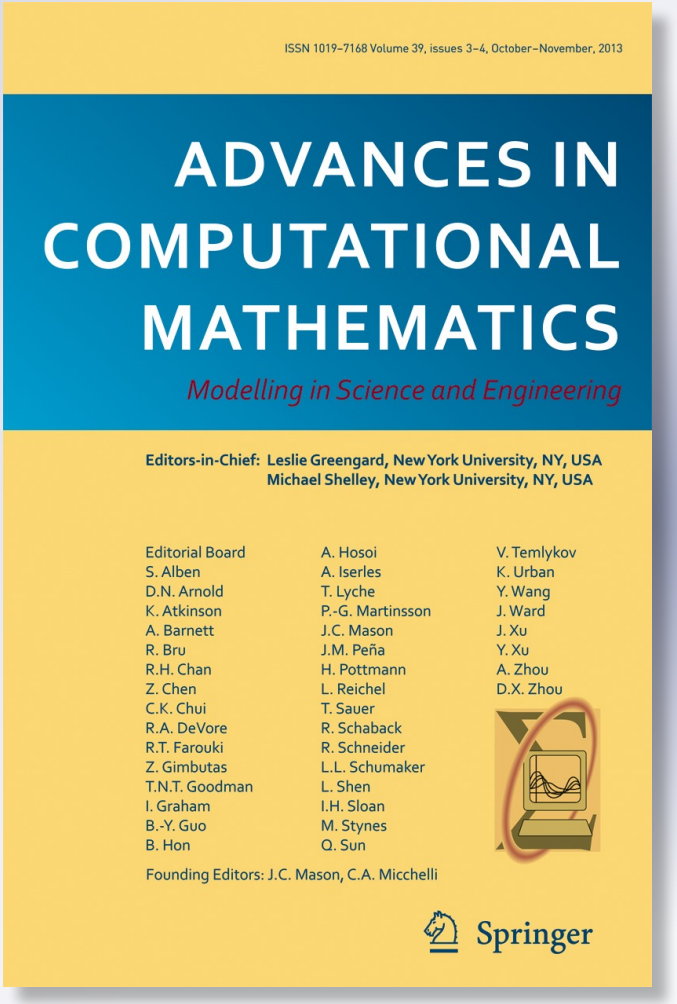
An h-adaptive RKDG method with troubled-cell indicator for two-dimensional hyperbolic conservation laws

Hongqiang Zhu & Jianxian Qiu

Advances in Computational Mathematics
Modelling in Science and Engineering

ISSN 1019-7168
Volume 39
Combined 3-4

Adv Comput Math (2013) 39:445-463
DOI 10.1007/s10444-012-9287-7



Springer

Adv Comput Math (2013) 39:445–463
DOI 10.1007/s10444-012-9287-7

An *h*-adaptive RKDG method with troubled-cell indicator for two-dimensional hyperbolic conservation laws

Hongqiang Zhu · Jianxian Qiu

Received: 23 December 2011 / Accepted: 5 November 2012 /
Published online: 30 November 2012

Abstract In Zhu and Qiu (J Comput Phys 228:6957–6976, 2009), we systematically investigated adaptive Runge-Kutta discontinuous Galerkin (RKDG) methods for hyperbolic conservation laws with different indicators, with an objective of obtaining efficient and reliable indicators to obtain better performance for adaptive computation to save computational cost. In this follow-up paper, we extend the method to solve two-dimensional problems. Although the main idea of the method for two-dimensional case is similar to that for one-dimensional case, the extension of the implementation of the method to two-dimensional case is nontrivial because of the complexity of the adaptive mesh with hanging nodes. We lay our emphasis on the implementation details including adaptive procedure, solution projection, solution reconstruction and troubled-cell indicator. Extensive numerical experiments are presented to show the effectiveness of the method.

Keywords Runge-Kutta discontinuous Galerkin method · Troubled-cell indicator · *h*-adaptive method

Communicated by: Ben Yu Guo.

The research was partially supported by NSFC grant 10931004, 11126287, 11201242, NJUPT grant NY211029 and ISTCP of China grant No. 2010DFR00700.

H. Zhu
School of Natural Science, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210046, People's Republic of China
e-mail: hongqiang.zhu@gmail.com

J. Qiu (✉)
School of Mathematical Sciences, Xiamen University, Xiamen, Fujian 361005, People's Republic of China
e-mail: jxqiu@xmu.edu.cn

Mathematics Subject Classifications (2010) 65M60 · 65M99 · 35L65

1 Introduction

In [22], we presented an *h*-adaptive Runge-Kutta discontinuous Galerkin (RKDG) method using troubled-cell indicators for one-dimensional hyperbolic conservation laws. Numerical tests demonstrated the effectiveness of the method in saving computational cost and improving solution quality when proper troubled-cell indicators were used. In this paper, we extend the method to solve the two-dimensional nonlinear hyperbolic systems of conservation laws

$$\begin{cases} u_t + f(u)_x + g(u)_y = 0, & \text{in } \Omega \times (0, T], \\ u(x, y, 0) = u_0(x, y), & \text{in } \Omega. \end{cases} \tag{1.1}$$

Here, the subscripts t, x and y denote partial differentiation with respect to time and the spatial coordinates, and u, u_0, f and g are m-vectors on the problem domain $\Omega \times [0, T]$.

A major development of the RKDG method for nonlinear time-dependent hyperbolic conservation laws was carried out by Cockburn et al. in a series of papers [3–7], in which a framework to solve these problems was established. They adopted explicit, nonlinearly stable high order Runge-Kutta time discretizations [19], DG space discretizations with exact or approximate Riemann solvers as interface fluxes and TVB (total variation bounded) nonlinear limiter [18] to achieve nonoscillatory properties, and the method managed the advantage of flexibility in handling complicated geometry, *h-p* adaptivity, and efficiency of parallel implementation. We will briefly review this method in Section 2. Detailed description of the method as well as its implementation can be found in the review paper [8].

The solutions of nonlinear hyperbolic conservation laws often exhibit a wide range of localized structures, such as shock waves, contact discontinuities, and rarefaction waves. So it is desirable to use mesh refinement to cluster grid cells in regions where they are most needed, for example, around shocks or other regions where the solution has steep gradients or complicated structures. For time-dependent problems this should be done in an adaptive manner. The refined region has to move adaptively with the interesting structure.

The DG method is a class of finite element methods using completely discontinuous piecewise polynomial space for the numerical solution and the test functions, which makes it easy to handle *h*-adaptive strategies since the mesh refining or coarsening can be done without taking into account the continuity restrictions through cell interfaces. To the best of our knowledge only a few *h*-adaptive DG methods are available in literature for nonlinear time-dependent hyperbolic conservation laws. We refer, for instance, to Flaherty et al. [1, 10, 11, 17] and Dedner et al. [9]. We also refer to Hartmann and Houston [12], where duality techniques were used for designing adaptive strategy. The key point of the success of *h*-adaptive methods is the indicator to identify

where the mesh should be refined and coarsened. In this paper, we use the same h-adaptive strategy as in [22], i.e. use the 'troubled-cell' indicator to identify where the mesh should be refined and coarsened. For DG method, 'troubled cells' are those cells which might need the limiting procedure; this also means that discontinuities might appear in these cells. So a straightforward strategy for the mesh adaptation is to refine the troubled cells and coarsen the others. The detailed algorithm will be described in Section 3.1.

The present paper is organized as follows. Firstly in Section 2, we give a brief review of RKDG method in two space dimensions. Then in Section 3, we show the two-dimensional h-adaptive RKDG method and the implementation details. After that in Section 4, we present a series of numerical results to validate our adaptive algorithm. Finally, we give our concluding remarks in Section 5.

2 Review of RKDG method in two space dimensions

To define the RKDG method, we first discretize (1.1) in space using the discontinuous Galerkin method. For simplicity, we do that for u being a scalar ($m = 1$). If u is vector-valued, one simply proceeds in component way.

Given a triangulation $\mathcal{T}_h$ of the domain Ω, we seek the approximate solution $u_h(t)$ in the finite element space of discontinuous functions

$$V_h^k = \{v_h \in L^\infty(\Omega) : v_h|_K \in \mathbb{P}^k, \forall K \in \mathcal{T}_h\}.$$

We multiply (1.1) by a test function $v(x, y) \in V_h^k$, integrate over cell K, and integrate by parts:

$$\frac{d}{dt}\int_K u(x, y, t)v(x, y)dxdy - \int_K F(u) \cdot \nabla v dxdy + \sum_{e\in\partial K}\int_e F(u) \cdot n_{e,K} v ds = 0 \tag{2.1}$$

where $F = (f, g)$, and $n_{e,K}$ is the outward unit normal to the edge e. The volume integral term $\int_K F(u) \cdot \nabla v dxdy$ can be computed either exactly or by a numerical quadrature of sufficiently high order of accuracy. The line integral in (2.1) is typically discretized by a Gaussian quadrature with sufficient accuracy

$$\int_e F(u) \cdot n_{e,K} v ds \approx |e| \sum_{l=1}^{q} \omega_l F(u(G_l, t)) \cdot n_{e,K} v(G_l)$$

where $F(u(G_l, t)) \cdot n_{e,K}$ is replaced by a monotone numerical flux. In this paper, we use the simple Lax-Friedrichs flux

$$F(u(G_l, t)) \cdot n_{e,K} \approx \frac{1}{2}\left[(F(u^-(G_l, t)) + F(u^+(G_l, t))) \cdot n_{e,K} - \alpha(u^+(G_l, t) - u^-(G_l, t))\right]$$

where α is taken as an upper bound for the eigenvalues of the Jacobian in the direction of $n_{e,K}$, and u^- and u^+ are the values of u inside and outside the cell K at the Gaussian point G_l.

The semidiscrete scheme (2.1) is an ODE system. One discretizes it using the total variation diminishing (TVD) Runge-Kutta time discretization introduced in [19], which completes the definition of RKDG method.

3 Algorithm and implementation details

In this section we give the two-dimensional h-adaptive algorithm along with its implementation details. We extend the one-dimensional h-adaptive RKDG method in [22] to the two-dimensional case. The idea is similar for the two-dimensional case. We still use the troubled-cell indicators to identify the troubled cells and adapt the mesh by refining the troubled cells and coarsening the others. But there is a big difference: hanging nodes are inevitable in the two-dimensional local mesh refinement.

Note that we have not imposed any restrictions to the triangulation $\mathcal{T}_h$. In fact, the RKDG procedure described in Section 2 is totally consistent with irregular meshes, including meshes with hanging nodes. In this paper we adopt rectangular meshes. So we restrict $\mathcal{T}_h$ to rectangular mesh with hanging nodes permitted, and restrict Ω to a domain that can be partitioned into uniform rectangles.

Although the main idea of the method for two-dimensional case is similar to that for one-dimensional case, the extension of the implementation of the method to two-dimensional case is nontrivial because of the complexity of the adaptive mesh with hanging nodes. In what follows we first give the algorithm and then discuss the implementation issues.

3.1 Algorithm

The following flowchart illustrates the two-dimensional h-adaptive RKDG method.

ALGORITHM Given the maximum refinement level LEV and the final time T,

- Step 1. Partition the domain into uniform rectangles, compute the degrees of freedom $\{u_K^{(l)}(t_0)\}$ from the initial data $u_0(x, y)$, and set all cells' initial mesh level $\{lev_K(t_0)\}$ to be 0.
- Step 2. Suppose we have known the mesh $\mathcal{T}_h(t_n)$, the mesh level $\{lev_K(t_n)\}$ and the degrees of freedom $\{u_K^{(l)}(t_n)\}$ at time level t_n. Do the troubled-cell indicator procedure, mark every cell as a troubled cell or an untroubled cell.
  - For a troubled cell K, if its mesh level $lev_K(t_n) = LEV$, do nothing. If $lev_K(t_n) < LEV$, divide it into four uniform rectangles and increase the four new cells' mesh level by one.
  - For four untroubled cells which come from one dividing at some earlier time level, merge them and decrease the new cell's mesh level by one.

Now we get the new mesh $\mathcal{T}_h(t_{n+1})$ and the corresponding new mesh level $\{lev_K(t_{n+1})\}$.

- Step 3. Using L_2 projection, project the degrees of freedom $\{u_K^{(l)}(t_n)\}$ on the mesh $\mathcal{T}_h(t_n)$ to the new mesh $\mathcal{T}_h(t_{n+1})$.
- Step 4. Evolve the solution from t_n to t_{n+1} by the RKDG procedure and get $\{u_K^{(l)}(t_{n+1})\}$ on the mesh $\mathcal{T}_h(t_{n+1})$.
- Step 5. If $t_{n+1} < T$, go to Step 2.

3.2 Basis and L_2 projection

In order to simplify the implementation and calculation, we form the following orthogonal basis for V_h^k

$$v_{\frac{i(i+1)}{2}+j}(x, y) = W_{i-j}(x)W_j(y), \quad i = 0, \ldots, k, \quad j = 0, \ldots, i,$$

which are tensor products of one-dimensional orthogonal basis of Legendre polynomials

$$\begin{cases} W_0(x) = 1, \\ W_l(x) = \dfrac{1}{2^l l!}\dfrac{d^{\,l}(x^2-1)^l}{dx^l}, \quad l > 0. \end{cases}$$

Then the local orthogonal basis over cell K is given by

$$v_l^{(K)}(x, y) = v_l\left(\frac{2(x - x_K)}{\Delta x_K}, \frac{2(y - y_K)}{\Delta y_K}\right), \quad l = 0, ..., Q_k,$$

in which $Q_k = k(k + 3)/2$, (x_K, y_K) is the center of rectangle K, and Δx_K and Δy_K are lengths of K's sides in the direction of x and y respectively. Now the numerical solution $u_h(x, t)$ in the space V_h^k can be expressed as

$$u_h(x, y, t)|_K = \sum_{l=0}^{Q_k} u_K^{(l)}(t)v_l^{(K)}(x, y) \tag{3.1}$$

where $u_K^{(l)}(t)(l = 0, ..., Q_k)$ are the degrees of freedom. Particularly, $u_K^{(0)}(t)$ is the cell average of u_h over K.

Let us now deduce the formulas of L_2 projection. Suppose we have already known u_h on mesh $\mathcal{T}_h(t_n)$, and we need to determine the degrees of freedom $u_{K'}^{(l)}(t_n)(l = 0, ..., Q_k)$ in the new cell $K' \in \mathcal{T}_h(t_{n+1})$. Let $u_h^{'}$ denote the L_2 projection of u_h, it should satisfy the following equation

$$\int_{K'} u_h^{'}|_{K'} v_l^{(K')}(x, y)dxdy = \int_{K'} u_h v_l^{(K')}(x, y)dxdy, \quad l = 0, ..., Q_k.$$

Represent $u_h^{'}|_{K'}$ using (3.1) and change the integral variables to get

$$u_{K'}^{(l)}(t_n) = \frac{4}{b_l \Delta x_{K'} \Delta y_{K'}} \int_{K'} u_h v_l^{(K')}(x, y) dxdy, \quad l = 0, \ldots, Q_k \tag{3.2}$$

where

$$b_l = \int_{-1}^{1} \int_{-1}^{1} (v_l(x, y))^2 dxdy, \quad l = 0, \ldots, Q_k$$

are constants. For u_h is a piecewise polynomial, the integral in (3.2) can be computed exactly.

Now we are ready to give the formulas of L_2 projection. When four cells K_1, K_2, K_3, K_4 are merged to a new cell K' (see the left sketch in Fig. 1), the new degrees of freedom computed by (3.2) are as follows when $k = 2$ (for simplicity we drop the time variable).

$$\begin{aligned}
u_{K'}^{(0)} &= \frac{1}{4}\left(u_{K_1}^{(0)} + u_{K_2}^{(0)} + u_{K_3}^{(0)} + u_{K_4}^{(0)}\right), \\
u_{K'}^{(1)} &= \frac{3}{8}\left(-u_{K_1}^{(0)} + u_{K_2}^{(0)} - u_{K_3}^{(0)} + u_{K_4}^{(0)}\right) + \frac{1}{8}\left(u_{K_1}^{(1)} + u_{K_2}^{(1)} + u_{K_3}^{(1)} + u_{K_4}^{(1)}\right), \\
u_{K'}^{(2)} &= \frac{3}{8}\left(-u_{K_1}^{(0)} - u_{K_2}^{(0)} + u_{K_3}^{(0)} + u_{K_4}^{(0)}\right) + \frac{1}{8}\left(u_{K_1}^{(2)} + u_{K_2}^{(2)} + u_{K_3}^{(2)} + u_{K_4}^{(2)}\right), \\
u_{K'}^{(3)} &= \frac{5}{16}\left(-u_{K_1}^{(1)} + u_{K_2}^{(1)} - u_{K_3}^{(1)} + u_{K_4}^{(1)}\right) + \frac{1}{16}\left(u_{K_1}^{(3)} + u_{K_2}^{(3)} + u_{K_3}^{(3)} + u_{K_4}^{(3)}\right), \\
u_{K'}^{(4)} &= \frac{9}{16}\left(u_{K_1}^{(0)} - u_{K_2}^{(0)} - u_{K_3}^{(0)} + u_{K_4}^{(0)}\right) + \frac{3}{16}\left(-u_{K_1}^{(1)} - u_{K_2}^{(1)} + u_{K_3}^{(1)} + u_{K_4}^{(1)}\right) \\
&\quad + \frac{3}{16}\left(-u_{K_1}^{(2)} + u_{K_2}^{(2)} - u_{K_3}^{(2)} + u_{K_4}^{(2)}\right) + \frac{1}{16}\left(u_{K_1}^{(4)} + u_{K_2}^{(4)} + u_{K_3}^{(4)} + u_{K_4}^{(4)}\right), \\
u_{K'}^{(5)} &= \frac{5}{16}\left(-u_{K_1}^{(2)} - u_{K_2}^{(2)} + u_{K_3}^{(2)} + u_{K_4}^{(2)}\right) + \frac{1}{16}\left(u_{K_1}^{(5)} + u_{K_2}^{(5)} + u_{K_3}^{(5)} + u_{K_4}^{(5)}\right).
\end{aligned}$$

For $k = 1$, only the first three formulas are needed.

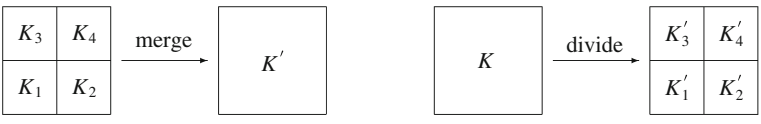


Fig. 1 Sketches of merging *(left)* and dividing *(right)* in the adaptive mesh

When a cell K is divided into four subcells $K_1^{'}$, $K_2^{'}$, $K_3^{'}$, $K_4^{'}$ (see the right sketch in Fig. 1), the new degrees of freedom for $k = 2$ can be computed by setting

$$\begin{aligned}
K_1^{'}: &\quad K' \to K_1^{'}, \quad \lambda_{1x} = -1/4, \quad \lambda_{1y} = -1/4, \quad \lambda_{2x} = 1/2, \quad \lambda_{2y} = 1/2,\\
K_2^{'}: &\quad K' \to K_2^{'}, \quad \lambda_{1x} = 1/4, \quad \lambda_{1y} = -1/4, \quad \lambda_{2x} = 1/2, \quad \lambda_{2y} = 1/2,\\
K_3^{'}: &\quad K' \to K_3^{'}, \quad \lambda_{1x} = -1/4, \quad \lambda_{1y} = 1/4, \quad \lambda_{2x} = 1/2, \quad \lambda_{2y} = 1/2,\\
K_4^{'}: &\quad K' \to K_4^{'}, \quad \lambda_{1x} = 1/4, \quad \lambda_{1y} = 1/4, \quad \lambda_{2x} = 1/2, \quad \lambda_{2y} = 1/2
\end{aligned}$$

in

$$\begin{aligned}
u_{K'}^{(0)} &= u_K^{(0)} + 2\lambda_{1x}u_K^{(1)} + 2\lambda_{1y}u_K^{(2)} + \left(6\lambda_{1x}^2 + \frac{1}{2}\lambda_{2x}^2 - \frac{1}{2}\right)u_K^{(3)}\\
&\quad + 4\lambda_{1x}\lambda_{1y}u_K^{(4)} + \left(6\lambda_{1y}^2 + \frac{1}{2}\lambda_{2y}^2 - \frac{1}{2}\right)u_K^{(5)},\\
u_{K'}^{(1)} &= \lambda_{2x}\left(u_K^{(1)} + 6\lambda_{1x}u_K^{(3)} + 2\lambda_{1y}u_K^{(4)}\right),\\
u_{K'}^{(2)} &= \lambda_{2y}\left(u_K^{(2)} + 2\lambda_{1x}u_K^{(4)} + 6\lambda_{1y}u_K^{(5)}\right),\\
u_{K'}^{(3)} &= \lambda_{2x}^2 u_K^{(3)},\\
u_{K'}^{(4)} &= \lambda_{2x}\lambda_{2y}u_K^{(4)},\\
u_{K'}^{(5)} &= \lambda_{2y}^2 u_K^{(5)}
\end{aligned} \tag{3.3}$$

where

$$\lambda_{1x} = \frac{x_{K'} - x_K}{\Delta x_K}, \qquad \lambda_{2x} = \frac{\Delta x_{K'}}{\Delta x_K},$$
$$\lambda_{1y} = \frac{y_{K'} - y_K}{\Delta y_K}, \qquad \lambda_{2y} = \frac{\Delta y_{K'}}{\Delta y_K}.$$

For $k = 1$, one just sets $u_K^{(3)} = u_K^{(4)} = u_K^{(5)} = 0$ in (3.3).

3.3 Solution reconstruction

For solving conservation laws with strong shocks in the solutions, a nonlinear limiter is required in the RKDG method to detect discontinuities and control spurious oscillations near such discontinuities. The limiter is composed of two parts: one is to detect the discontinuous regions, the other is the solution reconstruction which is to control the oscillations. Similar to [22], in this paper troubled-cell indicators are used to detect the discontinuous regions as the first part of limiter, and are also used to control mesh adaptation.

For the second part of limiter, the one-dimensional WENO (weighted essentially nonoscillatory) type reconstruction described in [22] is hard to be extended for our two-dimensional algorithm because of the complexity of

the two-dimensional adaptive mesh with hanging nodes. Specifically, different cells may have different numbers of neighbors, different neighbors of a cell may have different sizes and a cell may have different neighbors at different time levels. The mesh varies not only in time, but also in space. And WENO methodology is mesh dependent, which is why it is hard for us to design inexpensive WENO based solution reconstruction. However, we find an easy way to solve this problem. If a cell K needs solution reconstruction, we build a local uniform mesh which consists of $(2k+1) \times (2k+1)$ cells with K in the center. See an example in Fig. 2. We use these imaginary cells instead of the real cells to reconstruct the solution in K. Since the imaginary local mesh is uniformly rectangular, the WENO solution reconstruction introduced by Qiu and Shu in [16] can be applied directly. The only problem left is to compute the cell averages in the imaginary local mesh. This can be done easily for we have already got the L_2 projection formula (3.2).

Since both of the initial mesh and the imaginary local mesh are uniformly rectangular, a cell $\tilde{K}$ in the imaginary local mesh coincides with (i) a real cell (denoted by K), or (ii) a subcell of a real cell, or (iii) a union of several real cells (denoted by $K_1, \ldots, K_m$). In case (i), the cell averages of $\tilde{K}$ are equal to the cell averages of K. In case (ii), the cell averages of $\tilde{K}$ can be computed by the first formula in (3.3). In case (iii), from (3.2) we can derive $u_{\tilde{K}}^{(0)} = \left(\sum_{i=1}^{m} u_{K_i}^{(0)} S_{K_i}\right) / S_{\tilde{K}}$ where S_{K_i} and $S_{\tilde{K}}$ denote the area of K_i and $\tilde{K}$ respectively.

3.4 Troubled-cell indicators

Qiu and Shu gave a considerably detailed review of various troubled-cell indicators in [15]. These troubled-cell indicators were used by us in [22] to design the one-dimensional h-adaptive RKDG method, and we compared the performance of these indicators. The numerical tests in [22] showed that the

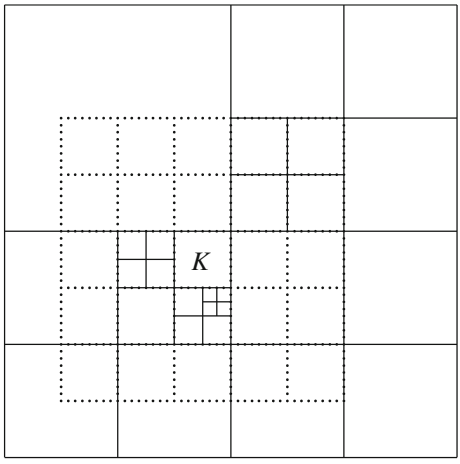


Fig. 2 The imaginary local uniform mesh of K (dotted lines) when $k = 2$

indicator based on the shock-detection technique introduced by Krivodonova et al. [13] (we termed it as KXRCF indicator) was the most efficient and reliable. In this paper, we would like to focus on this indicator.

The KXRCF troubled-cell indicator works in the following way. Partition the boundary of a cell K into two portions ∂K^- and ∂K^+, where the flow is into ($\vec{v}\cdot\vec{n}<0$, $\vec{n}$ is the normal vector to ∂K) and out of ($\vec{v}\cdot\vec{n}>0$) cell K, respectively. Cell K is identified as a troubled cell, if

$$\frac{\left|\int_{\partial K^-}(u^h|_K - u^h|_{K_n})ds\right|}{h_K^{\frac{k+1}{2}}\left|\partial K^-\right|\|u^h|_K\|} > 1$$

where h_K is the radius of the circumscribed circle in cell K, K_n is the neighbor of K on the side of ∂K^- and the norm is based on the maximum norm taken at the integration quadrature points.

4 Numerical results

In this section we provide a series of numerical examples to illustrate the good behavior of the two-dimensional h-adaptive RKDG method described in Section 3. Attention has not been paid to the issue of time discretization efficiency, so global time steps are used in Runge-Kutta method, which are proportional to the smallest cell size at each time level. Study of local time-stepping scheme for the method is the subject of future work.

In all examples, we only plot the results obtained with a particular choice of initial mesh and with $LEV = 4$ because of space limitation. We have verified with the aid of successive refinements of initial mesh, that in all cases, the approximations are numerically convergent.

Example 4.1 In this first example we consider the nonlinear scalar Burgers equation in two dimensions

$$u_t + \left(\frac{u^2}{2}\right)_x + \left(\frac{u^2}{2}\right)_y = 0$$

with the following initial conditions (ICs)

$$\text{IC-1:}\quad u(x,y,0) = \begin{cases} 0.1, & x>0,\ y>0,\\ 2.5, & x<0,\ y>0,\\ 1.1, & x<0,\ y<0,\\ 1.5, & x>0,\ y<0, \end{cases}$$

$$\text{IC-2:}\quad u(x,y,0) = \begin{cases} 1.1, & x>0,\ y>0,\\ 3.1, & x<0,\ y>0,\\ 2.1, & x<0,\ y<0,\\ 0.1, & x>0,\ y<0. \end{cases}$$

We plot the contours and mesh at $t = 0.8$ in Figs. 3 and 4. These figures clearly indicate that almost all the mesh refinements are at the discontinuities and our adaptive strategy is effective in identifying and following important features such as shocks. We use very few cells and get very sharp shocks.

Example 4.2 Two-dimensional Euler equations for Riemann problem [2, 14]. We solve the two-dimensional Euler equations of gas dynamics

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{pmatrix}=0 \tag{4.1}$$

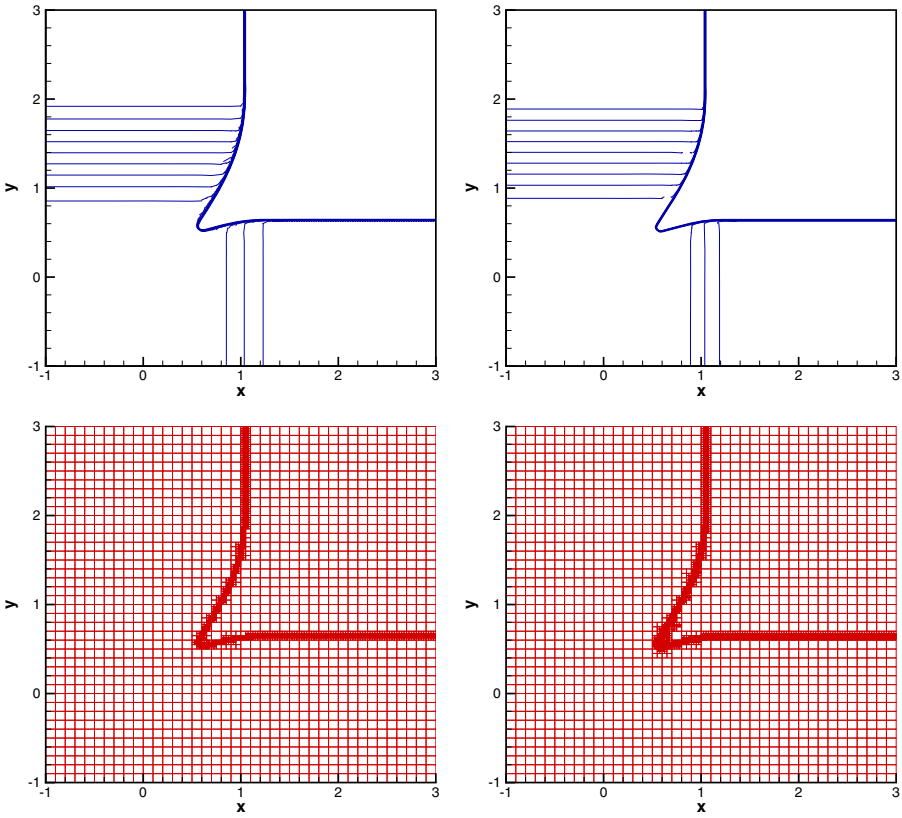


Fig. 3 Burgers equation for Riemann problem with IC-1, 40×40 initial mesh, 15 equally spaced contours *(top)* from 0.249 to 2.351 and the mesh *(bottom)* at $t = 0.8$, $k = 1$ *(left)* and $k = 2$ *(right)*

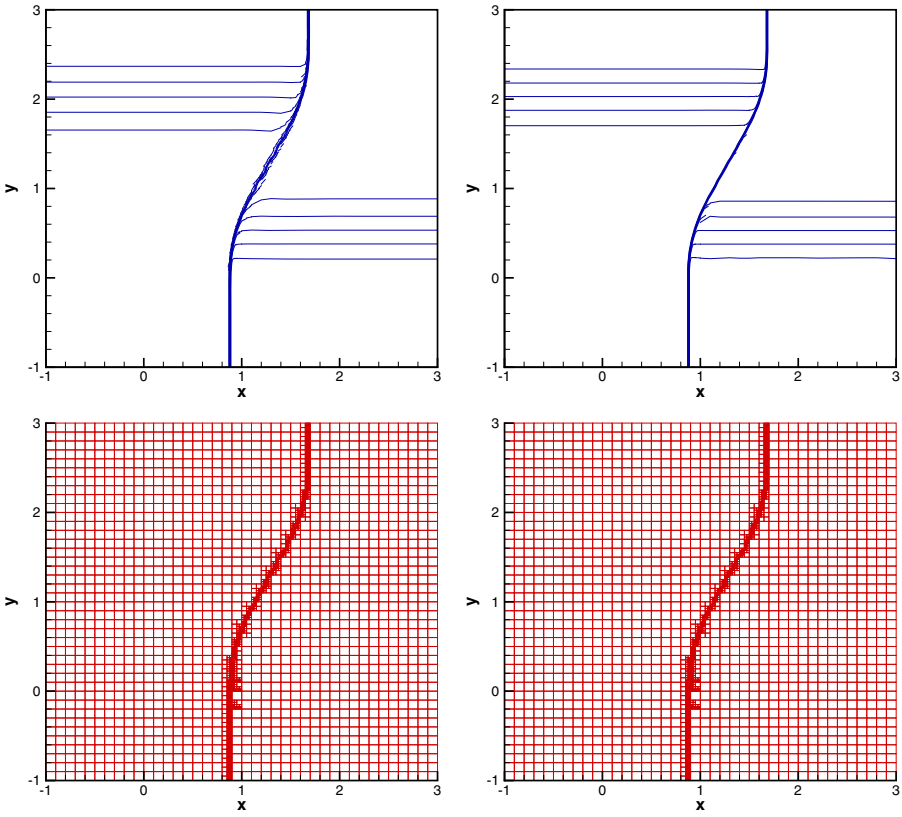


Fig. 4 Burgers equation for Riemann problem with IC-2, 40×40 initial mesh, 15 equally spaced contours *(top)* from 0.286 to 2.913 and the mesh *(bottom)* at $t = 0.8$, $k = 1$ *(left)* and $k = 2$ *(right)*

in which ρ, E and p represent the density, total energy and pressure, and u and v are the velocity components in the x- and y-directions, respectively. The system is completed by the equation of state

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

with gas constant $\gamma = 1.4$. The initial conditions are:

$$\text{IC-1:} \quad (\rho, u, v, p)^T = \begin{cases} (0.5313, 0, 0, 0.4)^T, & x > 0.5, y > 0.5, \\ (1, 0.7276, 0, 1)^T, & x < 0.5, y > 0.5, \\ (0.8, 0, 0, 1)^T, & x < 0.5, y < 0.5, \\ (1, 0, 0.7276, 1)^T, & x > 0.5, y < 0.5, \end{cases}$$

$$\text{IC-2:}\quad (\rho, u, v, p)^T = \begin{cases} (1.1, 0, 0, 1.1)^T, & x > 0.5,\ y > 0.5, \\ (0.5065, 0.8939, 0, 0.35)^T, & x < 0.5,\ y > 0.5, \\ (1.1, 0.8939, 0.8939, 1.1)^T, & x < 0.5,\ y < 0.5, \\ (0.5065, 0, 0.8939, 0.35)^T, & x > 0.5,\ y < 0.5. \end{cases}$$

We show the results in Figs. 5 and 6. We can see from the figures that in all cases, fine meshes (whose mesh levels are greater than zero) are used at or near the discontinuities while coarsest meshes (whose mesh levels are equal to zero) are used in smooth regions. The meshes fulfill our expectations and the shocks are well resolved.

Example 4.3 Forward facing step problem [20]. This is a standard test problem for high resolution schemes. The problem starts with a uniform, right-going

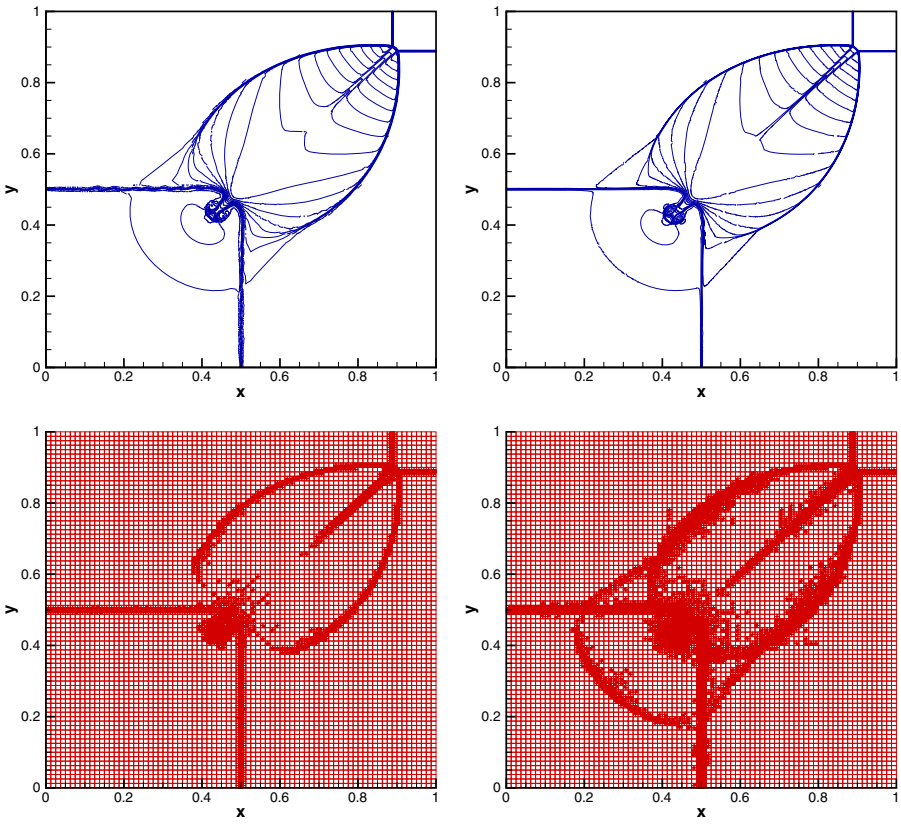


Fig. 5 Euler equations for Riemann problem with IC-1, 80×80 initial mesh, 30 equally spaced density contours *(top)* from 0.54 to 1.70 and the mesh *(bottom)* at $t = 0.25$, $k = 1$ *(left)* and $k = 2$ *(right)*

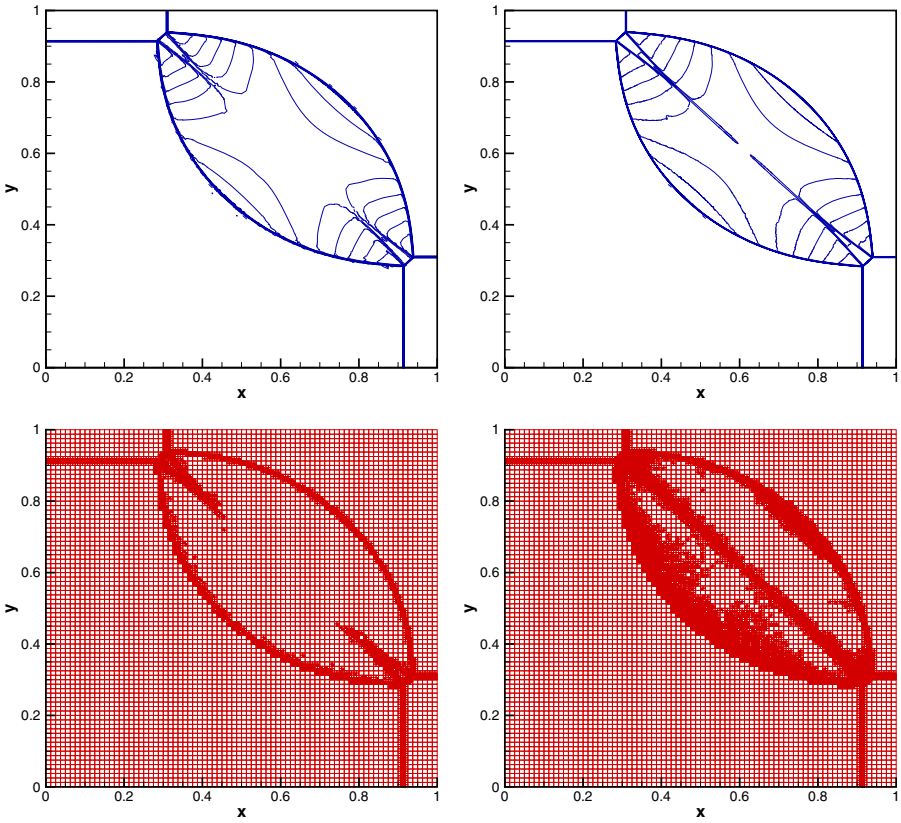


Fig. 6 Euler equations for Riemann problem with IC-2, 80×80 initial mesh, 30 equally spaced density contours *(top)* from 0.505 to 1.922 and the mesh *(bottom)* at $t = 0.25$, $k = 1$ *(left)* and $k = 2$ *(right)*

Mach 3 flow in a wind tunnel of 3 units long and 1 unit wide which contains a 0.2 units high step located 0.6 units from the left-hand end of the tunnel. Reflecting boundary conditions are used along the walls of the tunnel, and inflow and outflow boundary conditions are applied on the left and right boundaries, respectively. The problem is run until a simulation time of 4.0.

The corner of the step is a singularity, which leads to an erroneous entropy layer at the downstream bottom wall, as well as a spurious Mach stem at the bottom wall. But we do not modify our scheme near the corner because the artifacts will decrease when the corner singularity is better resolved [7], which can be automatically achieved by the adaptive algorithm in this paper. See the numerical results in Fig. 7 as an illustration. The figure also shows that the shocks are captured very well, and so is the upper slip line from the triple point in the $k = 2$ case.

Example 4.4 Shock passing a backward facing corner (diffraction). This problem has been used in [7, 21]. In the computation, negative density and/or

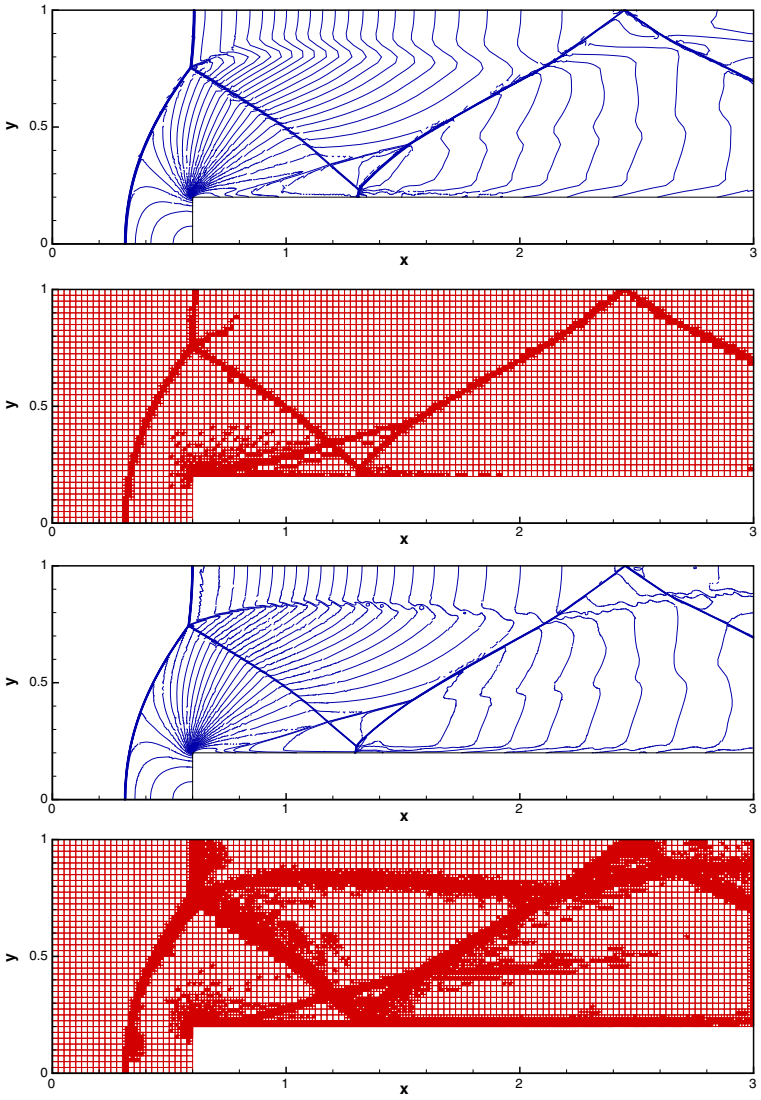


Fig. 7 Forward facing step problem, 120×40 initial mesh, 30 equally spaced density contours from 0.32 to 6.15 and the mesh at $t = 4$, $k = 1$ *(upper two)* and $k = 2$ *(lower two)*

pressure may appear below and to the right of the corner. We use the same positivity correction procedure as in [7] to avoid blow-ups. The computational domain is the union of $[0, 1] \times [6, 11]$ and $[1, 13] \times [0, 11]$. A pure right-moving shock of Mach 5.09 is initially positioned at $x = 0.5, 6 < y < 11$. The undisturbed air ahead of the shock has a density of 1.4 and a pressure of 1. The boundary conditions are inflow at $x = 0, 6 < y < 11$, reflective at the walls $0 < x < 1$, $y = 6$ and $x = 1, 0 < y < 6$, and outflow at the remainder. The

density contours and the mesh at $t = 2.3$ are presented in Fig. 8. We observe from the figures that the adaptive method generates finest mesh along the shocks and uses coarse mesh in the smooth region. As a result, shocks are captured sharply and efficiently.

Example 4.5 Double Mach reflection problem [20]. This is again a standard test problem for high resolution schemes. We use exactly the same setup as in [20]. This problem is governed by the two-dimensional Euler equations (4.1). The computational domain is $[0, 4] \times [0, 1]$, although only part of it, $[0, 3] \times [0, 1]$ is shown. The reflecting wall lies at the bottom of the computational domain starting from $x = \frac{1}{6}$. Initially a right-moving Mach 10 shock is positioned at $x = \frac{1}{6}$, $y = 0$, making a 60° angle with the x-axis. The undisturbed air ahead of the shock has a density of 1.4 and a pressure of 1. On the left and right boundaries, the inflow and outflow boundary conditions are used, respectively. For the bottom boundary, the reflective boundary condition is

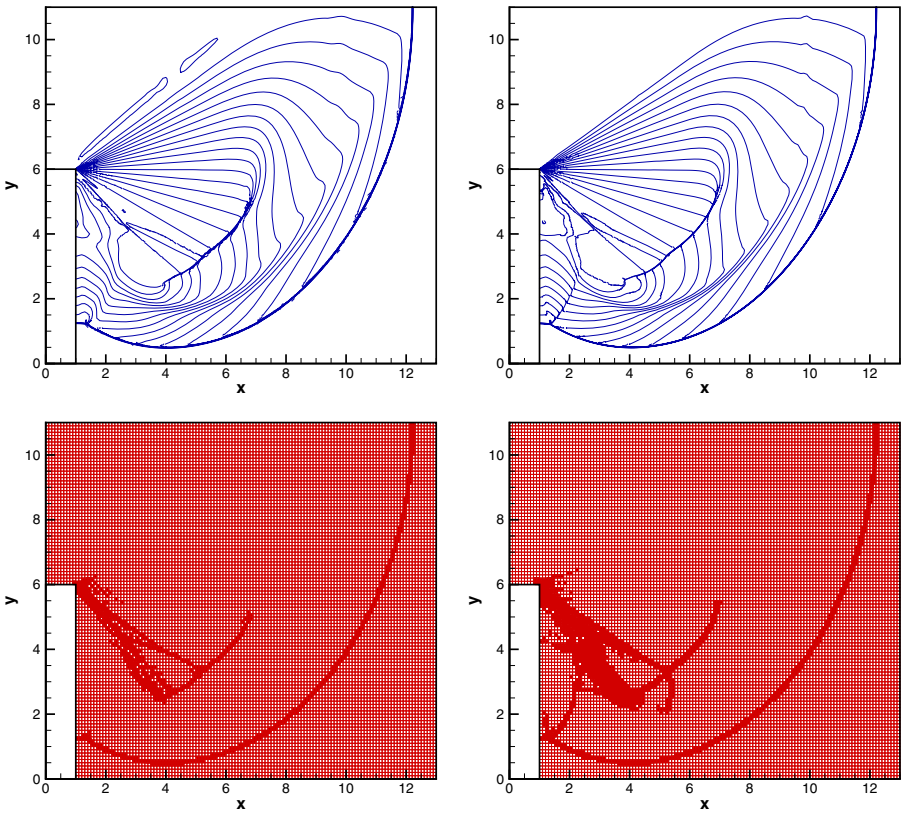


Fig. 8 Shock diffraction problem, 130×110 initial mesh, 20 equally spaced density contours *(top)* from 0.066227 to 7.0668 and the mesh *(bottom)* at $t = 2.3$, $k = 1$ *(left)* and $k = 2$ *(right)*

applied at the wall and the exact post-shock condition is used for the rest. Boundary conditions at the top correspond to the exact motion of a Mach 10 shock. The problem is run till a simulation time of 0.2 is reached.

We present the density contours and the mesh in Fig. 9, and show the more detailed zoomed-in figures around the double Mach stem in Fig. 10. Again

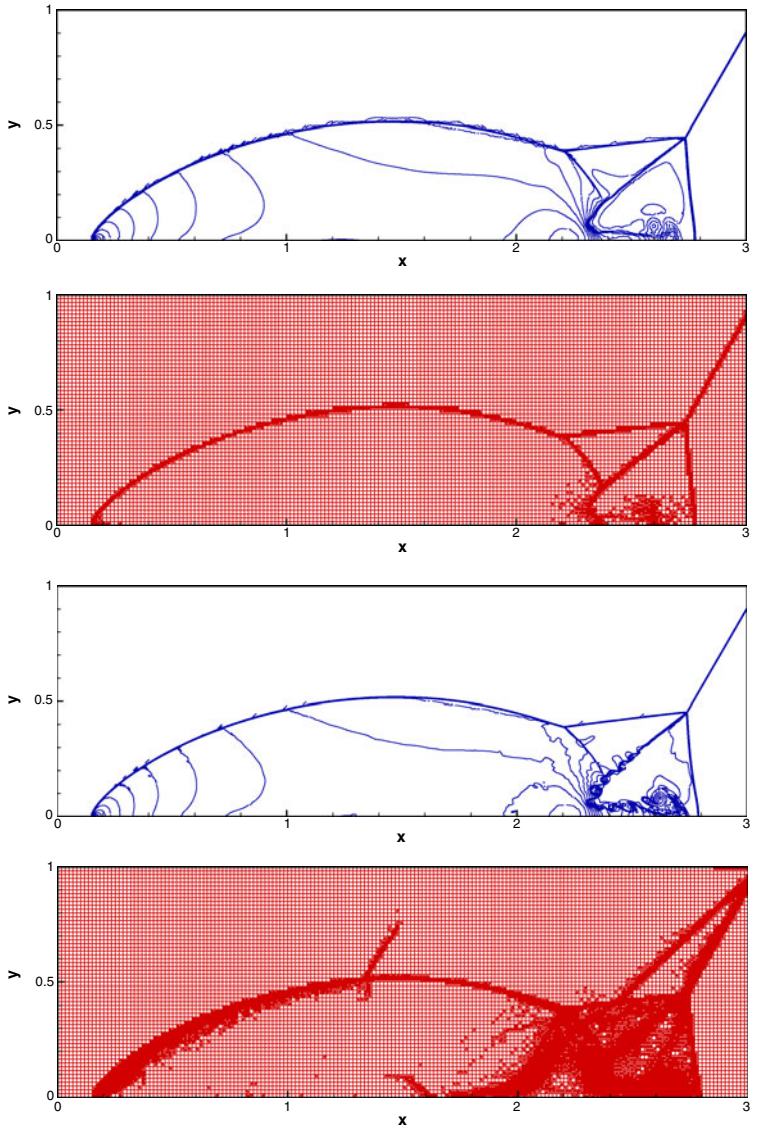


Fig. 9 Double Mach reflection problem, 240×60 initial mesh, 30 equally spaced density contours from 1.5 to 22.7 and the mesh at $t = 0.2$, $k = 1$ *(upper two)* and $k = 2$ *(lower two)*

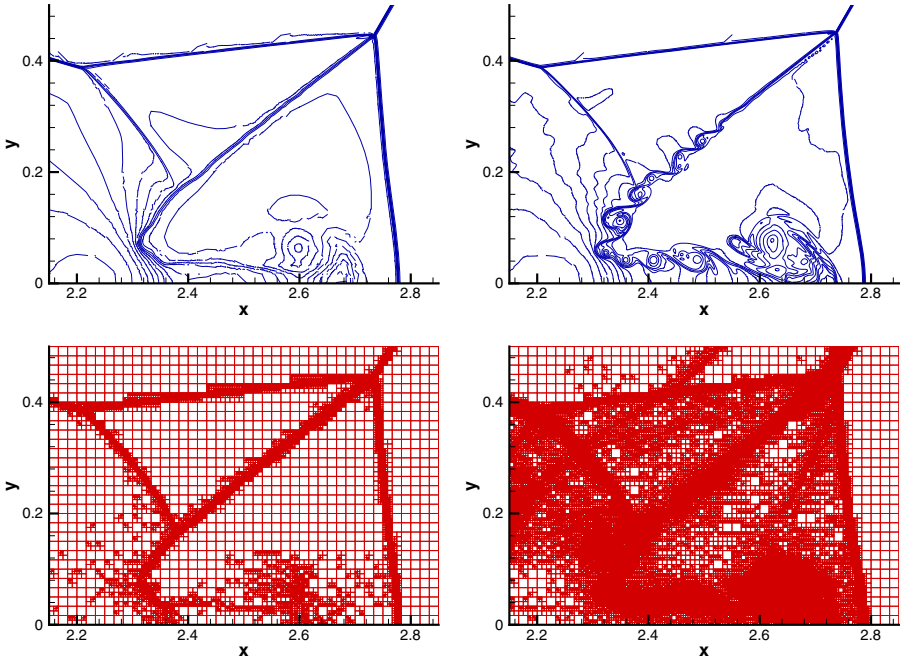


Fig. 10 Double Mach reflection problem, zoomed-in region around the double Mach stem, 240×60 initial mesh, 30 equally spaced density contours *(top)* from 1.5 to 22.7 and the mesh *(bottom)* at $t = 0.2$, $k = 1$ *(left)* and $k = 2$ *(right)*

we can see that the shocks are well captured. When $k = 2$, fine details of the complicated flow structure under the triple Mach stem are captured too.

To gain a better understanding of the effectiveness of the adaptive strategy in this work, for all the cases above we show the corresponding mesh data in Table 1, including (a) N_0: number of initial cells; (b) TDT: total dividing

Table 1 Mesh data

Case	N_0	k	TDT	N_T	$\bar{N}$	PR
Example 4.1, IC-1	1600	1	4.4E+4	8104	8366.5	2.04
		2	5.9E+4	9814	10024.8	2.45
Example 4.1, IC-2	1600	1	6.3E+4	5836	6595.3	1.61
		2	1.3E+5	7165	9795.8	2.39
Example 4.2, IC-1	6400	1	4.9E+5	51613	42980.6	2.62
		2	4.1E+6	119092	95290.1	5.82
Example 4.2, IC-2	6400	1	6.0E+5	64450	50569.1	3.09
		2	3.2E+6	142348	93856.8	5.73
Example 4.3	4032	1	2.3E+6	46515	45698.5	4.43
		2	6.2E+7	160785	123959.2	12.01
Example 4.4	13700	1	1.4E+6	59657	40664.4	1.16
		2	4.8E+6	109952	62337.4	1.78
Example 4.5	14400	1	2.4E+6	79227	59661.5	1.62
		2	2.5E+7	257298	260783.7	7.07

times; (c) N_T: number of cells at the final time level; (d) $\bar{N}$: average number of cells, defined by $\bar{N} = (\sum_{q=0}^{TOT} N_q)/TOT$ where N_q is the number of cells at the q-th time level and TOT is the total number of time levels; and (e) PR: the percentage ratio of $\bar{N}$ to the number of cells if a fully refined mesh was used, i.e. $PR = 100\bar{N}/(4^{LEV}N_0)$. Total merging times TMT is not shown in the table as it can be calculated by $TMT = TDT - (N_T - N_0)/3$.

In the table we can see that all the values of PR are far less than 100, which means that our adaptive algorithm needs much less cells than the one adopting fixed mesh provide that they produce comparable solutions. As a result, our adaptive algorithm has the advantage of saving the computational cost and improving the solution quality.

5 Concluding remarks

In this paper, the one-dimensional *h*-adaptive RKDG method using troubled-cell indicators developed in [22] is extended to the two-dimensional case. The idea of the extension is straightforward, and we lay the emphasis on the numerical implementations. Numerical results of classical test problems show the capability of our adaptive method in identifying and following the important features such as shocks and complicated structures.

References

1. Biswas, R., Devine, K., Flaherty, J.: Parallel, adaptive finite element methods for conservation laws. Appl. Numer. Math. **14**, 255–283 (1994)
2. Brio, M., Zakharian, A., Webb, G.: Two dimensional Riemann solver for Euler equations of gas dynamics. J. Comput. Phys. **167**, 177–195 (2001)
3. Cockburn, B., Hou, S., Shu, C.-W.: The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case. Math. Comput. **54**, 545–581 (1990)
4. Cockburn, B., Lin, S.-Y., Shu, C.-W.: TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one dimensional systems. J. Comput. Phys. **84**, 90–113 (1989)
5. Cockburn, B., Shu, C.-W.: TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework. Math. Comput. **52**, 411–435 (1989)
6. Cockburn, B., Shu, C.-W.: The Runge-Kutta local projection P^1-discontinuous Galerkin finite element method for scalar conservation laws. Math. Model. Numer. Anal. (M^2AN) **25**, 337–361 (1991)
7. Cockburn, B., Shu, C.-W.: The Runge-Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems. J. Comput. Phys. **141**, 199–224 (1998)
8. Cockburn, B., Shu, C.-W.: Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. J. Sci. Comput. **16**, 173–261 (2001)
9. Dedner, A., Makridakis, C., Ohlberger, M.: Error control for a class of Runge-Kutta discontinuous Galerkin methods for nonlinear conservation laws. SIAM J. Numer. Anal. **45**, 514–538 (2007)
10. Devine, K., Flaherty, J.: Parallel adaptive *hp*-refinement techniques for conservation laws. Appl. Numer. Math. **20**, 367–386 (1996)

11. Flaherty, J., Loy, R., Shephard, M., Szymanski, B., Teresco, J., Ziantz, L.: Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. J. Parallel Distrib. Comput. **47**, 139–152 (1997)
12. Hartmann, R., Houston, P.: Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. SIAM J. Sci. Comput. **24**, 979–1004 (2002)
13. Krivodonova, L., Xin, J., Remacle, J.-F., Chevaugeon, N., Flaherty, J.: Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. Appl. Numer. Math. **48**, 323–338 (2004)
14. Lax, P., Liu, X.: Solution of two dimensional Riemann problems of gas dynamics by positive schemes. SIAM J. Sci. Comput. **19**(2), 319–340 (1998)
15. Qiu, J., Shu, C.-W.: A comparison of troubled-cell indicators for Runge-Kutta discontinuous Galerkin mehtods using weighted essentially nonosillatory limiters. SIAM J. Sci. Comput. **27**, 995–1013 (2005)
16. Qiu, J., Shu, C.-W.: Runge-Kutta discontinuous Galerkin method using WENO limiters. SIAM J. Sci. Comput. **26**, 907–929 (2005)
17. Remacle, J.-F., Flaherty, J., Shephard, M.: An adaptive discontinuous Galerkin technique with an orthogonal basis applied to compressible flow problems. SIAM Rev. **45**, 53–72 (2003)
18. Shu, C.-W.: TVB uniformly high-order schemes for conservation laws. Math. Comput. **49**, 105–121 (1987)
19. Shu, C.-W., Osher, S.: Efficient implementation of essentially non-oscillatory shock-capturing schemes. J. Comput. Phys. **77**, 439–471 (1988)
20. Woodward, P., Colella, P.: The numerical simulation of two-dimensional fluid flow with strong shocks. J. Comput. Phys. **54**, 115–173 (1984)
21. Zhang, X., Shu, C.-W.: On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes. J. Comput. Phys. **229**, 8918–8934 (2010)
22. Zhu, H., Qiu, J.: Adaptive Runge-Kutta discontinuous Galerkin methods using different indicators: one-dimensional case. J. Comput. Phys. **228**, 6957–6976 (2009)