Journal of Scientific Computing, Vol. 27, Nos. 1–3, June 2006 (© 2006)
DOI: 10.1007/s10915-005-9045-9

Moving Mesh Discontinuous Galerkin Method for Hyperbolic Conservation Laws

Ruo Li[1] and Tao Tang[2]

Received October 4, 2004; accepted (in revised form) February 22, 2005; Published online January 18, 2006

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 interpolates 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.

KEY WORDS: Moving mesh method; discontinuous Galerkin method; nonlinear conservation laws; monitor function.

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. [8, 11]) and Maxwell equation (see, e.g. [9, 17]). 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

[1] LMAM and School of Mathematical Sciences, Peking University, Peking 100871, P.R. China. E-mail: rli@math.pku.edu.cn
[2] Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong and Institute of Computational Mathematics, Chinese Academy of Sciences, Beijing, China. E-mail: ttang@hkbu.edu.hk

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 [3].

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. [1, 4, 6] and a recent review article [20]). 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. The key ingredients of the moving mesh methods include:

- **Mesh equations.** The mesh equations determine a one-to-one mapping from a regular domain in a parameter space to an irregularly shaped domain in physical space. By connecting points in the physical space corresponding to discrete points in the parameter space, the physical domain can be covered with a computational mesh suitable for the solution of finite difference/element equations. Choosing suitable mesh equations and solving them efficiently are very crucial for an effective moving mesh method;
- **Monitor function.** A monitor function is used to guide the mesh redistribution. It may depend on the solution arclength (in 1D), curvature, and *a posteriori* errors. In practice, local (spatial) smoothing of the monitor function is necessary (see, e.g. [5, 6]);
- **Interpolations.** If the mesh equations are time-dependent and are solved simultaneously with the given differential equations, then interpolation of dependent variables from the old mesh to the new mesh is unnecessary. Otherwise, some kind of interpolation is required to pass the solution information on the old mesh to the newly generated mesh.

In [15], a transformation from a logical domain to the physical domain is constructed by using harmonic mappings (see also [13]). The numerical procedure proposed in [15] has now been extended to solve several classes of problems such as the incompressible Navier–Stokes equations [12] and elliptic optimal control problems [14]. 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://dsec.pku.edu.cn/~rli`.

In this work, the DG method will be employed to solve the hyperbolic conservation laws, i.e., the numerical solution of the underlying PDEs at each time step can be given as long as the mesh is given. Hence, we only need to consider two issues of the three required inputs in the `AFEPack`, 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 can also resolve some weak singularities (such as rarefaction waves). To achieve this goal, 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 by using the moving mesh methods.

The paper is organized as follows. In Sec. 2, the general idea of DG method for conservation law is described and the moving mesh methods will be briefly described. In Sec. 3, several important issues relevant to the moving mesh DG methods will be addressed. Numerical results are given in Sec. 4 to demonstrate the effectiveness of the proposed method.

2. PRELIMINARIES

2.1. DG for Hyperbolic Conservation Laws

In this section, we follow [7, 10] to briefly discuss the DG method for hyperbolic conservation laws. Consider the nonlinear system of conservation laws

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

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

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

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

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

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 which 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

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

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) Lax-Friedrich flux. A limiter strategy is adopted following [7, 10].

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, 16] 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

$$\xi : x \mapsto \xi, \qquad \Omega \to \Omega_c$$

can be obtained by solving the following elliptic system

$$\nabla_x \left(m \nabla_x \xi\right) = 0, \tag{2.5}$$

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

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

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

$$u_h - u_h^{\text{new}} \perp V_h. \tag{2.7}$$

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

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

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

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

The orthogonal property (2.7) is then obtained by subtracting (2.9) from (2.8), which provides a general interpolation formulation. For more specific problems, this general idea can be further studied to obtain more effective formulations, see [12] for an application to incompressible flow simulations. In Sec. 3.2, this general interpolation idea will be applied to obtain a conservative interpolation useful for the moving mesh DG approaches.

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 $\mathcal{O}(1)$:

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

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. On the other hand, using η_K directly will cluster too many points in the shock regions, which is obviously unnecessary. To prevent this from happening, a threshhold $\bar{\eta}$ is used in the monitor function. More precisely, our monitor function is of the form

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

where $\alpha > 0$ is a user-defined parameter. The role 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 $\alpha = 200$). Moreover, the cut-off function $\bar{\eta}$ can be chosen as a constant or the average of the error function η:

$$\bar{\eta} = \frac{1}{|\Omega|} \int_\Omega \eta \, dx. \tag{3.3}$$

A similar averaging idea was also used in [2, 22].

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 $\overrightarrow{\delta x}$ as the node-displacement, which is piecewise linear and global continuous. Then the linear convection PDE is of the form (see [15, 16])

$$\frac{\partial u}{\partial \tau} - \overrightarrow{\delta x} \cdot \nabla u = 0, \tag{3.4}$$

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

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

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

There are certain geometrical requirements for finite elements in designing appropriate limiters for the DG method. However, these requirements may not be valid for meshes generated by the moving mesh methods. Below we will briefly describe the potential difficulty and the relevant method to handle it.

Let us first describe the limiters proposed in [7, 10] on triangle mesh. Assume the neighbors of K_0 are K_1, K_2 and K_3 (see Fig. 1(a)). The numerical solution u_h is piecewise linear on each element. Let C_i, $0 \leqslant i \leqslant 3$, be the barycenter of K_i, and B_j, $1 \leqslant j \leqslant 3$ be the midpoint of the common

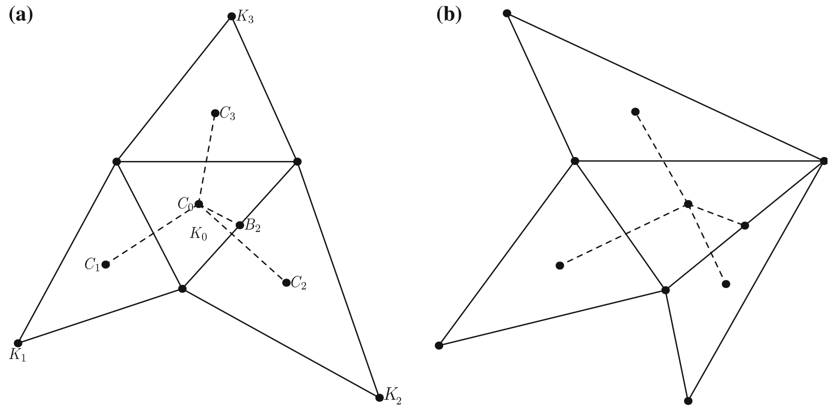

Fig. 1. (a) The limiter proposed in [10]; (b) an example of a not seriously distorted element, for which the limiter proposed in [10] is not applicable.

edge of K_0 and K_j (see Fig. 1(a), where only B_2 is labeled). The mean value of the numerical solution on element K_i, $0 \leqslant i \leqslant 3$ is denoted by $\bar{u}_i$. Then u_h on K_0 is determined by its values at B_j, $1 \leqslant j \leqslant 3$. Assume there are two nonnegative coefficients β_1 and β_2 such that

$$\overrightarrow{C_0B_2} \triangleq \beta_1 \overrightarrow{C_0C_2} + \beta_2 \overrightarrow{C_0C_3}, \tag{3.6}$$

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.

The limiter described above is applicable only if the nonnegative coefficient β_1 and β_2 exist. However, this is not always true as demonstrated in Fig. 1(b). For meshes generated by the moving mesh methods, the situations similar to that in Fig. 1(b) have been observed occasionally. 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:

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

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 $u(x, y; t) = u_0(x - t, y - t)$. Since the solution is very smooth, the monitor function used for this example is (3.2) with a small value of α, $\alpha = 1$. The resulting meshes using 10×10, 20×20, and 40×40 elements are displayed in Fig. 2. It is seen that due to the smoothness of the solution these meshes are almost uniform. The corresponding errors are plotted in Fig. 3 from which a second-order rate of convergence is observed.

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

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

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

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

In Fig. 4, the moving meshes at $t = \pi/4$ obtained by using 20×20 and 40×40 elements are presented. Since the solution of the problem has some typical singular behavior, our moving mesh algorithm takes the action in moving more elements into the areas with large solution gradients. In Fig. 5, numerical errors obtained by using 20×20, 40×40 and 80×80 elements are presented. It is seen that the order of convergence is higher than 2. For example, the error with the 20×20 mesh is about 1/7 of that with

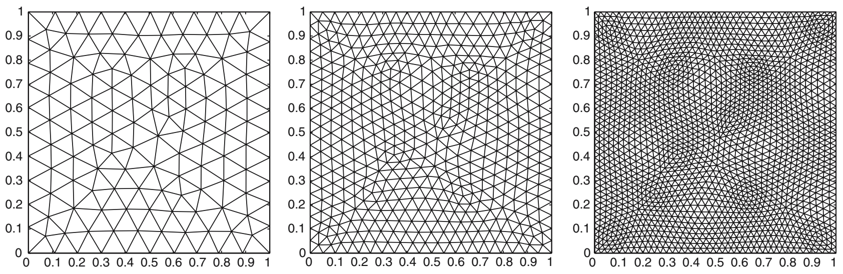

Fig. 2. Example 4.1: meshes obtained with 10×10, 20×20, 40×40 elements.

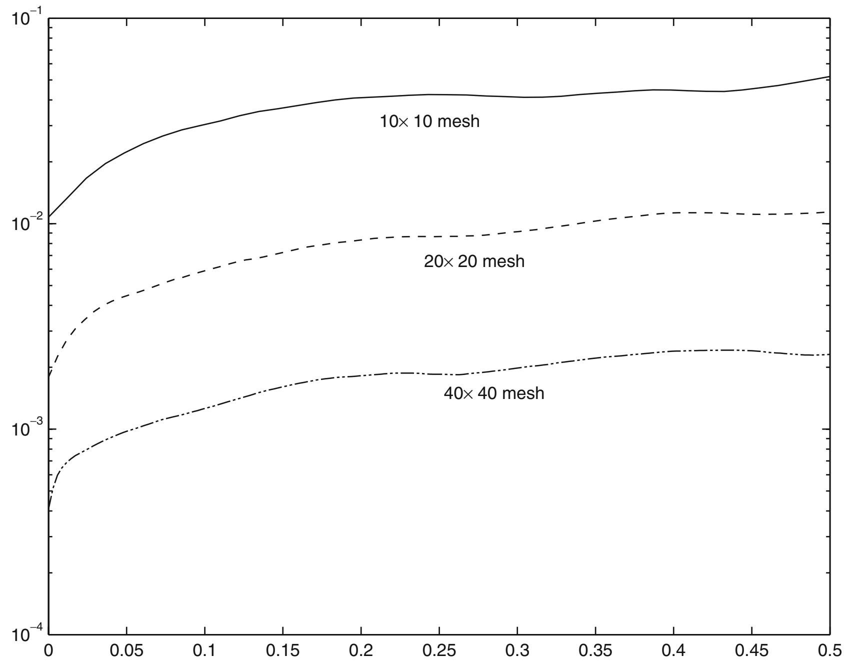

Fig. 3. Example 4.1: numerical errors obtained using three different meshes.

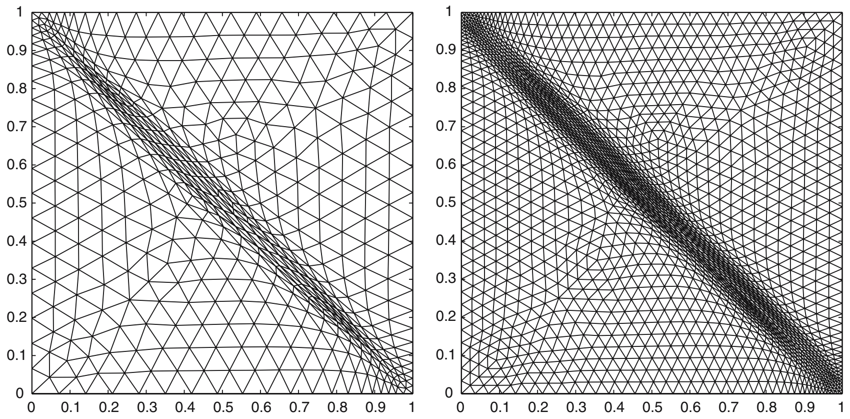

Fig. 4. Example 4.2: moving meshes using 20×20 and 40×40 elements.

the 40×40 mesh. The order enhancement for the moving mesh method to problems with large gradients has been observed for many other computations (see, e.g. [19]).

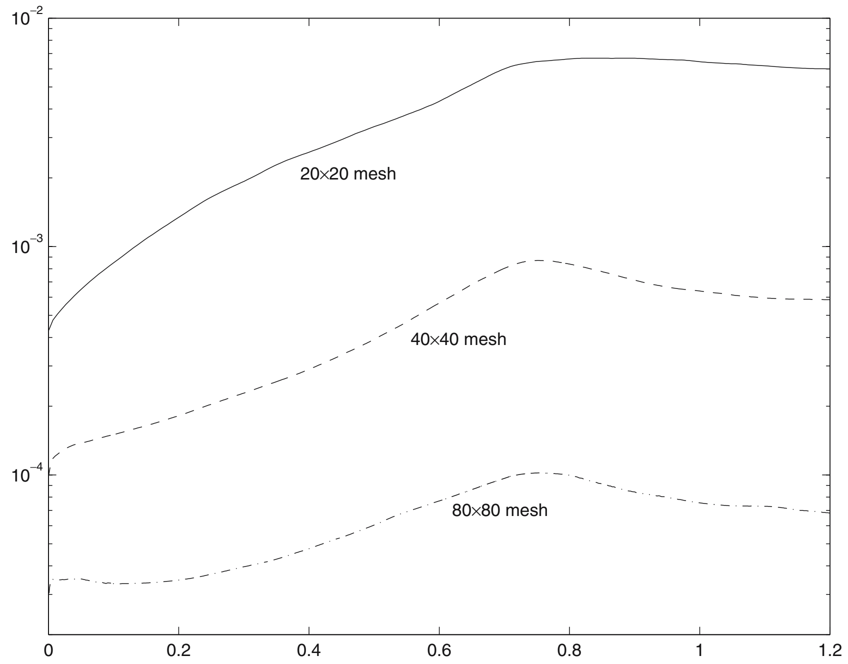

Fig. 5. Example 4.2: numerical errors obtained using three different meshes.

It is seen from Fig. 4 that the strength of mesh concentration along the *shock wave* is not uniform. The mesh near the corner looks a little different from that in the center of the domain. This is caused by the boundary grid redistribution, since the moving mesh method we currently adopted couples the interior and boundary grids. In the iteration part, we adopt an algebraic multigrid (AMG) method to solve (2.5). We generally require about three AMG iterations to achieve a tolerance at about 10^{-12}.

Unlike Example 4.1, the solution of this problem has large solution gradients as seen in Fig. 4. As a result, the monitor function used for this example is (3.2) with a large value of α. For this and the remaining numerical examples, we set $\alpha = 200$.

In both of the above two 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

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

where

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

The state relation is given by

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

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

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

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

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

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 $(u\,v)\cdot n_{e,K_1}\pm c$ and $(u\,v)\cdot n_{e,K_1}$.

Example 4.3 (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

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

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

The cut-off threshhold $\bar{\eta}$ is chosen a large constant 10^2. The parameter α for this example is also chosen as 10^2. Figure 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. Figure 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. [18]).

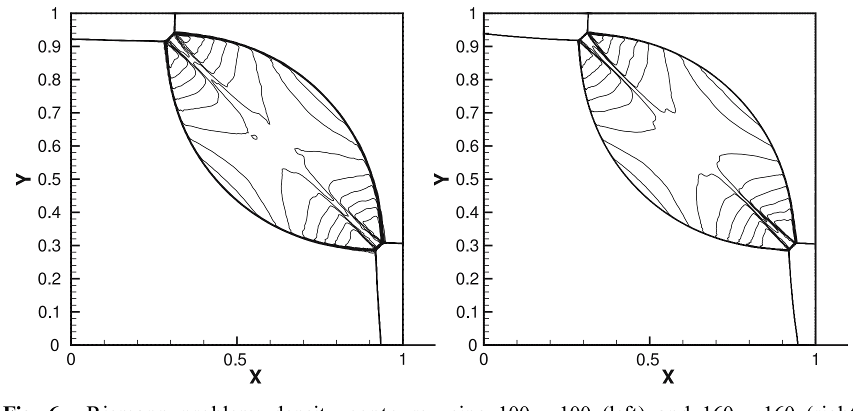

Fig. 6. Riemann problem: density contours using 100×100 (left) and 160×160 (right) elements.

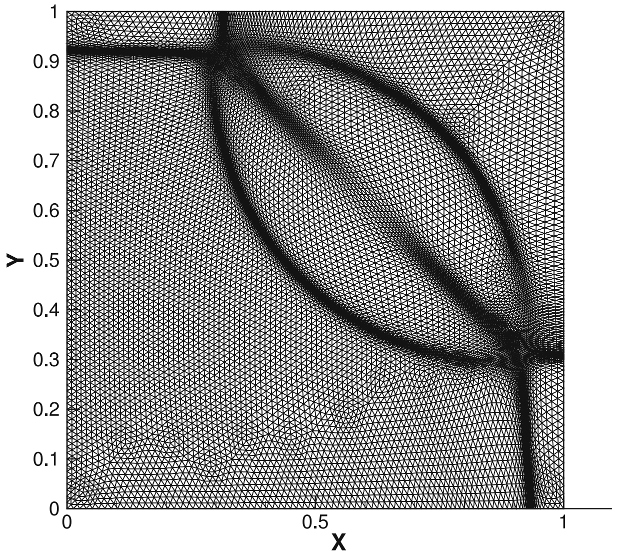

Fig. 7. Riemann problem: moving mesh with 100×100 elements.

Example 4.4 (Double Mach reflection problem). 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° 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.

This problem was studied by Woodward and Colella [21] and has been used as a standard test problem since then. The DG results obtained on fixed grids can be found in [8, 11], and the moving mesh results can be found in [18]. The cutoff threshhold $\bar{\eta}$ 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 $t = 0.2$. 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 more elements 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.

The results for Examples 4.3 and 4.4 have been obtained in [18] using the MUSCL finite volume discretization. The results (on meshes

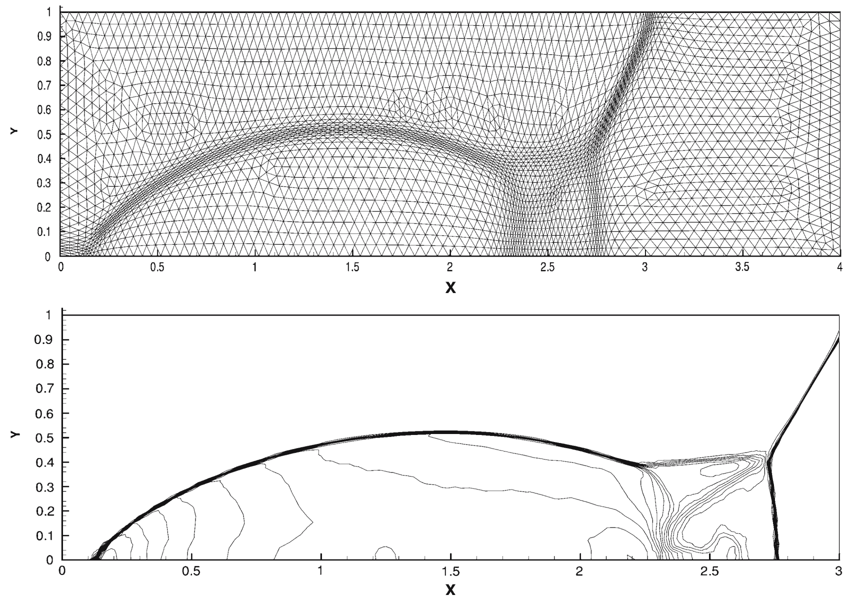

Fig. 8. Double Mach reflection problem: mesh and density contour obtained on 96×24 meshes.

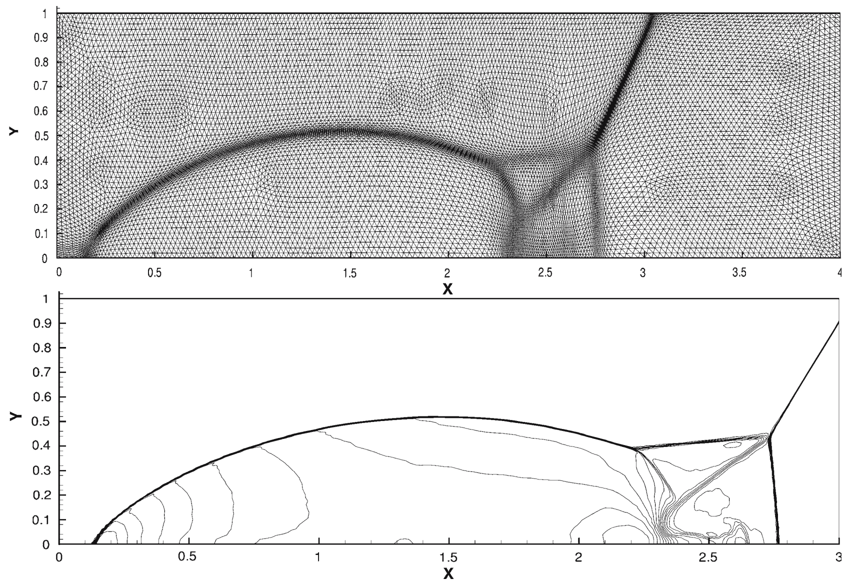

Fig. 9. Double Mach reflection problem: mesh and density contour obtained on 192×48 meshes.

and solutions) reported here are better than those presented in [18]. In particular, the quality of meshes obtained in this subsection is higher, and consequently more detailed structures of the singular solution can be resolved. This is partly due to the use of the more reasonable monitor functions. The use of the DG discretization also improves the solution quality due to the advantages of the DG method for discontinuous problems (as mentioned in the first section).

ACKNOWLEDGMENTS

The research of Li was supported in part by the special funds for Major State Basic Research Projects of China, and the Joint Research Institute of Applied Mathematics between Peking University and Hong Kong Baptist University. The research of T. Tang was supported in part by CERG grants of Hong Kong Research Grants Council, NSAF Grant 10476032 of China, and International Research Team on Complex System, Chinese Academy of Sciences.

REFERENCES

1. Azarenok, B. N. (2002). Variational barrier method of adaptive grid generation in hyperbolic problems of gas dynamics. *SIAM J. Numer. Anal.* **40**, 651–682.
2. Beckett, G., Mackenzie, J. A., Robertson, M. L., and Sloan, D. M. (2001). On the numerical solutions of one-dimensional PDEs using adaptive methods based on equidistribution. *J. Comput. Phys.* **167**, 372–392.
3. Bey, K. S., and Oden, J. T., (1996). hp-version discontinuous Galerkin methods for hyperbolic conservation laws. *Comput. Methods Appl. Mech. Engrg.* **133**, 259–286.
4. Cao, W. M., Huang, W.-Z., and Russell, R. D. (1999). An r-adaptive finite element method based upon moving mesh PDEs. *J. Comput. Phys.* **149**, 221–244.
5. Cao, W. M., Huang, W.-Z., and Russell, R. D. (1999). A study of monitor functions for two dimensional adaptive mesh generation. *SIAM J. Sci. Comput.* **20**, 1978–1994.
6. Cao, W. M., Huang, W.-Z., and Russell, R. D. (2001). An error indicator monitor function for an r-adaptive finite-element method. *J. Comput. Phys.* **170**, 871–892.
7. Cockburn, B. (1998). An introduction to the discontinuous Galerkin method for convection-dominated problems. In Cockburn, B., Johnson, C., Shu, C.-W., and Tadmor, E. (eds.), *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, Springer, Berlin, New York.
8. Cockburn, B., Karniadakis, G. E., and Shu, C. W. (eds.) (2000). *Discontinuous Galerkin Methods: Theory, Computation and Applications*, Springer, Berlin.
9. Cockburn, B., Li, F., and Shu, C. W. (2004). Locally divergence-free discontinuous Galerkin methods for the Maxwell equations. *J. Comput. Phys.* **197**, 588–610.
10. Cockburn, B., and Shu, C. W. (1998). The Runge–Kutta discontinuous Galerkin finite element method for conservation laws V: Multidimensional systems. *J. Comput. Phys.* **141**, 199–224.
11. Cockburn, B., and Shu, C. W. (2001). Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. *J. Sci. Comput.* **16**, 173–261.

12. Di, Y., Li, R., Tang, T., and Zhang, P. W. (2005). Moving mesh finite element methods for the incompressible Navier–Stokes equations. *SIAM J. Sci. Comput.* **26**, 1036–1056.
13. Dvinsky, A. S. (1991). Adaptive grid generation from harmonic maps on Riemannian manifolds. *J. Comput. Phys.* **95**, 450–476.
14. Li, R., Liu, W.-B., Ma, H.-P., and Tang, T. (2002). Adaptive finite element approximation for distributed elliptic optimal control problems. *SIAM J. Control Optim.* **41**, 1321–1349.
15. Li, R., Tang, T., and Zhang, P.-W. (2001). Moving mesh methods in multiple dimensions based on harmonic maps. *J. Comput. Phys.* **170**, 562–588.
16. Li, R., Tang, T., and Zhang, P.-W. (2002). A moving mesh finite element algorithm for singular problems in two and three space dimensions. *J. Comput. Phys.* **177**, 365–393.
17. Lu, T., Zhang, P.-W., and Cai, W. (2004). Discontinuous Galerkin methods for dispersive and lossy Maxwell's equations and PML boundary conditions. To appear in *J. Comput. Phys.*
18. Tang, H. Z., and Tang, T. (2003). Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws. *SIAM J. Numer. Anal.* **41**, 487–515.
19. Tang, H. Z., Tang, T., and Zhang, P.-W. (2003). An adaptive mesh redistribution method for nonlinear Hamilton-Jacobi equations in two- and three dimensions. *J. Comput. Phys.* **188**, 543–572.
20. Tang, T. (2005). Moving mesh methods for computational fluid dynamics. In Shi, Z.-C., Chen, Z., Tang, T. and Yu, D. (eds.), *Recent Advances in Adaptive Computations (Providence, USA)* Contemporary Mathematics, American Mathematical Society.
21. Woodward, P., and Colella, P. (1984). The numerical simulation of two-dimensional fluid flow with strong shocks. *J. Comput. Phys.* **54**, 115–173.
22. Zegeling, P. A. (2005). On resistive MHD models with adaptive moving meshes. To appear in *J. Sci. Comput.* **24**, 263–284.