac{\partial w_h}{\partial x} + (p_h + \rho_h v_h^2) \frac{\partial w_h}{\partial y} \right\} dx &= 0, \\ \frac{d}{dt} \int_K e_h w_h dx + \sum_{e \in \partial K} \int_e h_{e,K}^4 w_h d\Gamma - \int_K \left\{ (u_h (p_h + e_h)) \frac{\partial w_h}{\partial x} + (v_h (p_h + e_h)) \frac{\partial w_h}{\partial y} \right\} dx &= 0, \end{aligned}$$

where the boundary integrals are integrated numerically by using a two-point Gaussian formula with integration points at $\pm 1/\sqrt{3}$, and \int_K is implemented as a three-point Gaussian formula with the integration points located at the midpoint of the three edges. The numerical flux on edge e is set as

$$h_{e,K_1}(a,b) = \frac{1}{2} \left[(E F) \cdot n_{e,K_1} - \alpha_{e,K_1} (b - a) \right],$$

where K_1 and K_2 are the two neighboring elements sharing the common edge e and a, b are $U_h^i|_{K_1}$ and $U_h^i|_{K_2}$, respectively. The value of α_{e,K_1} is great than the maximal eigenvalue of $\partial_U (E F) \cdot n_{e,K_1}|_{K_1}$ and $\partial_U (E F) \cdot n_{e,K_1}|_{K_2}$. Since the eigenvalues of $\partial_U E$ and $\partial_U F$ are $u \pm c, u$ and $v \pm c, v$, respectively, it can be shown that the eigenvalues of $\partial_U (E F) \cdot n_{e,K_1}$ are $(uv) \cdot n_{e,K_1} \pm c$ and $(uv) \cdot n_{e,K_1}$.

4.2.1 Riemann problem

The 2D Riemann problem is now a standard test problem. It is defined in the unit square domain and the initial condition is given by

$$(\rho, u, v, p) = \begin{cases} (1.1, 0.0, 0.0, 1.1), & \text{if } x > 0.5, y > 0.5, \\ (0.5065, 0.8939, 0.0, 0.35), & \text{if } x < 0.5, y > 0.5, \\ (1.1, 0.8939, 0.8939, 1.1), & \text{if } x < 0.5, y < 0.5, \\ (0.5065, 0.0, 0.8939, 0.35), & \text{if } x > 0.5, y < 0.5, \end{cases} \quad (4.5)$$

where p is the pressure. The constant γ in (4.4) is set to be 1.4.

The cutoff threshold $\bar{\eta}$ is chosen a large constant 10^2 . The parameter α for this example is also chosen as 10^2 . Fig. 6 presents the density contours obtained by using 100×100 and 160×160 elements. It is seen that the use of the monitor (3.2) can catch the ditch along the center line of the leaf-shaped structure. Fig. 7 shows the adaptive mesh with 100×100 elements, from which it is observed that more elements move not only to the shock regions but also to the areas with weaker structure. This is due to the use of the monitor function (3.2), which is in contrast with the gradient-based monitors commonly used for hyperbolic problems, see, e.g., [17].

4.2.2 Double Mach reflection problem

This problem was studied by Woodward and Colella [19] and has been used as a standard test problem since then. Its DG results can be found in [10, 7] and the moving mesh results for can be found in [17]. The domain is a rectangle $[0, 4] \times [0, 1]$. A right moving shock is initially positioned at $(1/6, 0)$ and makes a 60 degree angle with the x -axis. The inflow is with Mach number 10. The boundary condition at bottom is the exact post shock condition from 0 to $1/6$ and is reflective for the rest. At the top boundary, the flow values are set to describe the exact motion of the Mach 10 shock. On the left and right boundaries, the inflow and outflow boundary conditions are used respectively.

The cutoff threshold $\bar{\eta}$ for this example is chosen as 10^3 , and the parameter α in (3.2) is again chosen as 200. We calculate this problem on 96×24 and 192×48 meshes to time

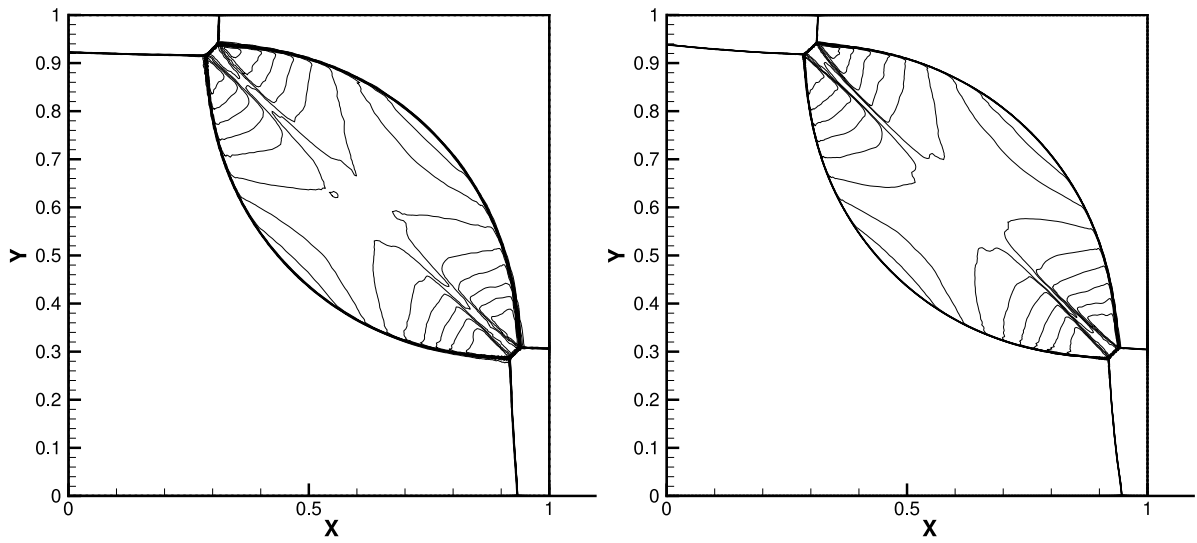


Figure 6: Riemann problem: density contours using 100×100 (top) and 160×160 (bottom) elements.

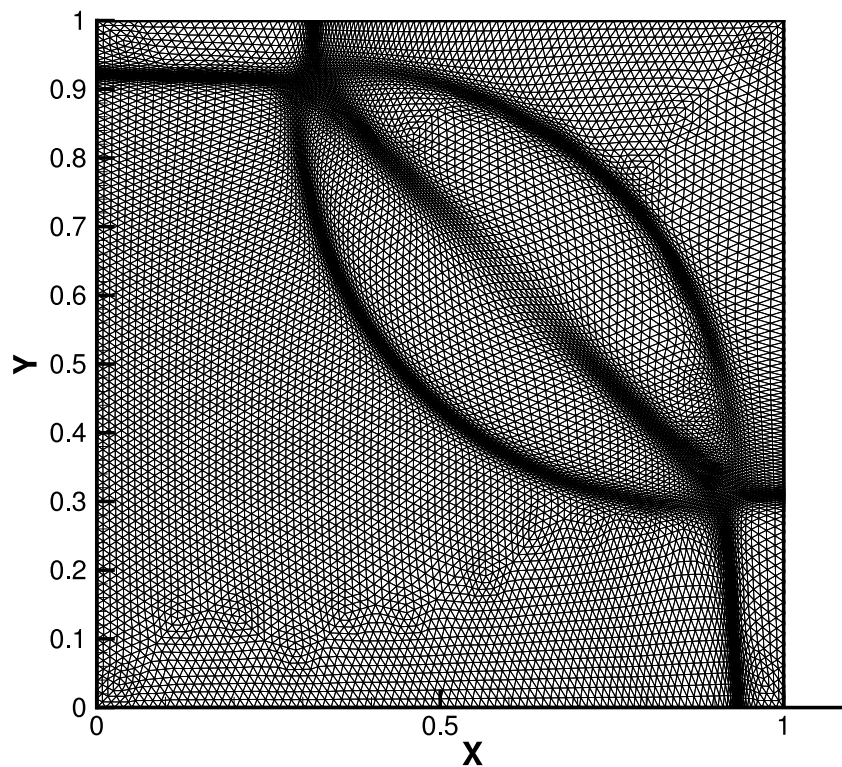


Figure 7: Riemann problem: moving mesh with 100×100 elements.

$t = 0.2$. The mesh and density contours using the two meshes are plotted in Figs. 8 and 9, respectively. The density contour is plotted on $[0, 3] \times [0, 1]$. It can be seen that the mesh grids are clustered in the regions where the shocks and the detailed structures are located. As expected, more accurate solutions are obtained with the finer mesh.

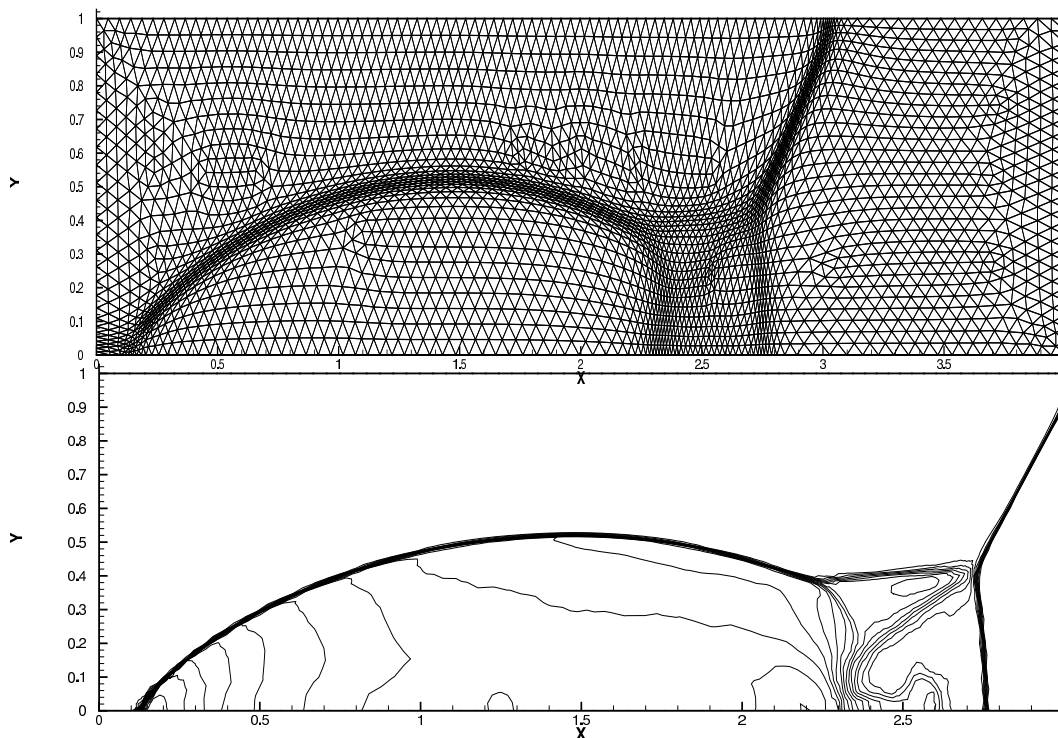


Figure 8: Double Mach reflection problem: Mesh and density contour obtained on 96×24 meshes.

Acknowledgments The research of Li was supported in part by the special funds for Major State Basic Research Projects of China. The research of T. Tang was supported in part by CERG grants of Hong Kong Research Grants Council and International Research Team on Complex System, Chinese Academy of Sciences.

References

- [1] B.N. Azarenok, *Variational barrier method of adaptive grid generation in hyperbolic problems of gas dynamics*, SIAM J. Numer. Anal. **40** (2002), 651–682.
- [2] M. J. Baines, *Moving finite elements*, Oxford University Press, 1994.
- [3] G. Beckett, J. A. Mackenzie, M. L. Robertson, and D. M. Sloan, *On the numerical solutions of one-dimensional pdes using adaptive methods based on equidistribution*, J. Comput. Phys. **167** (2001), 372–392.
- [4] K.S. Bey and J.T. Oden, *hp-version discontinuous Galerkin methods for hyperbolic conservation laws*, Comput. Methods Appl. Mech. Engrg. **133** (1996), 259–286.
- [5] N. Carlson and K. Miller, *Design and application of a gradient-weighted moving finite code, Part II, 2-D*, SIAM J. Sci. Comput. **19** (1998), 766.

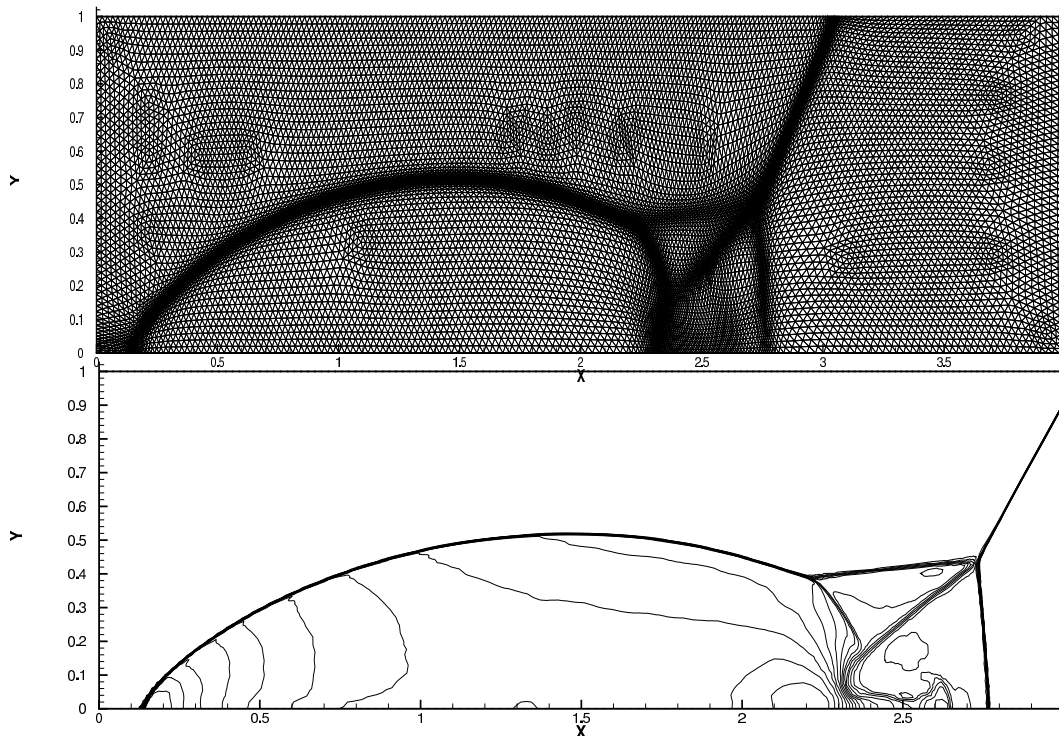


Figure 9: Double Mach reflection problem: Mesh and density contour obtained on 192×48 meshes.

- [6] B. Cockburn, *An introduction to the discontinuous Galerkin method for convection-dominated problems*, Advanced Numerical Approximation of Nonlinear Hyperbolic Equations (B. Cockburn, C. Johnson, C.-W. Shu, and E. Tadmor, eds.), Springer, Berlin, New York, 1998.
- [7] B. Cockburn, G.E. Karniadakis, and C.W. Shu (eds.), *Discontinuous Galerkin methods: Theory, Computation and Applications*, Berlin, Springer, 2000.
- [8] B. Cockburn, F. Li, and C.W. Shu, *Locally divergence-free discontinuous Galerkin methods for the Maxwell equations*, J. Comput. Phys. **197** (2004), 588–610.
- [9] B. Cockburn and C.W. Shu, *The runge-kutta discontinuous galerkin finite element method for conservation laws v: Multidimensional systems*, J. Comput. Phys. **141** (1998), 199–224.
- [10] ———, *Runge-kutta discontinuous Galerkin methods for convection-dominated problems*, J. Sci. Comput. **16** (2001), 173–261.
- [11] Y. Di, R. Li, T. Tang, and P.W. Zhang, *Moving mesh finite element methods for the incompressible Navier-Stokes equations*, to appear in SIAM Journal on Scientific Computing (2003).
- [12] A.S. Dvinsky, *Adaptive grid generation from harmonic maps on Riemannian manifolds*, J. Comput. Phys. **95** (1991), 450–476.

- [13] R. Li, W.-B. Liu, H.-P. Ma, and T. Tang, *Adaptive finite element approximation for distributed elliptic optimal control problems*, SIAM J. Control Optim. **41** (2002), 1321–1349.
- [14] R. Li, T. Tang, and P.-W. Zhang, *A moving mesh finite element algorithm for singular problems in two and three space dimensions*, J. Comput. Phys. **177** (2002), 365–393.
- [15] R. Li, T. Tang, and P.W. Zhang, *Moving mesh methods in multiple dimensions based on harmonic maps*, J. Comput. Phys. **170** (2001), 562–588.
- [16] T. Lu, P.-W. Zhang, and W. Cai, *Discontinuous Galerkin methods for dispersive and lossy Maxwell's equations and PML boundary conditions*, to appear in J. Comput. Phys. (2004).
- [17] H.Z. Tang and T. Tang, *Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws*, SIAM J. Numer. Anal. **41** (2003), 487–515.
- [18] H.Z. Tang, T. Tang, and P.-W. Zhang, *An adaptive mesh redistribution method for nonlinear hamilton-jacobi equations in two- and three dimensions*, J. Comput. Phys. **188** (2003), 543–572.
- [19] P. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, J. Comput. Phys. **54** (1984), 115–173.
- [20] P. A. Zegeling, *On resistive mhd models with adaptive moving meshes*, To appear in J. Sci. Comput. (2004).