Published in final edited form as:
J Comput Phys. 2014 December 1; 278: 348–377. doi:10.1016/j.jcp.2014.08.042.

A New Runge-Kutta Discontinuous Galerkin Method with Conservation Constraint to Improve CFL Condition for Solving Conservation Laws

Zhiliang Xu[‡], **Xu-Yan Chen**[¶], and **Yingjie Liu**[§]

Zhiliang Xu: zxu2@nd.edu; Xu-Yan Chen: xchen@math.gatech.edu; Yingjie Liu: yingjie@math.gatech.edu

[‡]Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556

[¶]School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332

[§]School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332

Abstract

We present a new formulation of the Runge-Kutta discontinuous Galerkin (RKDG) method [9, 8, 7, 6] for solving conservation Laws with increased CFL numbers. The new formulation requires the computed RKDG solution in a cell to satisfy additional conservation constraint in adjacent cells and does not increase the complexity or change the compactness of the RKDG method. Numerical computations for solving one-dimensional and two-dimensional scalar and systems of nonlinear hyperbolic conservation laws are performed with approximate solutions represented by piecewise quadratic and cubic polynomials, respectively. The hierarchical reconstruction [17, 33] is applied as a limiter to eliminate spurious oscillations in discontinuous solutions. From both numerical experiments and the analytic estimate of the CFL number of the newly formulated method, we find that: 1) this new formulation improves the CFL number over the original RKDG formulation by at least three times or more and thus reduces the overall computational cost; and 2) the new formulation essentially does not compromise the resolution of the numerical solutions of shock wave problems compared with ones computed by the RKDG method.

1 Introduction

In this paper, we introduce a simple and effective technique to improve the Courant-Friedrichs-Lewy (CFL) condition of the Runge-Kutta discontinuous Galerkin (RKDG) method for solving nonlinear conservation laws while essentially keeping the complexity and other nice features of RKDG unchanged. The discontinuous Galerkin (DG) method was firstly introduced by Reed and Hill [24] as a technique to solve neutron transport problems. In a series of papers by Cockburn, Shu *et al.* [9, 8, 7, 6], the RKDG method has been

developed for solving nonlinear hyperbolic conservation laws and related equations. In their formulation, DG is used for spatial discretization with flux values at cell edges computed by either Riemann solvers or monotone flux functions, the total variation bounded (TVB) limiter [27, 9] is employed to eliminate spurious oscillations and the total variation diminishing (TVD) Runge-Kutta (RK) method [29] is used for the temporal discretization to ensure the stability of the numerical approach while simplifying the implementation. The RKDG method is compact and can be formulated on arbitrary meshes. It has enjoyed great success in solving the Euler equations for gas dynamics, compressible Navier-Stokes equations, viscous MHD equations and many other equations, and motivated many related new numerical techniques [1, 22].

In [9], the RKDG method is shown to be linearly stable when the CFL factor is bounded by $\frac{1}{2q+1}$ for the second-order and the third-order schemes in the one-dimensional (1D) space, where q is the degree of the polynomial approximating the solution. In [32], the RKDG solution is projected to the staggered covolume mesh to obtain distributional derivatives and then is projected back on each Runge-Kutta step which is analytically shown in 1D to significantly increase the CFL number. It is found in [19] that the central DG scheme on overlapping cells with Runge-Kutta time-stepping can use a CFL number larger than the one that RKDG method can take on non-overlapping cells when the order of accuracy of these schemes is above the first order. Using integral deferred correction for time discretization with improved CFL condition can be found in [5]. In [35], a technique is introduced which incorporates neighboring cell averages as additional constraints into the RKDG method. This technique improves the CFL condition. However, due to the use of multiple Lagrangian multipliers, the computational cost also increases during each time step. It would be desirable if there is a simple technique to increase CFL number of the RKDG method without introducing too much computational overhead while still being compact and maintaining its other nice properties. In this paper, we further develop the strategy in [35] which mixes the RKDG method with some of the finite volume reconstruction features [3] to achieve this goal.

We impose additional conservation constraint on the numerical solution computed by the RKDG method in the sense that in addition to letting an approximate polynomial solution supported on a cell conserve the cell average of this cell, this polynomial matches the prescribed cell averages supported on adjacent neighbors of this cell in a least-square sense. This is achieved by introducing a penalty term to the energy functional associated with the RKDG formulation. The resulting linear system contains the same number of equations to be solved as in RKDG, and is referred to as the constrained RKDG method in the sections that follow. We illustrate the effectiveness of our technique by analytically estimating the CFL factor and using the 1D and two-dimensional (2D) third- and fourth-order accurate schemes to compute both smooth and discontinuous solutions test problems, respectively. The 2D test cases are solved on triangular meshes. In this study, we find that the constrained RKDG method increases the CFL number over the original RKDG method by three times or more, and essentially does not destroy the resolution of the discontinuous numerical solutions limited by hierarchical reconstruction (HR) [17]. However, there is also one limitation of the proposed method in terms of the cost efficiency when solving problems

with smooth solutions. From numerical tests reported in this paper, we observe that RKDG method can achieve better accuracy using less CPU time for solving the smooth solution problem. The constrained RKDG method needs two times more CPU time than that of the RKDG method to reach the same magnitude of the error.

We also point out that the computer memory requirement for the constrained RKDG method is the same as that for the RKDG method. The computer memory utilized by both methods is mainly for storing the degrees of freedom for each cell. Thus we do not perform a study on this aspect.

Using finite volume limiting techniques on solutions computed by the RKDG method for conservation laws has been explored by many researchers. In [23, 38, 39], the WENO finite volume reconstruction procedures are used as the limiter on cells where the solutions supported on these cells become oscillatory. In [21], Luo *et al.* developed a Hermite WENO-based limiter for the second order RKDG method on unstructured meshes following [23]. It would be convenient to use a compact limiting technique since the RKDG method is a compact method. The first of such limiters is the TVB projection limiter by Cockburn and Shu, which uses the lowest and (limited) first Legendre moments locally where non-smoothness is detected. Other compact limiting techniques which are supposed to remove spurious oscillations using information only from adjacent cells for any orders include the moment limiter [4] and the recently developed HR [17]. In [33], HR on 2D triangular meshes has been studied for the piecewise quadratic DG method; a partial neighboring cell technique has been developed and a component-wise WENO-type linear reconstruction is used on each hierarchical level. This new technique has good resolution and accuracy on unstructured meshes and is easy to implement since the weights on each hierarchical level are trivial to compute and essentially independent of the mesh. An up to fourth-order accurate point-wise HR for unstructured triangular meshes has been developed in [35]. Besides the techniques mentioned above, there are also many research works of limiters for high order schemes for solving various problems. One goal of the paper is to verify if our technique for improving the CFL number of RKDG works well with HR. We also present a local iteration technique and characteristic decomposition for HR which improves the resolution of the solutions computed by fourth-order accurate schemes in the vicinity of discontinuities.

The paper is organized as follows. Section 2 describes the conservation constrained RKDG formulation, analytical estimate of the CFL number and the HR limiting procedure. Results of numerical tests are presented in Section 3. Concluding remarks and a plan for the future work are included in Section 4.

2 Formulation of the Method

2.1 Outline of the approach

Here we summarize the conservation constrained Runge-Kutta discontinuous Galerkin finite element method for solving time dependent hyperbolic conservation laws (2.1)

$$\begin{cases} \frac{\partial u_k}{\partial t}+\sum_{j=1}^{d}\frac{\partial\left(F_{k,j}(\mathbf{u})\right)}{\partial x_j}=0, k=1,\ldots,p, \text{in}\Omega\times(0,\mathrm{T}), \\ \mathbf{u}(\mathbf{x},0)=\mathbf{u}_0(\mathbf{x}), \end{cases} \quad (2.1)$$

where $\Omega \subset R^d$, d is the spatial dimension, $d = 1$ or 2 in this paper. $\mathbf{u} = (u_1, \ldots, u_p)^T$ is the solution vector and the column vectors $\mathbf{F}_j(\mathbf{u}) = (F_{1,j}(\mathbf{u}), \ldots, F_{p,j}(\mathbf{u}))^T$ are the flux vectors. We use the following notation $\mathbf{x} = (x_1) \equiv (x)$ in 1D, and $\mathbf{x} = (x_1, x_2) \equiv (x, y)$ in 2D for spatial variables in the rest of the paper, respectively.

The method of lines approach is used to evolve the solution in time. Specifically, the third-or fourth-order accurate TVD RK time-stepping method is used, based on the accuracy of spatial discretization. At each RK stage, the constrained DG method is used. In the vicinities of discontinuities of the solution, the computed piecewise polynomial solution is limited by HR to remove spurious oscillations.

2.2 Conservation constrained discontinuous Galerkin method

In this section, we develop the conservation constrained DG method for solving Eq. (2.1) with $d = 1$, or 2 and $p = 1$ or $p > 1$. We use these 1D and 2D implementations as examples to present the essential ingredients of the method, while keeping in mind that this formulation can be naturally extended to $d = 3$.

2.2.1 1D Conservation constrained discontinuous Galerkin method—For the sake of presenting the main idea of the conservation constrained DG method, we concentrate on developing this method for solving the 1D scalar problem (2.2) in this subsection, and then extend it to the multi-D system case in the next subsection.

$$\begin{cases} \frac{\partial u}{\partial t}+\frac{\partial(f(u))}{\partial x}=0, \text{in}x\in[a,b]\times(0,\mathrm{T}), \\ u(x,0)=u_0(x), \end{cases} \quad (2.2)$$

In 1D, let $a=x_{\frac{1}{2}}<x_{\frac{3}{2}}<\ldots<x_{\mathscr{N}-\frac{1}{2}}<x_{\mathscr{N}+\frac{1}{2}}=b$ be a partition of domain $[a, b] \subset R$. The 1D cells, cell centers and cell sizes are defined by

$$\mathscr{I}_i \equiv (x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}), x_i \equiv \frac{1}{2}(x_{i-\frac{1}{2}}+x_{i+\frac{1}{2}}), \Delta x_i \equiv x_{i+\frac{1}{2}}-x_{i-\frac{1}{2}}, i=1,\ldots,\mathscr{N}, \quad (2.3)$$

respectively. Like the DG method, the approximate solution u_h is taken in the finite element space $V_h = \{\mathrm{P}(x) : \mathrm{P}|_{\mathscr{I}_i}$ is a polynomial of degree $\leq q$, $i = 1, \ldots, \mathscr{N}\}$.

In this work, the local basis function set $\mathscr{B}_i \equiv \{\phi_m^{(i)}(x): m=0,\ldots,r\}$ of the the finite element space V_h on $\mathscr{I}_i$ is chosen to be Legendre polynomials, $r = q$. For example,

$$\phi_0^{(i)}(x)=1, \phi_1^{(i)}(x)=x-x_i, \phi_2^{(i)}(x)=(x-x_i)^2-\frac{1}{12}\Delta x_i^2, \ldots. \quad (2.4)$$

On each cell $\mathscr{I}_i$, the approximate solution $u_h(t, x)$ to Eq. (2.2) is expressed as

$$u_h(t,x)=\sum_{m=0}^{r} c_m^{(i)}(t)\phi_m^{(i)}(x). \quad (2.5)$$

To derive the semi-discrete DG formulation for solving Eq. (2.2), $\upsilon_h \in$ Span $\{\mathscr{B}_i\}$ is multiplied to Eq. (2.2), integration over $\mathscr{I}_i$ and integration by parts lead to

$$\frac{d}{dt}\int_{\mathscr{I}_i} u\upsilon_h dx+[f(u(t,x))\upsilon_h(x)]\Big|_{\substack{x=x_{i+1/2}\\ x=x_{i-1/2}}} -\int_{\mathscr{I}_i} f(u)\frac{d\upsilon_h(x)}{dx}dx=0. \quad (2.6)$$

Since the approximate solution u_h is discontinuous across cell edges, the interfacial fluxes are not uniquely determined. The interfacial flux function $f(u_h)$ can be replaced by the Lax-Friedrich flux function (see e.g. [28]) defined as

$$h_{i-1/2}=h(u^-_{h,i-1/2},u^+_{h,i-1/2}) \equiv \frac{1}{2}\left(f(u^-_{h,i-1/2})+f(u^+_{h,i-1/2})\right)+\frac{\alpha}{2}(u^-_{h,i-1/2}-u^+_{h,i-1/2}),$$

where α is the largest characteristic speed, $u^+_{h,i-1/2}$ and $u^-_{h,i-1/2}$ are the right- and left-hand limits of u_h respectively at $x_{i-1/2}$.

Equation (2.6) then leads to the semi-discrete DG scheme

$$\frac{d}{dt}\int_{\mathscr{I}_i} u_h\phi_m^{(i)}d\mathbf{x}+\Delta_+\left(h_{i-1/2}\phi_m^{(i)}(x_{i-1/2})\right)-\int_{\mathscr{I}_i} f(u_h)\frac{d\phi_m^{(i)}(x)}{dx}dx=0, m=0,\ldots,r, \quad (2.7)$$

for which each coefficient $c_m^{(i)}(t)$ of u_h must satisfy. Here Δ_+ denotes the forward difference operator $\Delta_+ a_i = a_{i+1} - a_i$.

The resulting system of ordinary differential equations can be solved by a TVD Runge-Kutta method [29] which builds on convex combinations of several forward Euler schemes of (2.7).

Our additional conservation constraint is performed within each of the component forward Euler scheme. To be specific, a forward Euler scheme for solving Eq. (2.7) can be written as

$$\int_{\mathscr{I}_i} u_h^{n+1}\phi_m^{(i)}dx=\int_{\mathscr{I}_i} u_h^n\phi_m^{(i)}dx-\Delta t_n\left(\Delta_+\left(h_{i-1/2}\phi_m^{(i)}(x_{i-1/2})\right)-\int_{\mathscr{I}_i} f(u_h^n)\frac{d\phi_m^{(i)}(x)}{dx}dx\right), m=0,\ldots,r, \quad (2.8)$$

where the superscript n denotes the time level t_n, and $\Delta t_n \equiv t_{n+1} - t_n$. In particular, by solving Eq. (2.8) with $m = 0$, we obtain the cell average of u_h^{n+1} over cell $\mathscr{I}_i$, denoted by $\overline{u_i^{n+1}}$, just as with a finite volume scheme.

Now suppose the cell averages $\left\{\overline{u_i^{n+1}}\right\}_{i=1}^{\mathscr{N}}$ have been computed on all cells. We do not compute the rest of the degrees of freedom of u_h^{n+1} on cell $\mathscr{I}_i$ by using Eq. (2.8). Instead, we

let u_h^{n+1} on cell $\mathscr{I}_i$ minimize an energy functional (variational to (2.8)) subject to that it conserves additional given cell averages not only in cell $\mathscr{I}_i$ but also in some of its neighbors.

In order to do so, let's rewrite (2.8) in cell $\mathscr{I}_i$ as

$$\int_{\mathscr{I}_i} u_h^{n+1} \upsilon_h dx = \mathscr{L}_{\mathscr{I}_i}(\upsilon_h), \quad (2.9)$$

for any $\upsilon_h \in \text{Span}\{\mathscr{B}_i\}$. Here $\mathscr{L}_{\mathscr{I}_i}(\upsilon_h)$ represents the right-hand-side of (2.8) with $\phi_m^{(i)}$ being replaced by υ_h so that

$$\mathscr{L}_{\mathscr{I}_i}(\upsilon_h) = \int_{\mathscr{I}_i} u_h^n \upsilon_h dx - \Delta t_n \left(\Delta_+ \left(h_{i-1/2} \upsilon_h(x_{i-1/2}) \right) - \int_{\mathscr{I}_i} f(u_h^n) \frac{d\upsilon_h(x)}{dx} dx \right). \quad (2.10)$$

It is easy to see that $\mathscr{L}_{\mathscr{I}_i}(\upsilon_h)$ a linear bounded functional defined on the finite element space on $\mathscr{I}_i$. The variational form of (2.9) is to find u_h^{n+1} in the finite element space on $\mathscr{I}_i$ such that it minimizes the energy functional

$$E(\upsilon_h) = \frac{1}{2} \int_{\mathscr{I}_i} (\upsilon_h)^2 dx - \mathscr{L}_{\mathscr{I}_i}(\upsilon_h). \quad (2.11)$$

In Sec. 2.1.1 of [35], the conservation constrained RKDG formulation on cell $\mathscr{I}_i$ was described as replacing each component forward Euler scheme by finding u_h^{n+1} in the finite element space on $\mathscr{I}_i$, such that

$$E(u_h^{n+1}) = \text{Minimum of} \{E(\upsilon_h) : \upsilon_h \in \text{Span}\{\mathscr{B}_i\}\}, \text{subject to} \frac{1}{\Delta x_J} \int_{\mathscr{I}_J} \upsilon_h dx = \overline{u_J^{n+1}}, J = i-1, i, i+1. \quad (2.12)$$

Here we typically choose the set $\{\mathscr{I}_J : J = i-1, i, i+1\}$ consists of cell $\mathscr{I}_i$ and its adjacent cells.

This constrained minimization problem (2.12) can be solved by the method of Lagrangian multiplier as follows

$$\int_{\mathscr{I}_i} u_h^{n+1} \phi_m^{(i)} dx - \mathscr{L}_{\mathscr{I}_i}(\phi_m^{(i)}) = \sum_{J=i-1}^{i+1} \frac{\lambda_J^{(i)}}{\Delta x_J} \int_{\mathscr{I}_J} \phi_m^{(i)} dx, m = 0, \ldots, r \quad \frac{1}{\Delta x_J} \int_{\mathscr{I}_J} u_h^{n+1} dx = \overline{u_J^{n+1}}, J = i-1, i, i+1, \quad (2.13)$$

where $\left\{ \lambda_j^{(i)} \right\}$ are Lagrangian multipliers. The coefficients $c_m^{(i)}(t_{n+1})$ of u_h^{n+1} (see Eq. (2.5)) are determined by solving the linear system (2.13). Note that the left-hand-side of the first equation of (2.13) is in the same form as equation (2.9) or (2.8).

Even though this technique increases the CFL number, the use of Lagrangian multipliers also increases the dimensions of the linear system to be solved. In order to overcome this problem, we introduce a new minimization problem without any constraint as follows:

Find $\tilde{u}_h^{n+1} \in \mathrm{Span}\{\mathscr{B}_i\}$ such that

$$E_2(\tilde{u}_h^{n+1}) = \mathrm{Min}\{E_2(v_h) : v_h \in \mathrm{Span}\{\mathscr{B}_i\}\}, \quad (2.14)$$

where

$$E_2(v_h) = \left(\frac{1}{\Delta x_i}\right) E(v_h) + \mu \sum_{J=i-1}^{i+1} \left(\frac{1}{\Delta x_J}\int_{\mathscr{I}_J} v_h dx - \overline{u_J^{n+1}}\right)^2, \quad (2.15)$$

and $\mu \geq 0$ is a constant. Note that when $\mu = 0$ the formulation returns to the standard DG with forward Euler time-stepping.

The variational formulation of problem (2.14) is to find $\tilde{u}_h^{n+1} \in \mathrm{Span}\{\mathscr{B}_i\}$ such that

$$\left(\frac{1}{\Delta x_i}\right)\left(\int_{\mathscr{I}_i} \tilde{u}_h^{n+1}\phi_m^{(i)} dx - \mathscr{L}_{\mathscr{I}_i}(\phi_m^{(i)})\right) + 2\mu \sum_{J=i-1}^{i+1} \left(\frac{1}{\Delta x_J}\int_{\mathscr{I}_J} \phi_m^{(i)} dx\right)\left(\frac{1}{\Delta x_J}\int_{\mathscr{I}_J} \tilde{u}_h^{n+1} dx - \overline{u_J^{n+1}}\right) = 0, m = 0, \ldots, r. \quad (2.16)$$

In order to preserve the cell average $\overline{u_i^{n+1}}$ defined on $\mathscr{I}_i$, we obtain $u_h^{n+1} \in \mathrm{Span}\{\mathscr{B}_i\}$ by modifying $\tilde{u}_h^{n+1}$ as follows

$$u_h^{n+1} \equiv \tilde{u}_h^{n+1} + \overline{u_i^{n+1}} - \frac{1}{\Delta x_i}\int_{\mathscr{I}_i} \tilde{u}_h^{n+1} dx. \quad (2.17)$$

Specifically, let the solution $\tilde{u}_h^{n+1}$ to Eq. (2.16) take the form

$$\tilde{u}_h^{n+1} = \sum_{m=0}^{r} \tilde{c}_m^{(i)}(t_{n+1})\phi_m^{(i)}(x).$$

u_h^{n+1} is obtained by modifying the 0th degree coefficient $\tilde{c}_0^{(i)}(t_{n+1})$ of $\tilde{u}_h^{n+1}$

$$c_0^{(i)}(t_{n+1}) = \tilde{c}_0^{(i)}(t_{n+1}) + \overline{u_i^{n+1}} - \frac{1}{\Delta x_i}\int_{\mathscr{I}_i} \tilde{u}_h^{n+1} dx \quad (2.18)$$

and letting

$$c_m^{(i)}(t_{n+1}) = \tilde{c}_m^{(i)}(t_{n+1}), \text{for} m = 1, \ldots, r. \quad (2.19)$$

To sum up, solving Eq. (2.16) and subsequently enforcing conservation of the solution by Eq. (2.17) complete the new conservation constrained DG method.

Remark 1. It's easy to verify that energy functional E_2 defined in Eq. (2.15) is invariant (subject to a scalar multiplication) under any affine change of coordinates.

Remark 2. Note that the linear system (2.16) consists of the same number of equations as in a RKDG step (2.8). Therefore the complexity of the new method is close to that of the standard RKDG (without using orthogonal basis functions).

The linear system (2.16) has a unique solution. In fact, consider the associate homogeneous system with $\mathscr{L}_{\mathscr{I}_i}(\upsilon_h) = 0$ and $\overline{u_J^{n+1}}=0$ for all $J = i - 1, i, i + 1$,

$$\left(\frac{1}{\Delta x_i}\right)\left(\int_{\mathscr{I}_i}\tilde{u}_h^{n+1}\upsilon_h dx\right)+2\mu\sum_{J=i-1}^{i+1}\left(\frac{1}{\Delta x_J}\int_{\mathscr{I}_J}\upsilon_h dx\right)\left(\frac{1}{\Delta x_J}\int_{\mathscr{I}_J}\tilde{u}_h^{n+1}dx\right)=0, \quad (2.20)$$

for any $\upsilon_h \in \mathrm{Span}\{\mathscr{B}_i\}$. Let $\upsilon_h=\tilde{u}_h^{n+1}$. We conclude that $\int_{\mathscr{I}_i}|\tilde{u}_h^{n+1}|^2 dx=0$, which implies $\tilde{u}_h^{n+1}\equiv 0$.

Remark 3. A compromised formulation with only one Lagrangian multiplier can be written as follows:

Find $\tilde{u}_h^{n+1} \in \mathrm{Span}\{\mathscr{B}_i\}$ such that

$$E_2(\tilde{u}_h^{n+1})= \begin{array}{l}\mathrm{Min}\{E_2(\upsilon_h):\upsilon_h \in \mathrm{Span}\{\mathscr{B}_i\}\} \\ \text{subject to}\ \frac{1}{\Delta x_i}\int_{\mathscr{I}_i}\upsilon_h dx=\overline{u_i^{n+1}}\end{array}. \quad (2.21)$$

This method has similar complexity and CFL numbers to that of the formulation (2.14), (2.15) and (2.17).

2.2.2 2D Conservation constrained discontinuous Galerkin method—In this subsection, we discuss how to develop the constrained DG method for solving 2D hyperbolic systems represented by Eq. (2.1). The physical domain $\Omega \subset R^2$ is partitioned into a collection of $\mathscr{N}$ non-overlapping cells $\mathscr{T}_h = \{\mathscr{K}_i : i = 1, \ldots, \mathscr{N}\}$ so that $\Omega=\bigcup_{i=1}^{\mathscr{N}}\mathscr{K}_i$. Here $\mathscr{K}_i$ represents a triangular cell and for simplicity, it is assumed that there are no hanging nodes in $\mathscr{T}_h$. In this work, each component of the approximate solution $\mathbf{u}_h$ is taken in the finite element space

$$V_h=\{\mathrm{P}:\mathrm{P}|_{\mathscr{K}_i}\text{is a polynomial of degree} \le q\}.$$

We note that the same notation $\mathscr{B}_i$ is also used to denote the basis function set which spans the finite element space on 2D cell $\mathscr{K}_i$ to avoid introducing too many notations. Specifically, for the 2D case we define

$$\mathscr{B}_i=\left\{\phi_m^{(i)}(x-x_i,y-y_i):m=0,\ldots,r\right\}=\left\{1,\frac{(x-x_i)}{\sqrt{|\mathscr{K}_i|}},\frac{(y-y_i)}{\sqrt{|\mathscr{K}_i|}},\frac{(x-x_i)^2}{(\sqrt{|\mathscr{K}_i|})^2},\frac{(x-x_i)(y-y_i)}{(\sqrt{|\mathscr{K}_i|})^2},\frac{(y-y_i)^2}{(\sqrt{|\mathscr{K}_i|})^2},\ldots,\frac{(y-y_i)^q}{(\sqrt{|\mathscr{K}_i|})^q}\right\}, \quad (2.22)$$

where $\mathbf{x}_i \equiv (x_i, y_i)$ is the centroid of $\mathscr{K}_i$, $r+1 = (q+1)(q+2)/2$, and $|\mathscr{K}_i|$ is the area of cell $\mathscr{K}_i$. Here $\mathscr{B}_i$ is a 2D polynomial basis function set of degree at most q in cell $\mathscr{K}_i$, which consists

of the monomials of 2D Taylor expansions about the cell centroid, (x_i, y_i), and scaled by the area of the cell raised to proper powers.

Without loss of generality, on each cell $\mathscr{K}_i$, the approximate solution $u_{h,k}(t, \mathbf{x})$ of the k^{th} equation of (2.1) is expressed as

$$u_{h,k}(t,\mathbf{x})=\sum_{m=0}^{r} c_{m,k}^{(i)}(t)\phi_m^{(i)}(\mathbf{x}). \quad (2.23)$$

The semi-discrete DG formulation for solving the k^{th} equation of (2.1) can be expressed as

$$\frac{d}{dt}\int_{\mathscr{K}_i} u_{h,k}\phi_m^{(i)}d\mathbf{x}+\int_{\partial\mathscr{K}_i} h_k\phi_m^{(i)}d\Gamma-\int_{\mathscr{K}_i}\mathbf{F}_k(\mathbf{u}_h)\cdot\nabla\phi_m^{(i)}d\mathbf{x}=0, m=0,\ldots,r, \quad (2.24)$$

for which the coeffients $c_{m,k}^{(i)}(t)$ of $u_{h,k}$ must satisfy, where $\mathbf{F}_k = (F_{k,0}(\mathbf{u}), \ldots, F_{k,1}(\mathbf{u}))$.

In this 2D case, we also choose the numerical flux function h_k of Eq. (2.24) to be the Lax-Friedrich flux function, defined as

$$h_k=h_k(\mathbf{u}_h^{in},\mathbf{u}_h^{out}) \equiv \frac{1}{2}(\mathbf{F}_k(\mathbf{u}_h^{in})\cdot\mathbf{n}_i+\mathbf{F}_k(\mathbf{u}_h^{out})\cdot\mathbf{n}_i)+\frac{\alpha}{2}(u_{h,k}^{in}-u_{h,k}^{out}), k=1,\ldots,p,$$

where α is the largest characteristic speed, and

$$\mathbf{u}_h^{in}(\mathbf{x},t)=\lim_{\mathbf{y}\to\mathbf{x},\mathbf{y}\in\mathscr{K}_i^{\text{int}}}\mathbf{u}_h(\mathbf{y},t), \quad (2.25)$$

$$\mathbf{u}_h^{out}(\mathbf{x},t)=\lim_{\mathbf{y}\to\mathbf{x},\mathbf{y}\notin\overline{\mathscr{K}_i}}\mathbf{u}_h(\mathbf{y},t). \quad (2.26)$$

Here $\mathscr{K}_i^{\text{int}}$ stands for the interior of cell $\mathscr{K}_i$ and $\overline{\mathscr{K}_i}$ is the closure of $\mathscr{K}_i$.

Similar to the implementation for solving the 1D scalar equation described in subsection 2.2.1, a forward Euler scheme for solving Eq. (2.24) can be written as

$$\int_{\mathscr{K}_i} u_{h,k}^{n+1}\phi_m^{(i)}d\mathbf{x}=\int_{\mathscr{K}_i} u_{h,k}^{n}\phi_m^{(i)}d\mathbf{x}-\Delta t_n\left(\int_{\partial\mathscr{K}_i} h_k^n\phi_m^{(i)}d\Gamma-\int_{\mathscr{K}_i}\mathbf{F}_k(\mathbf{u}_h^n)\cdot\nabla\phi_m^{(i)}d\mathbf{x}\right), m=0,\ldots,r. \quad (2.27)$$

By solving Eq. (2.27) with $m = 0$, we obtain the cell average of $u_{h,k}^{n+1}$ over cell $\mathscr{K}_i$, denoted by $\overline{u_{i,k}^{n+1}}$, just as with a finite volume scheme.

Now suppose the cell averages $\{\overline{u_{i,k}^{n+1}}\}_{i=1}^{\mathscr{N}}$ have been computed on all cells. We compute the remaining degrees of freedom of $u_{h,k}^{n+1}$ on cell $\mathscr{K}_i$ by using the constrained DG method.

In order to do so, let's rewrite (2.27) in cell $\mathscr{K}_i$ as

$$\int_{\mathscr{K}_i} u_{h,k}^{n+1} \upsilon_h d\mathbf{x} = \mathscr{L}_{\mathscr{K}_i}(\upsilon_h), \quad (2.28)$$

for any $\upsilon_h \in \text{Span}\{\mathscr{B}_i\}$. Here $\mathscr{L}_{\mathscr{K}_i}(\upsilon_h)$ represents the right-hand-side of (2.27) with $\phi_m^{(i)}$ being replaced by υ_h, and is a linear bounded functional defined on the finite element space on $\mathscr{K}_i$.

Let's redefine the energy functionals for this 2D case

$$E(\upsilon_h) = \frac{1}{2}\int_{\mathscr{K}_i} (\upsilon_h)^2 d\mathbf{x} - \mathscr{L}_{\mathscr{K}_i}(\upsilon_h), \quad (2.29)$$

and

$$E_2(\upsilon_h) = \left(\frac{1}{|\mathscr{K}_i|}\right) E(\upsilon_h) + \mu \sum_{j \in N(\mathscr{T}_{C,i})} \left(\frac{1}{|\mathscr{K}_j|}\int_{\mathscr{K}_j} \upsilon_h d\mathbf{x} - \overline{u_{j,k}^{n+1}}\right)^2, \quad (2.30)$$

$N(\mathscr{T}_{C,i})$ consists of indices of cells in the set $\mathscr{T}_{C,i}$ defined by Eq. (2.32), and $\mu \geq 0$ is a constant.

Following the idea explained in subsection 2.2.1, we introduce the following minimization problem (2.31) for the 2D case for finding $u_{h,k}^{n+1}$ in the finite element space on $\mathscr{K}_i$ without any constraint:

Find $\tilde{u}_{h,k}^{n+1} \in \text{Span}\{\mathscr{B}_i\}$ such that

$$E_2(\tilde{u}_{h,k}^{n+1}) = \text{Min}\{E_2(\upsilon_h) : \upsilon_h \in \text{Span}\{\mathscr{B}_i\}\}, \quad (2.31)$$

where $E_2(\upsilon_h)$ is defined by Eq. (2.30) for the 2D case. Note again that when $\mu = 0$ the formulation returns to the standard DG with forward Euler time-stepping as in the 1D case.

Here we define the set $\mathscr{T}_{C,i}$ as

$$\mathscr{T}_{C,i} = \{\mathscr{K}_i, \text{and some immediate neighbors of} \mathscr{K}_i\}. \quad (2.32)$$

In 2D, this set includes $\mathscr{K}_i$ and some of its neighbors, which are cells sharing same edges or vertices with $\mathscr{K}_i$. See also Sec. 2.3.1 for specific implementations of $\mathscr{T}_{C,i}$.

The solution $\tilde{u}_{h,k}^{n+1} \in \text{Span}\{\mathscr{B}_i\}$ to the problem (2.31) is obtained by solving the following variational problem

$$\left(\frac{1}{|\mathscr{K}_i|}\right)\left(\int_{\mathscr{K}_i} \tilde{u}_{h,k}^{n+1}\phi_m^{(i)} d\mathbf{x} - \mathscr{L}(\phi_m^{(i)})\right) + 2\mu \sum_{j \in N(\mathscr{T}_{C,i})} \left(\frac{1}{|\mathscr{K}_j|}\int_{\mathscr{K}_j} \phi_m^{(i)} d\mathbf{x}\right)\left(\frac{1}{|\mathscr{K}_j|}\int_{\mathscr{K}_j} \tilde{u}_{h,k}^{n+1} d\mathbf{x} - \overline{u_{j,k}^{n+1}}\right) = 0, m = 0, \ldots, r. \quad (2.33)$$

Let the solution $\tilde{u}_{h,k}^{n+1}$ to Eq. (2.33) be expressed as $\tilde{u}_{h,k}^{n+1} = \sum_{m=0}^{r} \tilde{c}_{m,k}^{(i)}(t_{n+1})\phi_m^{(i)}(\mathbf{x})$.

The constrained DG solution $u_{h,k}^{n+1} \in \mathrm{Span}\{\mathscr{B}_i\}$ defined on $\mathscr{K}_i$, which preserves the cell average $\overline{u_{i,k}^{n+1}}$, is obtained by modifying $\tilde{u}_{h,k}^{n+1}$ by

$$u_h^{n+1} \equiv \tilde{u}_h^{n+1} + \overline{u_i^{n+1}} - \frac{1}{|\mathscr{K}_i|}\int_{\mathscr{K}_i} \tilde{u}_h^{n+1} d\mathbf{x}. \quad (2.34)$$

Specifically,

$$c_{0,k}^{(i)}(t_{n+1}) = \tilde{c}_{0,k}^{(i)}(t_{n+1}) + \overline{u_{i,k}^{n+1}} - \frac{1}{|\mathscr{K}_i|}\int_{\mathscr{K}_i} \tilde{u}_{h,k}^{n+1} d\mathbf{x} \quad (2.35)$$

and

$$c_{m,k}^{(i)}(t_{n+1}) = \tilde{c}_{m,k}^{(i)}(t_{n+1}), \text{for} m = 1, \ldots, r. \quad (2.36)$$

Remark 4. It's again easy to verify that energy functional E_2 defined in Eq. (2.30) is invariant (subject to a scalar multiplication) under any affine change of coordinates.

Remark 5. Note that the linear system (2.33) consists of the same number of equations as in a RKDG step (2.27). Therefore the complexity of the 2D constrained DG method is close to that of the 2D RKDG (without using orthogonal basis functions).

The linear system (2.33) has a unique solution. In fact, consider the associate homogeneous system with $\mathscr{L}_{\mathscr{K}_i}(\upsilon_h) = 0$ and $\overline{u_{j,k}^{n+1}} = 0$ for all $j \in N(\mathscr{T}_{C,i})$,

$$\left(\frac{1}{|\mathscr{K}_i|}\right)\left(\int_{\mathscr{K}_i} \tilde{u}_{h,k}^{n+1} \upsilon_h d\mathbf{x}\right) + 2\mu \sum_{j \in N(\mathscr{T}_{C,i})} \left(\frac{1}{|\mathscr{K}_j|}\int_{\mathscr{K}_j} \upsilon_h d\mathbf{x}\right)\left(\frac{1}{|\mathscr{K}_j|}\int_{\mathscr{K}_j} \tilde{u}_{h,k}^{n+1} d\mathbf{x}\right) = 0, \quad (2.37)$$

for any $\upsilon_h \in \mathrm{Span}\{\mathscr{B}_i\}$. Let $\upsilon_h = \tilde{u}_{h,k}^{n+1}$. We conclude that $\int_{K_i} \left|\tilde{u}_{h,k}^{n+1}\right|^2 d\mathbf{x} = 0$, which implies $\tilde{u}_{h,k}^{n+1} \equiv 0$.

Remark 6. A compromised formulation with only one Lagrangian multiplier can be written as follows:

Find $\tilde{u}_h^{n+1} \in \mathrm{Span}\{\mathscr{B}_i\}$ such that

$$E_2(\tilde{u}_{h,k}^{n+1}) = \begin{array}{l} \mathrm{Min}\{E_2(\upsilon_h) : \upsilon_h \in \mathrm{Span}\{\mathscr{B}_i\}\} \\ \text{subject to} \frac{1}{|\mathscr{K}_i|}\int_{\mathscr{K}_i} \upsilon_h d\mathbf{x} = \overline{u_{i,k}^{n+1}}. \end{array} \quad (2.38)$$

This method has similar complexity and CFL numbers to that of the formulation (2.31) and 2.34.

2.3 Implementation

To summarize the steps taken to implement the constrained DG method for solving multi-D system of conservation laws, assume we employ a *s*-stage TVD Runge-Kutta method [28] to solve Eq. (2.24), which can be written in the form (neglecting subscript *k* of $u_{h,k}$ for convenience and when there is no confusion):

$$\int_{\mathscr{K}_i} u_h^{(j)} \upsilon_h d\mathbf{x} = \sum_{l=0}^{j-1} \alpha_{jl} \left(\int_{\mathscr{K}_i} u_h^{(l)} \upsilon_h d\mathbf{x} + \Delta t_n \frac{\beta_{jl}}{\alpha_{jl}} L(u_h^{(l)}, \upsilon_h) \right), j = 1, \ldots, s \equiv \sum_{l=0}^{j-1} \alpha_{jl} \int_{\mathscr{K}_i} u_h^{(j,l+1)} \upsilon_h d\mathbf{x}, \quad (2.39)$$

with

$$u_h^{(0)} = u_h^n, u_h^{(s)} = u_h^{n+1}. \quad (2.40)$$

Here α_{jl} and β_{jl} are coefficients of the Runge-Kutta method at the j^{th} stage, and

$$L(u_h, \upsilon_h) = -\int_{\partial \mathscr{K}_i} h_k \upsilon_h d\Gamma + \int_{\mathscr{K}_i} \mathbf{F}_k(\mathbf{u}_h) \cdot \nabla \upsilon_h d\mathbf{x}.$$

In particular, $u_h^{(j,l+1)} \in \mathrm{Span}\{\mathscr{B}_i\}$ is determined by

$$\int_{\mathscr{K}_i} u_h^{(j,l+1)} \upsilon_h d\mathbf{x} = \int_{\mathscr{K}_i} u^{(l)} \upsilon_h d\mathbf{x} + \Delta t_n \frac{\beta_{jl}}{\alpha_{jl}} L(u_h^{(l)}, \upsilon_h) \equiv \mathscr{L}(\upsilon_h), \forall \upsilon_h \in \mathrm{Span}\{\mathscr{B}_i\}.$$

This is a forward Euler scheme as in (2.27) with the time step size $\Delta t_n \beta_{jl}$, and will firstly be solved with $\upsilon_h = 1$ to obtain cell averages and subsequently be solved by being replaced similarly by the modification as in Eqs. (2.31) and (2.34).

This technique can also be applied to the classical 4th order Rung-Kutta method with the DG spatial discretization. The 4 stages are written as follows

$$
\begin{aligned}
&\int_{\mathscr{K}_i} u_h^{n+1/2-} \upsilon_h d\mathbf{x} \\
&\quad = \int_{\mathscr{K}_i} u_h^n \upsilon_h d\mathbf{x} \\
&\quad - \frac{1}{2}\Delta t_n \int_{\partial \mathscr{K}_i} h_k^n \upsilon_h d\Gamma \\
&\quad + \frac{1}{2}\Delta t_n \int_{\mathscr{K}_i} \mathbf{F}_k^n(\mathbf{u}_h) \\
&\quad \cdot \nabla \upsilon_h d\mathbf{x} \equiv \mathscr{L}_1(\upsilon_h), \int_{\mathscr{K}_i} u_h^{n+1/2+} \upsilon_h d\mathbf{x} \\
&\quad = \int_{\mathscr{K}_i} u_h^n \upsilon_h d\mathbf{x} \\
&\quad - \frac{1}{2}\Delta t_n \int_{\partial \mathscr{K}_i} h_k^{n+1/2-} \upsilon_h d\Gamma \\
&\quad + \frac{1}{2}\Delta t_n \int_{\mathscr{K}_i} \mathbf{F}_k^{n+1/2-}(\mathbf{u}_h) \\
&\quad \cdot \nabla \upsilon_h d\mathbf{x} \equiv \mathscr{L}_2(\upsilon_h), \int_{\mathscr{K}_i} u_h^{n+1-} \upsilon_h d\mathbf{x} = \int_{\mathscr{K}_i} u_h^n \upsilon_h d\mathbf{x} - \Delta t_n \int_{\partial \mathscr{K}_i} h_k^{n+1/2+} \upsilon_h d\Gamma + \Delta t_n \int_{\mathscr{K}_i} \mathbf{F}_k^{n+1/2+}(\mathbf{u}_h) \\
&\quad \cdot \nabla \upsilon_h d\mathbf{x} \equiv \mathscr{L}_3(\upsilon_h), \int_{\mathscr{K}_i} u_h^{n+1} \upsilon_h d\mathbf{x} \\
&\quad = \int_{\mathscr{K}_i} u_h^n \upsilon_h d\mathbf{x} \\
&\quad - \Delta t_n \int_{\partial \mathscr{K}_i} h_k^* \upsilon_h d\Gamma \\
&\quad + \Delta t_n \int_{\mathscr{K}_i} \mathbf{F}_k^*(\mathbf{u}_h) \\
&\quad \cdot \nabla \upsilon_h d\mathbf{x} \equiv \mathscr{L}_4(\upsilon_h),
\end{aligned}
\tag{2.41}
$$

where $h_k^{n+1/2-}$ denotes the numerical flux evaluated with $u_h^{n+1/2-}$ from the previous stage and

$$h_k^* = \frac{1}{6}\left(h_k^n + 2h_k^{n+1/2-} + 2h_k^{n+1/2+} + h_k^{n+1-}\right),$$

similarly for $\mathbf{F}_k^{n+1/2-}$, $\mathbf{F}_k^{n+1/2+}$ and other fluxes. The modification (2.31) and (2.34) that has been applied to $\mathscr{L}$ in equation (2.28) can be applied to $\mathscr{L}_1$, $\mathscr{L}_2$, $\mathscr{L}_3$ and $\mathscr{L}_4$ to modify the values of $u_h^{n+1/2-}$, $u_h^{n+1/2+}$, u_h^{n+1-} and u_h^{n+1} respectively. Similar modification can be applied to each of the *s* stages in (2.39) (rather than to its forward Euler schemes) to reduce the number of changes.

2.3.1 Choices of cells being used as constraints—Here we present several choices of the set $\mathscr{T}_{C,i}$ defined in Eq. (2.32) of cells that are used as constraints. In general, any of adjacent cells can be used as constraint cells. A bigger CFL number can be obtained with respect to using more constraint cells; while the numerical error in the solution also increases slightly with more constraint cells. See also Sec. 3 for numerical test results showing this trend.

For the 1D third- and fourth-order accurate constrained RKDG schemes, the set of constraint cells for solving for the solution supported on $\mathscr{I}_i$ cell is $\{\mathscr{I}_{i-1}, \mathscr{I}_i, \mathscr{I}_{i+1}\}$. The resulting 1D third- and fourth-order accurate constrained RKDG schemes are denoted as "1D Constrained RKDG3-3Cell" and "1D Constrained RKDG4-3Cell" in the following sections, respectively.

For the 2D third-order accurate constrained RKDG scheme, if we want to solve for the solution supported on cell $\mathscr{K}_0$ in Fig. 1, we choose either $\mathscr{T}_{C,0} = \{\mathscr{K}_0, \mathscr{K}_1, \mathscr{K}_2, \mathscr{K}_3\}$ or $\mathscr{T}_{C,0} = \{\mathscr{K}_0, \mathscr{K}_1, \mathscr{K}_2, \mathscr{K}_3, \mathscr{K}_6, \mathscr{K}_9, \mathscr{K}_{12}\}$. The resulting schemes are termed as "2D Constrained RKDG3-4Cell" and "2D Constrained RKDG3-7Cell", respectively.

For the solution supported on cell $\mathscr{K}_0$ in Fig. 1 and computed by the fourth-order accurate constrained DG scheme, we select $\mathscr{T}_{C,0} = \{\mathscr{K}_0, \mathscr{K}_1, \mathscr{K}_2, \mathscr{K}_3, \mathscr{K}_6, \mathscr{K}_9, \mathscr{K}_{12}\}$, $\mathscr{T}_{C,0} = \{\mathscr{K}_0, \mathscr{K}_1, \mathscr{K}_2, \mathscr{K}_3, \mathscr{K}_4, \mathscr{K}_5, \mathscr{K}_7, \mathscr{K}_8, \mathscr{K}_{10}, \mathscr{K}_{11}\}$, or $\mathscr{T}_{C,0} = \{\mathscr{K}_0, \mathscr{K}_1, \ldots, \mathscr{K}_{12}\}$. The resulting schemes are denoted as "2D Constrained RKDG4-7Cell", "2D Constrained RKDG4-10Cell" and '2D Constrained RKDG4-13Cell", respectively.

We note there are many other possible ways to select constraint cells. In the present work, we only test these aforementioned sets of constraint cells. We also note that when one side of a cell overlaps with the physical boundary of the domain and this cell does not have neighboring cells on this side, we simply include available edge and vertex adjacent cells which are inside the physical domain to construct $\mathscr{T}_{C,i}$. When this is the case, the collection of the constraint cells is not symmetric around the cell. In practice, the CFL number has to be reduced as well because of this.

Finally, we abbreviate the third- and fourth-order accurate RKDG schemes by "RKDG3" and "RKDG4" for the sake of convenience, respectively.

2.4 Analytical estimate of the CFL numbers

We use Fourier transform to analyze CFL numbers of the new constrained DG method for solving the 1D linear advection equation $u_t + u_x = 0$.

Consider a uniform partition $\{x_{i+1/2}\}$ on $\{-\infty, +\infty\}$ with the mesh size $\Delta x = x_{i+1/2} - x_{i-1/2}$, and cell centroid $x_i = (x_{i-1/2} + x_{i+1/2})/2$, $i = 0, \pm1, \pm2, \ldots$. Following [36, 19], we express a polynomial $u_h^n|_{\mathscr{K}_i}$ of degree r on cell $\mathscr{K}_i = (x_{i-1/2}, x_{i+1/2})$ as a linear combination of Lagrangian basis functions $\{l_j(x-x_i) : j = 0, 1, \ldots, r\}$ corresponding to an evenly distributed set of $r + 1$ nodes of Lagrangian interpolation $\{y_{i,j} : j = 0, 1, \ldots, r\}$, symmetric about x_i in cell $\mathscr{K}_i$, with $y_{i,j} \in (x_{i-1/2}, x_{i+1/2})$,

$$u_h^n|_{\mathscr{K}_i}(x) = \sum_{j=0}^{r} \xi_{i,j}^n l_j(x - x_i), \quad (2.42)$$

where $\xi_{i,j}^n$ is the coefficient in front of the j-th Lagrangian basis function, $\xi_{i,j}^n = u_h^n|_{\mathscr{K}_i}(y_{i,j})$. Now we are able to express a fully discrete conservation constrained RKDG method as

$$\boldsymbol{\xi}_i^{n+1} = \sum_{j=-s}^{s} A_j \boldsymbol{\xi}_{i+j}^n, \quad (2.43)$$

where $\boldsymbol{\xi}_i^{n+1} = (\xi_{i,0}^{n+1}, \xi_{i,1}^{n+1}, \cdots, \xi_{i,r}^{n+1})^T$, A_j is a $(r + 1) \times (r + 1)$ matrix, $j = -s, \ldots, s$, and s is a positive integer. Applying a (discrete) Fourier transform yields the following form

$$\hat{\xi}^{n+1} = B\hat{\xi}^{n}, \quad (2.44)$$

where B is a $(r+1) \times (r+1)$ matrix (the Fourier symbol). The CFL number of the scheme can be estimated from the spectral radius of matrix B. Mathematica is used for symbolic and numerical computation of the above procedure. In Tables 1–5, in which "p-w" stands for piecewise, we estimate the CFL numbers for $\mu = 0$, 0.01, 0.5, 100 and 500 (see Eqs. (2.31) and (2.30)), recalling that $\mu = 0$ represents the standard RKDG method. The TVD Runge-Kutta methods are used for the second- and third-order accurate time discretization, and the classical fourth-order accurate Runge-Kutta method is used for the fourth-order accurate time discretization. We observe about three times or more greater CFL numbers compared to those of the standard RKDG methods ($\mu = 0$), and that parameter μ is not sensitive for the new method as long as it is larger than a number, e.g., 0.5. In Tables 6 and 7, we remove the constraint from the right (upwind-type) and left neighboring cell respectively. Both have better CFL numbers compared to the case of $\mu = 0$. However, the upwind-type constraint doesn't seem to work significantly better than the downwind-type. Using constraints from both left and right neighboring cells seems to perform the best.

For the case of piecewise linear spatial space ($r = 1$) and forward Euler temporal discretization, we write down formula (2.43) explicitly with $\mu = 1$ as follows to compare with those of the standard RKDG method ($\mu = 0$).

$$\begin{aligned}\boldsymbol{\xi}_i^{n+1} = &\begin{pmatrix} -0.0612245\lambda & 0.183673\lambda \\ 0.0612245\lambda & -0.183673\lambda \end{pmatrix} \boldsymbol{\xi}_{i-2}^{n} \\ &+ \begin{pmatrix} 0.0612245 - 0.454082\lambda & 0.0612245 + 1.36224\lambda \\ -0.0612245 - 0.545918\lambda & -0.0612245 + 1.63776\lambda \end{pmatrix} \boldsymbol{\xi}_{i-1}^{n} + \begin{pmatrix} 0.510204 + 0.515306\lambda & 0.489796 - 1.66837\lambda \\ 0.489796 + 0.484694\lambda & 0.510204 - 1.33163\lambda \end{pmatrix} \boldsymbol{\xi}_{i}^{n} \\ &+ \begin{pmatrix} -0.0612245 - 0.0612245\lambda & -0.0612245 + 0.183673\lambda \\ 0.0612245 + 0.0612245\lambda & 0.0612245 - 0.183673\lambda \end{pmatrix} \boldsymbol{\xi}_{i+1}^{n}, \end{aligned} \quad (2.45$$

where $\lambda = \Delta t/\Delta x$.

As a comparison, the corresponding standard DG method ($\mu = 0$) yields the following formula.

$$\boldsymbol{\xi}_i^{n+1} = \begin{pmatrix} -1.25\lambda & 3.75\lambda \\ 0.25\lambda & -0.75\lambda \end{pmatrix} \boldsymbol{\xi}_{i-1}^{n} + \begin{pmatrix} 1 - 1.75\lambda & -0.75\lambda \\ 2.75\lambda & 1 - 2.25\lambda \end{pmatrix} \boldsymbol{\xi}_{i}^{n}. \quad (2.46)$$

The Fourier symbol of (2.45) is $(\rho_{ij})_{2\times 2}$ where

$$\begin{aligned} \rho_{11} &= 0.510204 + 0.515306\lambda - 0.515306\lambda\cos\eta - 0.0612245\lambda\cos 2\eta + i[0.392857\lambda\sin\eta + 0.0612245\lambda\sin 2\eta - 0.122449\sin\eta], \\ \rho_{12} &= 0.489796 - 1.66837\lambda + 1.54592\lambda\cos\eta + 0.183673\lambda\cos 2\eta + i[0.183673\lambda\sin 2\eta - 1.17857\lambda\sin\eta - 0.122449\sin\eta], \\ \rho_{21} &= 0.489796 + 0.484694\lambda - 0.484694\lambda\cos\eta + 0.0612245\lambda\cos 2\eta + i[0.122449\sin\eta + 0.607143\lambda\sin\eta - 0.0612245\lambda\sin 2\eta], \\ \rho_{22} &= 0.510204 - 1.33163\lambda + 1.45408\lambda\cos\eta - 0.183673\lambda\cos 2\eta + i[0.122449\sin\eta - 1.82143\lambda\sin\eta + 0.183673\lambda\sin 2\eta], \end{aligned} \quad (2.47)$$

$\eta = \xi\Delta x$ and ξ is the Fourier dual variable. On the other hand, the Fourier symbol of (2.46) looks like

$$\begin{pmatrix} 1-1.75\lambda-1.25\lambda\cos\eta+1.25i\lambda\sin\eta & -0.75\lambda+3.75\lambda\cos\eta-3.75i\lambda\sin\eta \\ 2.75\lambda+0.25\lambda\cos\eta-0.25i\lambda\sin\eta & 1-2.25\lambda-0.75\lambda\cos\eta+0.75i\lambda\sin\eta \end{pmatrix}. \quad (2.48)$$

Remark 7. Even though the new constrained DG method seems to have larger domain of dependence compared to the standard DG method, each of its explicit component steps is a compact method when neighboring cell averages are used for the constraint. A finite volume scheme based on reconstruction from cell averages generally has a larger CFL number than the standard RKDG method with corresponding order. This motivates us to incorporate the constraint from neighboring cell averages into the DG method without losing its compactness in implementation. For linear stability of the RKDG method, we refer readers to [10, 11] and references therein for more details.

2.5 Limiting by hierarchical reconstruction

To prevent non-physical oscillations in the vicinity of discontinuities, we apply HR with partial neighboring cells [33] to the solution computed at each of the Runge-Kutta stages. Since shock waves or contact discontinuities are all local phenomena, we apply the HR limiting procedure to a small region covering discontinuities. Specifically, we employ a detector introduced in [6] to identify cells which may contain oscillatory solutions. HR with partial neighboring cells is then applied to solutions supported on these cells. We first sketch the 2D HR with partial neighboring cells limiting procedure here. More details can be found in [33]. We then describe an extension of the HR limiting by using characteristic decomposition and local iteration.

HR initially introduced in [17, 18] decomposes the job of limiting a high-order polynomial supported on a cell, which may be spuriously oscillatory into a series of smaller jobs, each of which only involves the non-oscillatory reconstruction of a linear polynomial. This linear polynomial reconstruction can be easily achieved through classical processes such as the MUSCL reconstruction [13, 14, 15] used in [17], or a WENO-type combination used in [33]. Since the reconstruction of a linear polynomial can only use information from adjacent cells, HR can be formulated in multi-dimensions on a compact stencil. Using the basis function set (2.22), the approximate solution $u_h(\mathbf{x}-\mathbf{x}_i)$ on cell $\mathscr{K}_i$ is represented as

$$u_h(\mathbf{x}-\mathbf{x}_i)=\sum_{|\mathbf{m}|=0}^{q} c_{\mathbf{m}}\frac{(\mathbf{x}-\mathbf{x}_i)^{\mathbf{m}}}{(\sqrt{|\mathscr{K}_i|})^{|\mathbf{m}|}}. \quad (2.49)$$

Here $\mathbf{m}$ is an 2-tuple, and $\mathbf{m}\in\mathbb{N}_0^2$.

$u_h(\mathbf{x}-\mathbf{x}_i)$ may contain spurious oscillations. The HR procedure is to recompute the coefficients of polynomial $u_h(\mathbf{x}-\mathbf{x}_i)$ by using polynomials in cells (or partial neighboring cells [33]) adjacent to $\mathscr{K}_i$. These adjacent cells (or partial cells) are collected as the set $\{\mathscr{K}_j\}$

(which also contains cell $\mathscr{K}_i$) and the polynomials (of degree q) supported on them are denoted as $\{u_{h,j}(\mathbf{x}-\mathbf{x}_j)\}$ respectively. HR recomputes a set of new coefficients

$$\hat{c_{\mathbf{m}}}, \text{with} |\mathbf{m}|=q, q-1, \ldots, 0$$

to replace the original coefficients $c_{\mathbf{m}}$ of $u_h(\mathbf{x}-\mathbf{x}_i)$ iteratively from the highest to the lowest degree terms without losing the order of accuracy if the piecewise polynomial solution is locally smooth, and eliminates spurious oscillations of $u_h(\mathbf{x}-\mathbf{x}_i)$ otherwise.

To obtain $\hat{c_{\mathbf{m}}}$, we first compute *candidates* of $c_{\mathbf{m}}$, and then let the value of $\hat{c_{\mathbf{m}}}$ be

$$\hat{c_{\mathbf{m}}}=F(candidates\ \text{of} c_{\mathbf{m}}),$$

where F is a convex limiter of its arguments (e.g., the center biased minmod function used in [18], or the WENO-type combination in [33], where $F(a_1, a_2, \cdots, a_1)=\sum_{i=1}^{l}\theta_i a_i$, for some $\theta_i \geq 0$ and $\sum_{i=1}^{l}\theta_i=1$).

In order to find these *candidates* of $c_{\mathbf{m}}$, $|\mathbf{m}| = m$, with $1 \leq m \leq q$, we take a $(m-1)^{th}$ order partial derivative of $u_h(\mathbf{x}-\mathbf{x}_i)$ (and also polynomials in adjacent cells or partial cells), and express

$$\partial^{m-1}u_h(\mathbf{x}-\mathbf{x}_i)=L_h(\mathbf{x}-\mathbf{x}_i)+R_h(\mathbf{x}-\mathbf{x}_i),$$

where L_h is the linear part (containing the zeroth and first degree terms) and R_h is the remainder. Clearly, every coefficient in the first degree terms of L_h is in the set $\{c_{\mathbf{m}} : |\mathbf{m}| = m\}$.

In general, for every $\mathbf{m}$ subject to $|\mathbf{m}| = m$, $1 \leq m \leq q$, one can always take some $(m-1)^{th}$ order partial derivatives of $u_h(\mathbf{x}-\mathbf{x}_i)$ so that $c_{\mathbf{m}}$ is a coefficient in a first degree term of L_h. Thus, a "candidate" for a coefficient in a first degree term of L_h to be reconstructed is also the *candidate* for the corresponding $c_{\mathbf{m}}$.

In order to find a set of *candidates* for all coefficients in the first degree terms of $L_h(\mathbf{x}-\mathbf{x}_i)$, we need to know the new approximate cell averages of $L_h(\mathbf{x}-\mathbf{x}_i)$ on $d+1$ distinct mesh cells adjacent to cell $\mathscr{K}_i$, which is a key step. Assume $\mathscr{K}_{j0}, \mathscr{K}_{j1}, \cdots, \mathscr{K}_{jd} \in \{\mathscr{K}_j\}$ are these cells or partial cells and $\bar{L}_{j0}, \bar{L}_{j1}, \cdots, \bar{L}_{jd}$ are the corresponding new approximate cell averages. For example, in order to obtain $\bar{L}_{j1}$, we first compute

$$\bar{A}_{j_1}=\frac{1}{|\mathscr{K}_{j_1}|}\int_{\mathscr{K}_{j_1}}\partial^{m-1}u_{h,j_1}(\mathbf{x}-\mathbf{x}_{j_1})d\mathbf{x},$$

then

$$\bar{D}_{j_1}=\frac{1}{|\mathscr{K}_{j_1}|}\int_{\mathscr{K}_{j_1}}\hat{R}_h(\mathbf{x}-\mathbf{x}_i)d\mathbf{x},$$

where $\hat{R}_h(\mathbf{x}-\mathbf{x}_i)$ is the $R_h(\mathbf{x}-\mathbf{x}_i)$ with its coefficients replaced by previously computed new values. We can set $\bar{L}_{j1}=\bar{A}_{j1}-\bar{D}_{j1}$. Finally, a non-oscillatory reconstruction procedure [33] is applied to $\bar{L}_{j0}, \bar{L}_{j1}, \cdots, \bar{L}_{jd}$ to obtain *candidates* of $c_{\mathbf{m}}$ in the first degree terms of $L_h(\mathbf{x}-\mathbf{x}_i)$.

When $m=0$, the modified 0th degree coefficient $\hat{c_0}$ is chosen such that the cell average of the reconstructed polynomial is the same as cell average of the original $u_h(\mathbf{x}-\mathbf{x}_i)$.

More details of the HR implementations and related techniques can be found in [17, 18, 20, 33, 34, 35].

2.5.1 Hierarchical reconstruction using characteristic decomposition and local iteration—For all third-order schemes considered in the paper, we use component by component HR limiting with partial neighboring cells technique [33]. However, we noticed that resolutions of solutions to some Euler equations' test problems computed by fourth-order schemes were compromised when component by component HR limiting with partial neighboring cells was used. We introduce a HR using characteristic decomposition and local iteration to improve the resolution of the numerical solution computed by fourth-order schemes.

Let the approximate solution $\mathbf{u}_h(\mathbf{x}-\mathbf{x}_i)$ to system (2.1) be supported on cell $\mathscr{K}_i$. Let the outward unit normal on one side of $\mathscr{K}_i$ be $(n_{x,0}, n_{y,0})$; and the edge-adjacent cell of $\mathscr{K}_i$ on this side be $\mathscr{K}_{k0}$. We compute the average Jacobian A_0 by using the cell average values of $\mathscr{K}_i$ and $\mathscr{K}_{k0}$,

$$A_0=n_{x,0}\frac{\partial\mathbf{f}}{\partial\mathbf{u}}+n_{y,0}\frac{\partial\mathbf{g}}{\partial\mathbf{u}}, \quad (2.50)$$

where $\mathbf{f}=(F_{1,1}(\mathbf{u}), F_{2,1}(\mathbf{u}), \cdots, F_{p,1}(\mathbf{u}))^T$ and $\mathbf{g}=(F_{1,2}(\mathbf{u}), F_{2,2}(\mathbf{u}), \cdots, F_{p,2}(\mathbf{u}))^T$, with F being defined in Eq. (2.1). The Roe's mean matrix is used for Euler equations [25]. Let R_{A_0} be the matrix of right eigenvectors and L_{A_0} be the matrix of left eigenvectors of A_0, respectively. We project $\mathbf{u}_h(\mathbf{x}-\mathbf{x}_i)$ and $\{\mathbf{u}_{h,j}(\mathbf{x}-\mathbf{x}_j)\}$ supported on adjacent partial cells of $\mathscr{K}_i$ to obtain characteristic fields

$$\mathbf{v}_h(\mathbf{x}-\mathbf{x}_i)=L_{A_0}\mathbf{u}_h(\mathbf{x}-\mathbf{x}_i)$$

and

$$\mathbf{v}_{h,j}(\mathbf{x}-\mathbf{x}_j)=L_{A_0}\mathbf{u}_{h,j}(\mathbf{x}-\mathbf{x}_j),$$

respectively. Then the HR limiting with partial neighboring cells technique is applied to each of the characteristic fields to reconstruct a new $\hat{\mathbf{v}}_{h,0}(\mathbf{x}-\mathbf{x}_i)$. We next define

$$\hat{\mathbf{u}}_{h,0}(\mathbf{x}-\mathbf{x}_i)=R_{A_0}\hat{\mathbf{v}}_{h,0}(\mathbf{x}-\mathbf{x}_i).$$

We repeat this procedure on each side of the $\mathscr{K}_i$ to obtain three reconstructed polynomials, and denote them by $\hat{\mathbf{u}}_{h,0}(\mathbf{x}-\mathbf{x}_i)$, $\hat{\mathbf{u}}_{h,1}(\mathbf{x}-\mathbf{x}_i)$ and $\hat{\mathbf{u}}_{h,2}(\mathbf{x}-\mathbf{x}_i)$, respectively. Finally, each of the components of $\hat{\mathbf{u}}_{h,0}$, $\hat{\mathbf{u}}_{h,1}$ and $\hat{\mathbf{u}}_{h,2}$ are combined with weights introduced in [26] to obtain a new reconstructed $\mathbf{u}_h(\mathbf{x}-\mathbf{x}_i)$.

Next we introduce a local iteration technique to further reduce the small over/undershoots. For simplicity, we develop the local iteration technique for the case of scalar solution $u_h(\mathbf{x}-\mathbf{x}_i)$ (application of local iteration to system solution $\mathbf{u}_h(\mathbf{x}-\mathbf{x}_i)$ is similar). Assume we need to reconstruct the approximate scalar solution $u_h(\mathbf{x}-\mathbf{x}_i)$ supported on cell $\mathscr{K}_i$ by HR. When all new values of the coefficients of $u_h(\mathbf{x}-\mathbf{x}_i)$ have been computed, we will update $u_h(\mathbf{x}-\mathbf{x}_i)$ while keeping the solution on its neighboring cells unchanged. This leads to

$$u_h(\mathbf{x}-\mathbf{x}_i)=\sum_{|\mathbf{m}|=0}^{q}\hat{c_{\mathbf{m}}}\frac{(\mathbf{x}-\mathbf{x}_i)^{\mathbf{m}}}{(\sqrt{|\mathscr{K}_i|})^{|\mathbf{m}|}}. \quad (2.51)$$

We apply HR again to reconstruct (2.51). In other words, we apply HR twice (or more times if necessary) to update $u_h(\mathbf{x}-\mathbf{x}_i)$ with the solutions on its neighboring cells being temporally fixed. Oscillations in solution are partially due to large coefficient values associated with high degree terms of the polynomial representation of the solution. By using local iteration, magnitudes of coefficients of high degree terms can be even more lowered. Thus the local iteration technique can further reduce the possible remaining over/under-shoot without spreading out the diffusion.

Remark 8. We note that the HR limiting itself does not guarantee the positivity. When negative pressure is detected at a quadrature point of a cell after limiting, a simple scaling technique introduced in [33] is used to remove the negative pressure.

3 Numerical Examples

3.1 Accuracy test using 1D linear advection equation

We first test the capability of the constrained RKDG method to achieve the desired order of accuracy with a large CFL number, using the 1D linear advection equation

$$u_t+u_x=0,(x,t)\in(-1,1)\times(0,\mathrm{T}) \quad (3.1)$$

with periodic boundary conditions and the initial condition

$$u(x,t=0)=\frac{1}{2}+\sin(\pi x),-1\le x\le 1. \quad (3.2)$$

The uniform mesh is used to solve this test problem. The solution is computed up to time T = 2.0. The cell size, denoted by Δx, is listed in tables shown in this sub-section. Table 8

shows that the third-order accurate constrained RKDG scheme (1D Constrained RKDG3-3Cell) is stable with that the CFL number is equal to 1.6; while Table 10 shows that the fourth-order accurate constrained RKDG scheme (1D Constrained RKDG4-3Cell) is stable with that the CFL number equals 0.6. These results are in agreement with results by analytic estimate. We tested the maximum values of the CFL number that the third- and fourth-order accurate RKDG schemes can use numerically. We observed that these maximum values are around 0.2 and 0.1, respectively, which are also consistent with the analytic result shown in Table 1. Tables 8 and 10 show these numerical test results as well.

We studied how CFL number affects the errors in the constrained DG solution. Table 9 shows the L^1 and L^∞ errors of the solution computed by 1D Constrained RKDG3-3Cell method using CFL = 0.2; while Table 11 lists the L^1 and L^∞ errors of the solution computed by 1D Constrained RKDG4-3Cell method using CFL = 0.1. We notice that reducing the CFL number used by the 1D constrained DG methods can reduce magnitudes of L^1 and L^∞ errors by about 3 times. We also report that for the 2D test problems with smooth solutions presented in this paper, we do not observe that reducing the CFL number also significantly lowers the magnitude of errors in solutions computed by the 2D constrained DG methods.

We also studied the how the conservation penalty weight μ affects the CFL numbers that can be used numerically. Tables 12 and 13 list the accuracy test results with μ = 0.5, 5, 500 for the third- and fourth-order accurate constrained RKDG schemes, respectively. It is clear that the allowed maximum CFL numbers for these two schemes are not affected by the choice of μ values. Additionally, the L^1 and L^∞ errors do not seem to be affected by the μ values as well.

3.2 Accuracy test using using 1D Burgers' equation with a smooth solution

Here we test the maximum CFL number that the 1D constrained RKDG method can achieve by using the 1D Burgers' equation

$$u_t+\left(\frac{1}{2}u^2\right)_x=0,(x,t)\in(-1,1)\times(0,\mathrm{T}), \quad (3.3)$$

with a periodic boundary condition and the initial condition

$$u(x,t=0)=\frac{1}{2}+\sin(\pi x),-1\le x\le 1. \quad (3.4)$$

The uniform mesh is used to solve this test problem. The solution is computed up to time T = 0.5/π, when it is still smooth. Δ*x* listed in tables shown in this sub-section is the cell size.

Table 14 shows that the third-order accurate 1D Constrained RKDG3-3Cell scheme can use a CFL number being equal to 1.6; while Table 15 shows that the fourth-order accurate 1D Constrained RKDG4-3Cell scheme is stable with a CFL number equaling 0.6 for this 1D nonlinear problem test case.

Similar to the 1D linear advection equation test case demonstrated in Sec. 3.1, the numerical study of the choice of μ values using 1D Burgers' equation test case shows that the allowed

maximum CFL numbers and L^1 and L^∞ errors of numerical solutions are not sensitive to μ values as well. These results are summarized in Tables 16 and 17, respectively.

3.3 Accuracy test using 2D linear advection equation

To assess the maximum CFL number that the constrained RKDG methods can use on 2D triangular meshes, we start with solving the following initial-boundary-value problem of the 2D linear advection equation

$$\begin{array}{ll} u_t+u_x+u_y=0, & (x,y,t)\in\Omega\times(0,\mathrm{T}) \\ u(x,y,t=0)=\frac{1}{4}+\frac{1}{2}\sin(\pi(x+y)), & (x,y)\in\Omega. \end{array} \quad (3.5)$$

The domain Ω is the square [−1, 1] × [−1, 1]. The periodic boundary condition is used in both directions. For the convenience of implementing the periodic boundary condition, the triangular mesh is obtained by perturbing a uniform triangulation. See Fig. 2 for a typical mesh used for the accuracy test. We adopt the following definition of the CFL number for this test case:

$$\text{Maximum of}\left\{\frac{\Delta t}{\mathscr{D}}\right\}, \quad (3.6)$$

where $\mathscr{D}$ is the diameter of the inscribed circle of a triangle.

We computed the solution up to time T = 2.0. The typical triangle edge length, denoted by *h*, is listed in tables shown in this section. The errors presented are for *u*.

Table 18 shows that the 2D third-order accurate constrained RKDG scheme using 7 cells as constraints (2D Constrained RKDG3-7Cell) is stable when CFL number is equal to 0.8; and the third-order accurate RKDG scheme is stable when CFL number is around 0.22.

Table 19 shows that the 2D fourth-order accurate constrained RKDG scheme using 10 cells as constraints (2D Constrained RKDG4-10Cell) is stable when CFL number is equal to 0.9; while the numerical test shows that the fourth-order accurate RKDG scheme is stable when CFL number is around 0.2.

We studied how penalty constant μ affects magnitudes of errors for this 2D linear equation test case. Tables 20 and 21 show that the L^1 and L^∞ errors computed by the 2D Constrained RKDG3-7Cell and RKDG4-10Cell schemes respectively increase only slightly when μ varies between 0.5 and 500. Thus both 2D schemes are not sensitive for the choice of μ value.

We tested how choices of constraint cells affect the CFL number used by 2D fourth-order accurate constrained DG schemes. When 7 cells are used as constraints, the maximum CFL number the 2D Constrained RKDG4-7Cell scheme can take is around 0.35. When 13 cells are used as constraints, the maximum CFL number that can be reached by the 2D Constrained RKDG4-13Cell scheme is about 1.3. However, the differences between magnitudes of L^1 and L^∞ errors computed by these 2D fourth-order accurate constrained

RKDG schemes (with 7, 10 and 13 cells used as constraints respectively) are small. See Tables 22 and 19 for this conclusion.

3.4 Accuracy test using 2D Burgers' equation with a smooth solution

To assess the limit of the permissible CFL number used by the constrained RKDG method for solving 2D nonlinear scalar conservation laws, we solve the following initial-boundary-value problem of the 2D Burgers' equation

$$\begin{aligned} &u_t+\left(\tfrac{1}{2}u^2\right)_x+\left(\tfrac{1}{2}u^2\right)_y=0, && (x,y)\in\Omega\times(0,\mathrm{T})\\ &u(x,y,t{=}0)=\tfrac{1}{4}+\tfrac{1}{2}\sin(\pi(x+y)), && (x,y)\in\Omega, \end{aligned} \tag{3.7}$$

where Ω = [−1, 1] × [−1, 1]. The periodic boundary condition is used in both directions. The solution is computed up to T = 0.5/π, when it is still smooth. The triangular meshes utilized for the 2D linear advection equation test are also used for this convergence test.

We use the following definition of the CFL number for this test case:

$$\text{Maximum of}\left\{\frac{\Delta t|u|}{\mathscr{D}}\right\}, \tag{3.8}$$

where $\mathscr{D}$ is the diameter of the inscribed circle of a triangle. |*u*| is evaluated by the local cell average value. The errors presented in tables shown in this sub-section are for *u*. *h* listed in these tables is the edge length of the triangle.

Table 23 shows the L^1 and L^∞ errors and numerical orders of accuracy for using the 2D third-order accurate constrained RKDG scheme with 7 constraint cells (2D Constrained RKDG3-7Cell) for solving Eq. (3.7). With μ = 0.5, we are able to use a CFL number = 0.8 for computing the solution while achieving the desired order of accuracy. In this table, we also show that the maximum CFL number that the 2D third-order accurate RKDG scheme can use is about 0.22 by our numerical test. Table 24 shows that the 2D Constrained RKDG3-7Cell scheme is not sensitive to the choice of μ values.

Table 25 shows that when 4 cells are used as constraints, the 2D third-order accurate constrained RKDG scheme (2D Constrained RKDG3-4Cell) is permitted to use a CFL number = 0.3 for the nonlinear equation test case. In addition, the L^1 and L^∞ errors of the numerical solutions to Eq. (3.7) computed by the 2D Constrained RKDG3-4Cell scheme is about 3 ~ 4 times smaller than the ones of the solutions computed by the 2D Constrained RKDG3-7Cell scheme.

In Table 26, we demonstrate the L^1 and L^∞ errors and numerical orders of accuracy for using the 2D fourth-order accurate constrained RKDG scheme with 10 constraint cells (2D Constrained RKDG4-10Cell) and standard fourth-order accurate RKDG scheme to solve Eq. (3.7), respectively. With μ = 0.5, the 2D Constrained RKDG4-10Cell scheme is capable of using a CFL number = 0.9 for achieving the fourth-order accuracy; while the maximum CFL number that the 2D fourth-order accurate RKDG scheme can use is about 0.25 by our numerical test. Table 27 shows that the 2D Constrained RKDG4-10Cell scheme is not sensitive to the choice of μ values.

We also tested performance of the fourth-order accurate constrained RKDG scheme using different choices of constraint cells for μ = 0.5. Table 28 shows that when 7 cells are used as constraints, the 2D Constrained RKDG4-7Cell scheme is allowed to use a CFL number = 0.35; while the 2D Constrained RKDG4-13Cell scheme which uses 13 constraint cells can use a CFL number as big as 1.4 for this nonlinear equation test case.

From Tables 26–28, we notice that the L^1 and L^∞ errors of the numerical solutions to Eq. (3.7) computed by the 2D Constrained RKDG4-7Cell scheme is about 2 ~ 4 times smaller than the ones computed by the 2D Constrained RKDG4-13Cell method. And the L^1 and L^∞ errors of 2D Constrained RKDG4-10Cell scheme and 2D Constrained RKDG4-13Cell scheme are comparable.

In Fig. 3, the CPU times and L^1 errors are presented for solving Eq. (3.7) up to time T = $0.5/\pi$. We can clearly see that RKDG method obtains better accuracy using less CPU time for solving the smooth solution problem. To achieve the same magnitude of the error, 27 the constrained RKDG method takes about two times more CPU time than that of the RKDG method. In Fig. 4, we plot the CPU times and L^1 errors of these schemes for solving Eq. (3.7) up to time T = 0.45, when the shock wave has formed. The 3rd- and 4th-order HR limiters described in Sec. 2.5 are applied in the vicinity of the shock to limit the 3rd- and 4th-order solutions, respectively. When a shock wave forms, the total error is dominated by the one generated in the vicinity of the discontinuity. We compute the errors in the region $[-1, -0.45] \times [-1, -0.45]$, which is close to the location of the shock. We can see that the magnitudes of the L^1 errors computed by the RKDG and constrained RKDG schemes are comparable and the constrained RKDG scheme uses less CPU time.

To summarize, we found that the 2D constrained RKDG schemes can use a CFL number 3 ~ 4 times or more greater than the one that 2D RKDG schemes can take. And this increase is not sensitive to the μ value.

3.5 Test cases using 1D Euler equations with discontinuous solutions

We now assess the resolution and the non-oscillatory property of 1D numerical solutions computed by the constrained RKDG method and limited by HR. In this sub-section, we compute solutions of various shock tube problems modeled by the 1D Euler equations

$$\mathbf{u}_t + \mathbf{F}(\mathbf{u})_x = 0$$

with $\mathbf{u} = (\rho, \rho v, E)^T$, $\mathbf{F}(\mathbf{u}) = (\rho v, \rho v^2 + p, v(E+p))^T$, $p = (\gamma - 1)(E - \frac{1}{2}\rho v^2)$ and γ = 1.4.

μ = 0.5 is used by the constrained RKDG method for all 1D test problems.

3.5.1 1D Woodward-Colella blast wave problem—The 1D Woodward-Colella blast wave problem [31] is the Euler equations with an initial data

$$\begin{aligned}(\rho,\rho v,E) &= (1,0,2500), \quad \text{for} 0<x<0.1,\\ (\rho,\rho v,E) &= (1,0,0.025), \quad \text{for} 0.1<x<0.9,\\ (\rho,\rho v,E) &= (1,0,250), \quad \text{for} 0.9<x<1.\end{aligned}$$

We computed the numerical solutions using 400 equal size cells. Density profiles of the solutions are plotted at the time 0.038 and are shown in Fig. 5. We use CFL number 0.1 for the RKDG schemes, 0.5 for third-order constrained DG and 0.8 for fourth-order constrained RKDG schemes, respectively. We can clearly see that the constrained RKDG solution and the RKDG solution have almost identical resolution for both third- and fourth-order accurate cases for the 1D Woodward-Colella blast wave problem.

3.5.2 1D Lax problem—The 1D Lax problem [12] is the Euler equations with the Lax's initial data

$$\begin{aligned}(\rho,\rho v,E) &= (0.445,0.311,8.928), \quad \text{for} -1<x<0,\\ (\rho,\rho v,E) &= (0.5,0,1.4275), \quad \text{for} 0\le x<1.\end{aligned}$$

We computed the numerical solutions using 200 equal size cells. The density profiles of the solutions are plotted at the time 0.26 and are shown in Fig. 6. We use CFL number 0.1 for the RKDG schemes, 0.5 for third-order constrained DG and 0.8 for fourth-order constrained RKDG schemes, respectively.

From these 1D compressible gas flow test problems, we notice that HR works well with the constrained RKDG method. And we conclude that the 1D constraint RKDG method combined with HR limiter, achieves good quality results for problems containing strong shock waves in the solutions.

3.6 Test case using 2D Euler equations with discontinuous solutions

In this sub-section, we test 2D gas dynamics problems with discontinuities in solutions to assess the non-oscillatory property of numerical solutions computed by the 2D constrained RKDG method together with HR limiter. μ = 0.5 is used by the constrained RKDG method for all 2D test problems.

3.6.1 Flow past a forward facing step—This flow problem is again taken from [31]. The setup of the problem is the following: a right-going Mach 3 uniform flow enters a wind tunnel of 1 unit wide and 3 units long. The step is 0.2 units high and is located 0.6 units from the left side of the tunnel. The problem is initialized by a uniform, right-going Mach 3 flow, which has density 1.4, pressure 1.0, and velocity 3.0. The initial state of the gas is also used as the in-flow boundary condition at the left side boundary. At the right side boundary, the out-flow boundary condition is applied there. Reflective boundary condition is applied along the walls of the tunnel.

The corner of the step is a singularity. Unlike in [31] and in other studies, we do not modify our schemes near the corner, which is known to lead to an erroneous entropy layer at the downstream bottom wall, as well as a spurious Mach stem at the bottom wall. Instead, we

use the approach introduced in [6], which is to locally refine the mesh near the corner, to decrease these artifacts. The edge length of the triangle away from the corner is roughly equal to $\frac{1}{160}$. Near the corner, the edge length of the triangle is roughly equal to $\frac{1}{320}$.

We use this test case to compare results computed by constrained RKDG schemes with different CFL numbers and the RKDG schemes, respectively. Figs. 7 and 8 plot the contours of the numerical solutions. We can see that the resolutions of the solutions computed by the constrained RKDG schemes are comparable with the ones of the solutions computed by the RKDG schemes. Additionally, the constrained RKDG schemes are able to take the CFL number 0.5; while the DG schemes use 0.1. We clearly see that the bigger CFL numbers used by the constrained RKDG schemes do not compromise the resolution of the solutions to this test problem.

3.6.2 2D Shu-Osher problem—The Shu-Osher problem [30] is a benchmark for testing the resolution that high-order accurate methods can provide. Here we set up this test problem using a 2D triangular mesh to assess the performance of the 2D constrained RKDG method.

Solutions to this test problem are computed in a rectangular domain of [−5, 5] × [0, 0.1] with a uniform triangulation of 301 vertices in the *x*-direction and 4 vertices in the *y*-direction. The initial value of the velocity component in the *y*-direction is zero. The reflecting boundary condition is used in the *y*-direction. The initial data is as follows

$$(\rho, v, p) = \begin{cases} (3.857143, 2.629369, 10.333333), & \text{for} x < -4, \\ (1 + 0.2\sin(5x), 0, 1), & \text{for} x \geq -4 \end{cases}$$

Density profiles of the solutions along a line parallel to x-axis are plotted in Fig. 9 at time T = 1.8, against a fine grid solution, which is treated as the ”exact” solution. We can see that the all third- and fourth-order numerical schemes capture the solution profile of the Shu-Osher problem nicely.

3.6.3 Double Mach reflection—The Double Mach reflection problem is taken from [31]. We solve the Euler equations in a rectangular computational domain of [0, 4] × [0, 1]. A reflecting wall lies at the bottom of the domain starting from $x = \frac{1}{6}$. Initially a right-moving Mach 10 shock is located at $x = \frac{1}{6}$, *y* = 0, making a 60° angle with the *x* axis and extends to the top of the computational domain at *y* = 1. The reflective boundary condition is used at the wall.

We test our method on unstructured meshes with the triangle edge length roughly equal to $\frac{1}{400}$. The density contour of the flow in the [0, 3] × [0, 1] region at the time T = 0.2 is shown with 30 equally spaced contour lines. Figs. 10 and 11 are the contour plots of the numerical solutions computed by the third- and fourth-order RKDG and constrained RKDG schemes

respectively. Figs. 12 and 13 show the “blown-up” portion around the double Mach region. We can see that both the RKDG and constrained RKDG schemes successfully reproduce the vortex sheet roll-up, and the constrained RKDG method does not compromise the resolution of the solution compared with the RKDG method.

4 Concluding Remarks

In this work, we have developed a conservation constrained RKDG method for solving conservation Laws. The new formulation requires the computed RKDG solution defined on a cell to satisfy additional conservation constraints in adjacent cells (in the least-square sense) and does not increase the complexity or change the compactness of the original RKDG method. This conservation constrained RKDG method improves the CFL number over the RKDG method by 3 times or more. Moreover, for the test problems with discontinuous solutions limited by HR, the constrained RKDG method also produces results similar to ones computed the RKDG method.

We also note that the HR limiter with partial neighboring cell technique is extended by introducing characteristic decomposition and local iteration. Even though this extension increases the computational cost slightly during the limiting process, HR limiting is applied locally so it essentially won’t hurt the overall complexity. In the future, we will explore the constrained DG formulation with TVD multi-step time-marching method and develop better HR limiting technique so that we will be able to solve shock wave problems numerically with better resolutions and less computational cost.

Acknowledgments

Simulations were performed on the Notre Dame Center for Research Computing (CRC) High Performance Computing (HPC) supercomputer (http://crc.nd.edu).

Research was supported in part by NSF grants DMS-1115887 and DMS-0800612 and NIH grants 1 R01 GM100470-01 and 1 R01 GM095959-01A1.

Research was supported in part by NSF grant DMS-1115671.

References

1. Abgrall R. A review of residual distribution schemes for hyperbolic and parabolic problems: the July 2010 state of the art. Commun. Comput. Phys. 2012; 11:1043–1080.
2. Abgrall R. On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation. J. Comput. Phys. 1994; 114:45–58.
3. Barth T, Frederickson P. High order solution of the Euler equations on unstructured grids using quadratic reconstruction. AIAA Paper. 1990:90–0013.
4. Biswas R, Devine KD, Flaherty JE. Parallel, adaptive finite element methods for conservation laws. Appl. Numer. Math. 1994; 14:255–283.
5. Christlieb A, Ong B, Qiu J-M. Integral Deferred Correction Methods constructed with High Order Runge-Kutta Integrators. Math. Comp. 2010; 79:761–783.
6. Cockburn B, Shu C-W. The TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws V: multidimensional systems. J. Comput. Phys. 1998; 141:199–224.

7. Cockburn B, Hou S, Shu C-W. The TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case. Math. Comp. 1990; 54:545–581.
8. 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. 1989; 52:411–435.
9. Cockburn B, Shu C-W. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework. Math. Comp. 1989; 52:411–435.
10. Cockburn B, Shu C-W. The Runge-Kutta local projection P1-discontinuous Galerkin finite element method for scalar conservation laws. ESAIM: Mathematical Modelling and Numerical Analysis. 1991; 25:337–361.
11. Cockburn B, Shu C-W. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. J. Sci. Comput. 2001; 16(3):173–261.
12. Lax P. Weak solutions of nonlinear hyperbolic equations and their numerical computations. Comm. Pure Appl. Math. 1954; 7:159.
13. van Leer B. Toward the ultimate conservative difference scheme: II. Monotonicity and conservation combined in a second order scheme. J. Comput. Phys. 1974; 14:361–370.
14. van Leer B. Towards the ultimate conservative difference scheme: IV. A new approach to numerical convection. J. Comput. Phys. 1977; 23:276–299.
15. van Leer B. Towards the ultimate conservative difference scheme: V. A second order sequel to Godunov's method. J. Comput. Phys. 1979; 32:101–136.
16. van Leer B, Nomura S. Discontinuous Galerkin for diffusion. AIAA-2005-5108. 2005
17. Liu Y-J, Shu C-W, Tadmor E, Zhang M-P. Central discontinuous Galerkin methods on overlapping cells with a non-oscillatory hierarchical reconstruction. SIAM J. Numer. Anal. 2007; 45:2442–2467.
18. Liu Y-J, Shu C-W, Tadmor E, Zhang M-P. Non-oscillatory hierarchical reconstruction for central and finite volume schemes. Comm. Comput. Phys. 2007; 2:933–963.
19. Liu Y-J, Shu C-W, Tadmor E, Zhang M-P. L^2 stability analysis of the central discontinuous Galerkin method and a comparison between the central and regular discontinuous Galerkin methods. ESAIM: Math. Mod. Numer. Anal. 2008; 42:593–607.
20. Liu Y-J, Shu C-W, Xu Z-L. Hierarchical Reconstruction with up to Second Degree Remainder for Solving Nonlinear Conservation Laws. Nonlinearity. 2009; 22:2799–2812.
21. Luo H, Baum JD, Lohner R. A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids. J. Comput. Phys. 2007; 225:686–713.
22. Luo H, Luo L-Q, Nourgaliev Robert. A reconstructed discontinuous Galerkin method for the Euler equations on arbitrary grids. Commun. Comput. Phys. 2012; 12:1495–1519.
23. Qiu J, Shu C-W. Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method. II: Two dimensional case. Comput. Fluids. 2005; 34:642–663.
24. Reed W, Hill T. Triangular mesh methods for the neutron transport equation. Tech. report la-ur-73-479, Los Alamos Scientific Laboratory. 1973
25. Roe PL. Approximate Riemann solvers, parameter vectors, and difference schemes. J. Comput. Phys. 1981; 43:357.
26. Shi J, Hu C, Shu C-W. A Technique of Treating Negative Weights in WENO Schemes. J. Comput. Phys. 2002; 175:108–127.
27. Shu C-W. TVB uniformly high-order schemes for conservation laws. Math. Comp. 1987; 49:105–121.
28. Shu, C-W. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In: Cockburn, B.; Johnson, C.; Shu, C-W.; Tadmor, E.; Quarteroni, A., editors. In Advanced Numerical Approximation of Nonlinear Hyperbolic Equations. Berlin: Springer; 1998. p. 1697*Lecture Notes in Mathematics*
29. Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock capturing schemes. J. Comput. Phys. 1988; 77:439–471.

30. Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock capturing schemes, II. J. Comput. Phys. 1989; 83:32–78.
31. Woodward P, Colella P. Numerical simulation of two-dimensional fluid flows with strong shocks. J. Comput. Phys. 1984; 54:115.
32. Warburton T, Hagstrom T. Taming the CFL number for discontinuous Galerkin methods on structured meshes. SIAM J. Numer. Anal. 2008; 46:3151–3180.
33. Xu Z-L, Liu Y-J, Shu C-W. Hierarchical reconstruction for discontinuous Galerkin methods on unstructured grids with a WENO type linear reconstruction and partial neighboring cells. J. Comput. Phys. 2009; 228:2194–2212.
34. Xu Z-L, Lin G. Spectral/hp element method with hierarchical reconstruction for solving nonlinear hyperbolic conservation laws. Acta Mathematica Scientia. 2009; 29(B):1737–1748.
35. Xu Z-L, Liu Y-J, Du H, Lin G, Shu C-W. Point-wise hierarchical reconstruction for discontinuous Galerkin and finite volume methods for solving conservation laws. J. Comput. Phys. 2011; 230:6843–6865.
36. Zhang M, Shu C-W. An analysis of and a comparison between the discontinuous Galerkin and the spectral finite volume methods. Comput. Fluids. 2005; 34:581–592.
37. Zhou T, Li Y-F, Shu C-W. Numerical Comparison of WENO Finite Volume and Runge-Kutta Discontinuous Galerkin Methods. J. Sci. Comput. 2001; 16(2):145–171.
38. Zhu J, Qiu J-X, Shu C-W, Dumbser M. Runge-Kutta discontinuous Galerkin II: unstructured meshes. J. Comput. Phys. 2008; 227:4330–4353.
39. Zhu J, Qiu J-X. Runge-Kutta discontinuous Galerkin method using WENO-type limiters: three-dimensional unstructured meshes. J. Comput. Phys. 2012; 11:985–1005.

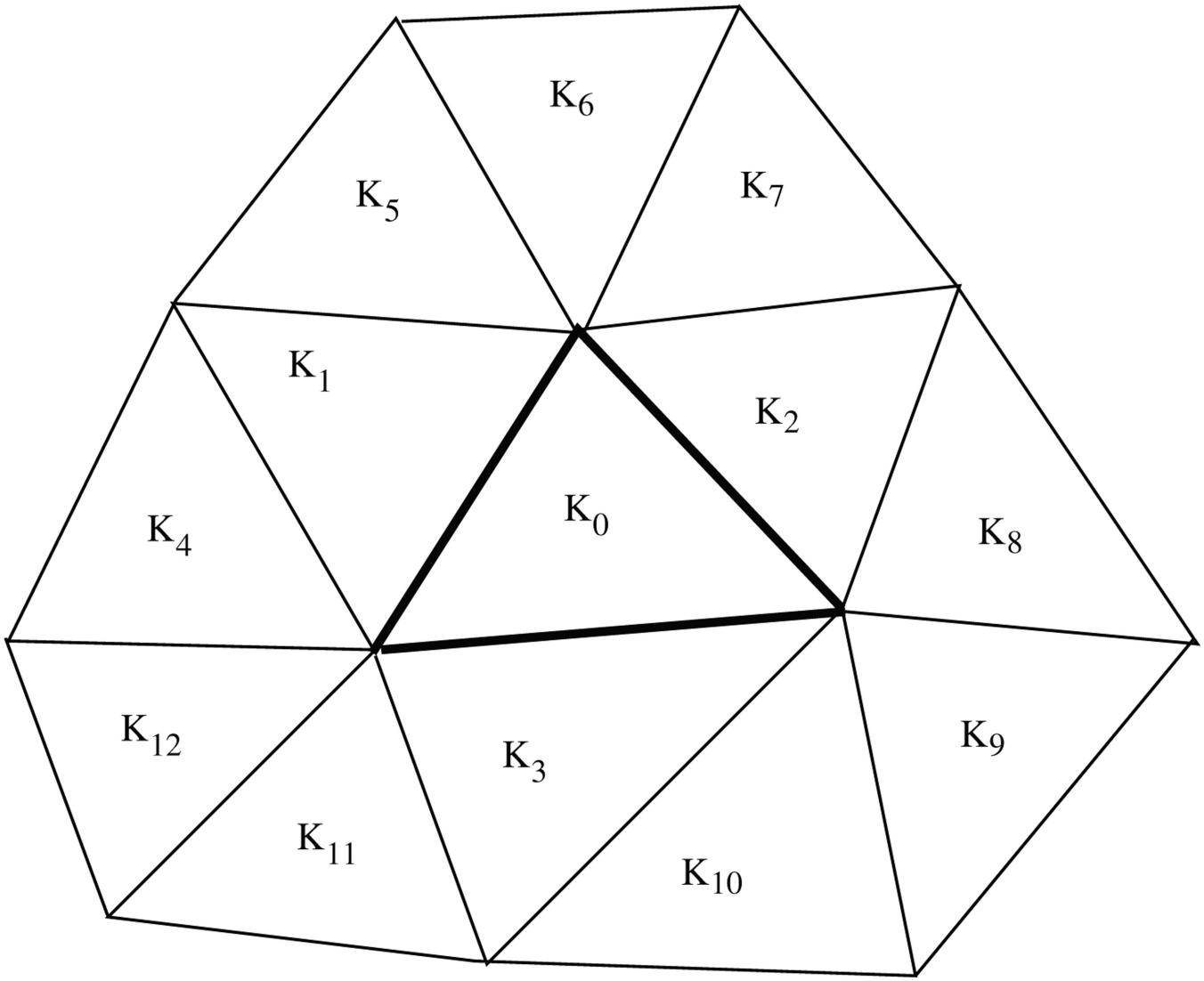

Figure 1.
Triangular cells that can be used as constraints for computing the solution supported on cell $\mathscr{K}_0$.

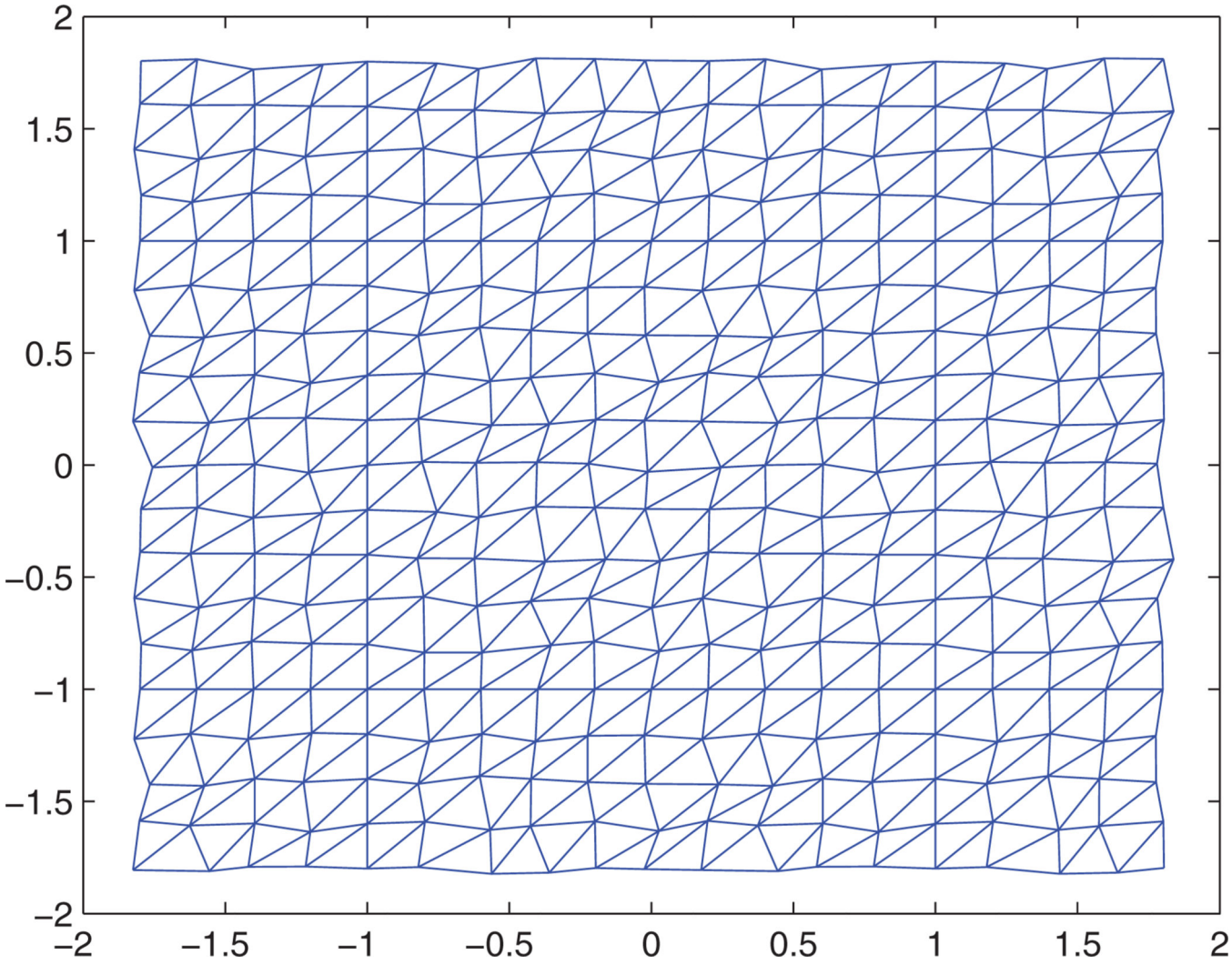

Figure 2.
Representative mesh for 2D accuracy tests.

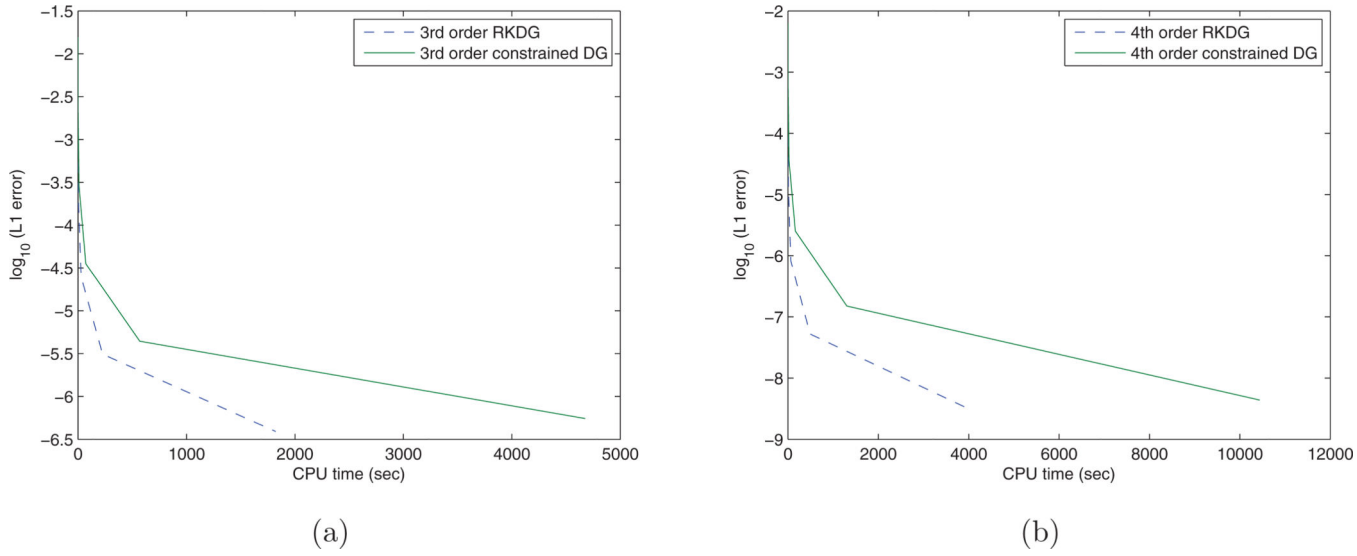

Figure 3.

L^1 errors versus CPU time for solving Eq. (3.7) up to time T = 0.5/π. (a) Comparison of the third-order RKDG and RKDG3-7Cell schemes. The CFL numbers are 0.22 and 0.8 respectively. (b) Comparison of the fourth-order RKDG and RKDG4-10Cell schemes. The CFL numbers are 0.25 and 0.9 respectively.

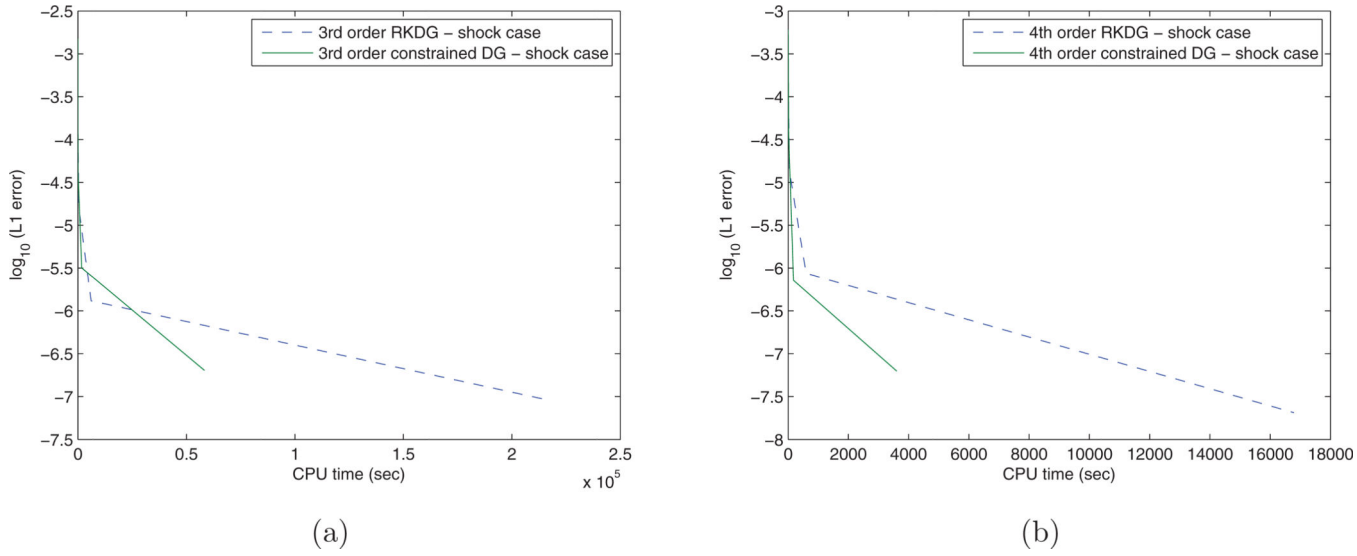

Figure 4.

L^1 errors versus CPU time for solving Eq. (3.7) up to time T = 0.45. The errors are computed in region [−1,−0.45] × [−1,−0.45]. (a) Comparison of the third-order RKDG and RKDG3-7Cell schemes. The CFL numbers are 0.22 and 0.8 respectively. (b) Comparison of the fourth-order RKDG and RKDG4-10Cell schemes. The CFL numbers are 0.25 and 0.9 respectively.

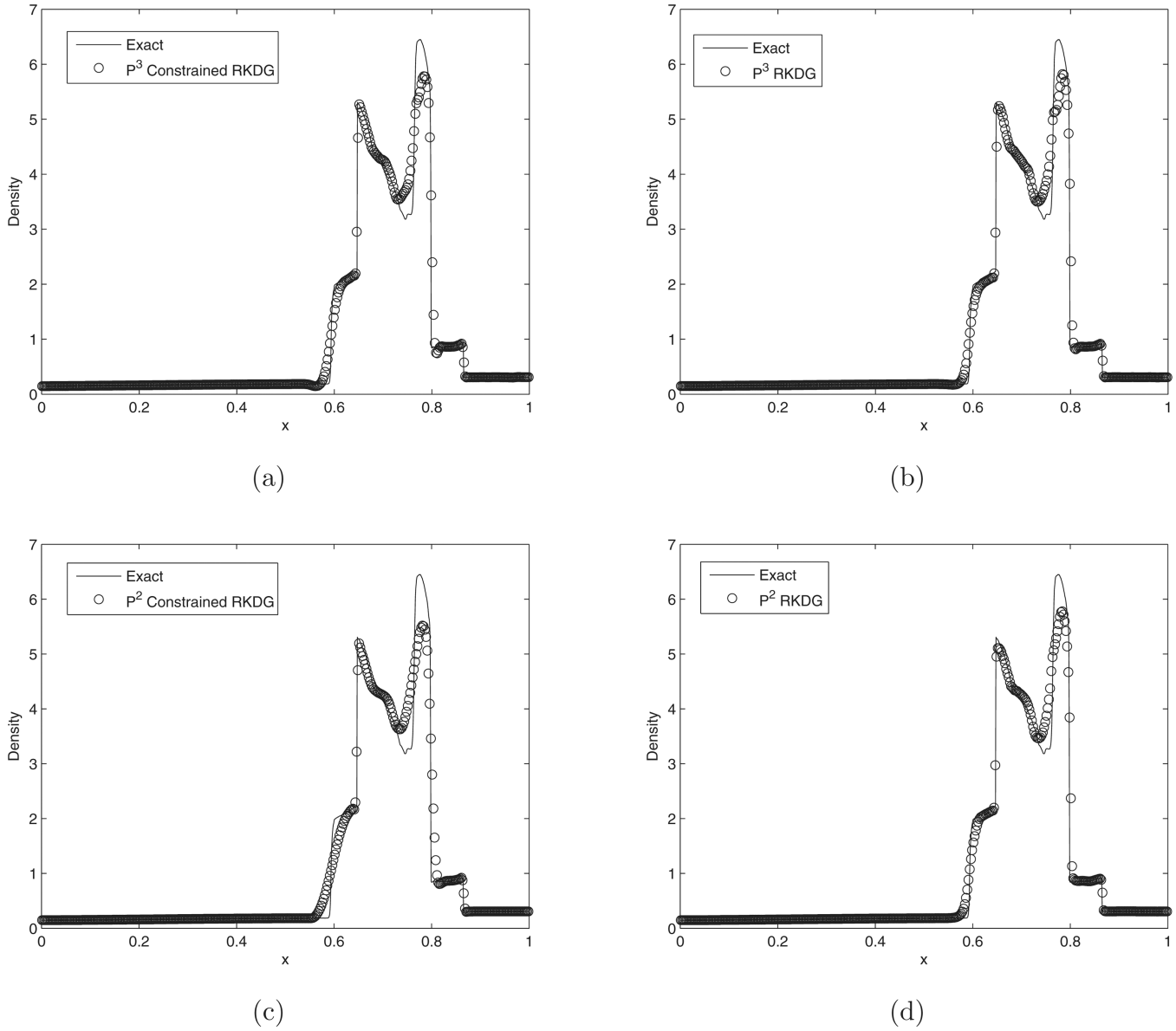

Figure 5.

Solutions of the 1D blast wave problem computed on 400 cells. (a) The 1D fourthorder accurate constrained RKDG4-3Cell solution compared with the “exact” solution, CFL = 0.5; (b) the 1D fourth-order accurate RKDG solution compared with the “exact” solution, CFL = 0.1; (c) the 1D third-order accurate constrained RKDG3-3Cell solution compared with the “exact” solution, CFL = 0.8; (d) the 1D third-order accurate RKDG solution compared with the “exact” solution, CFL = 0.1.

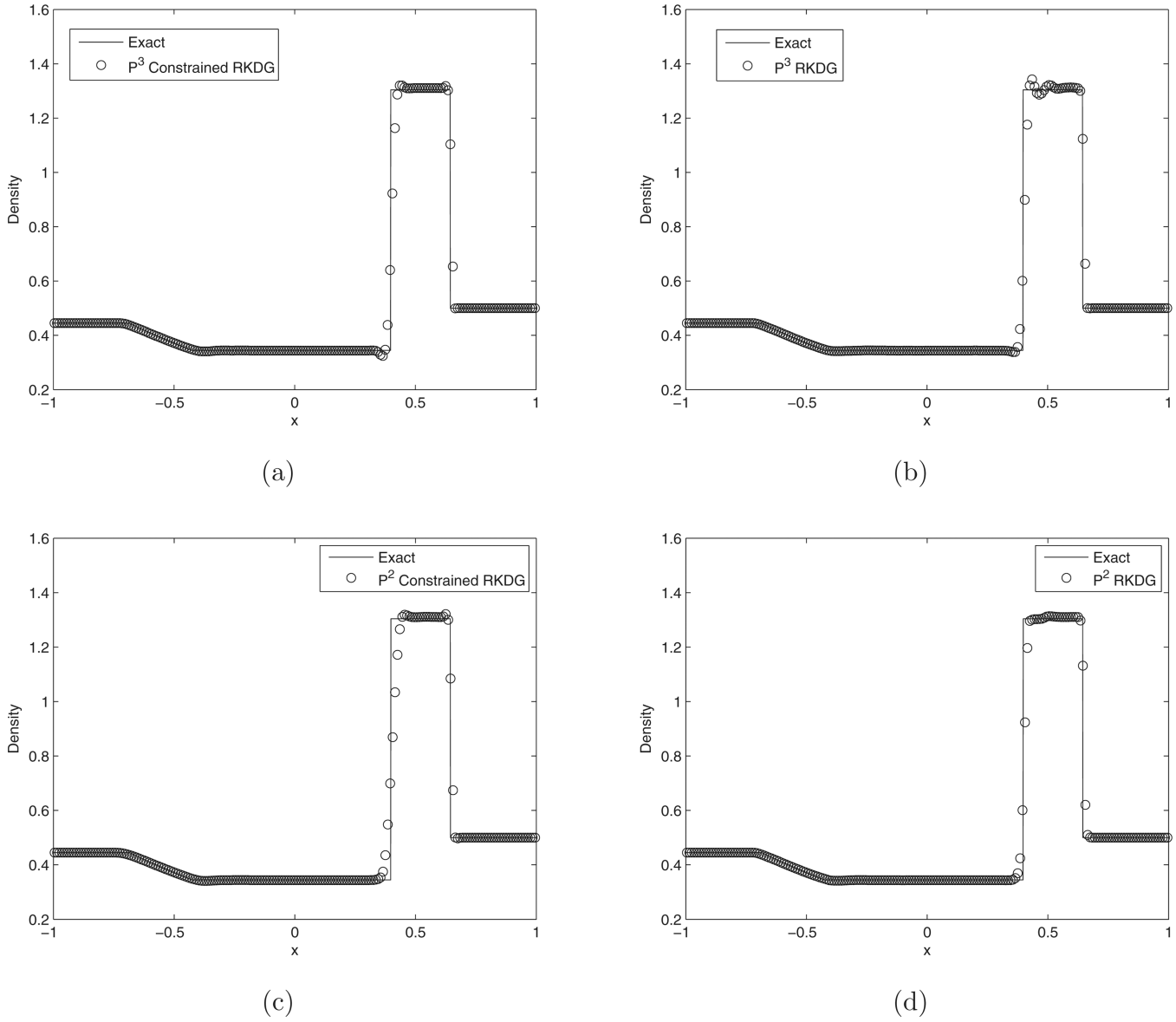

Figure 6.

Solutions of the 1D Lax shock tube problem computed on 200 cells. (a) The fourth-order accurate constrained RKDG4-3Cell solution compared with the “exact” solution, CFL = 0.5; (b) the fourth-order accurate RKDG solution compared with the “exact” solution, CFL = 0.1; (c) the third-order accurate constrained RKDG3-3Cell solution compared with the “exact” solution, CFL = 0.8; (b) the third-order accurate RKDG solution compared with the “exact” solution, CFL = 0.1.

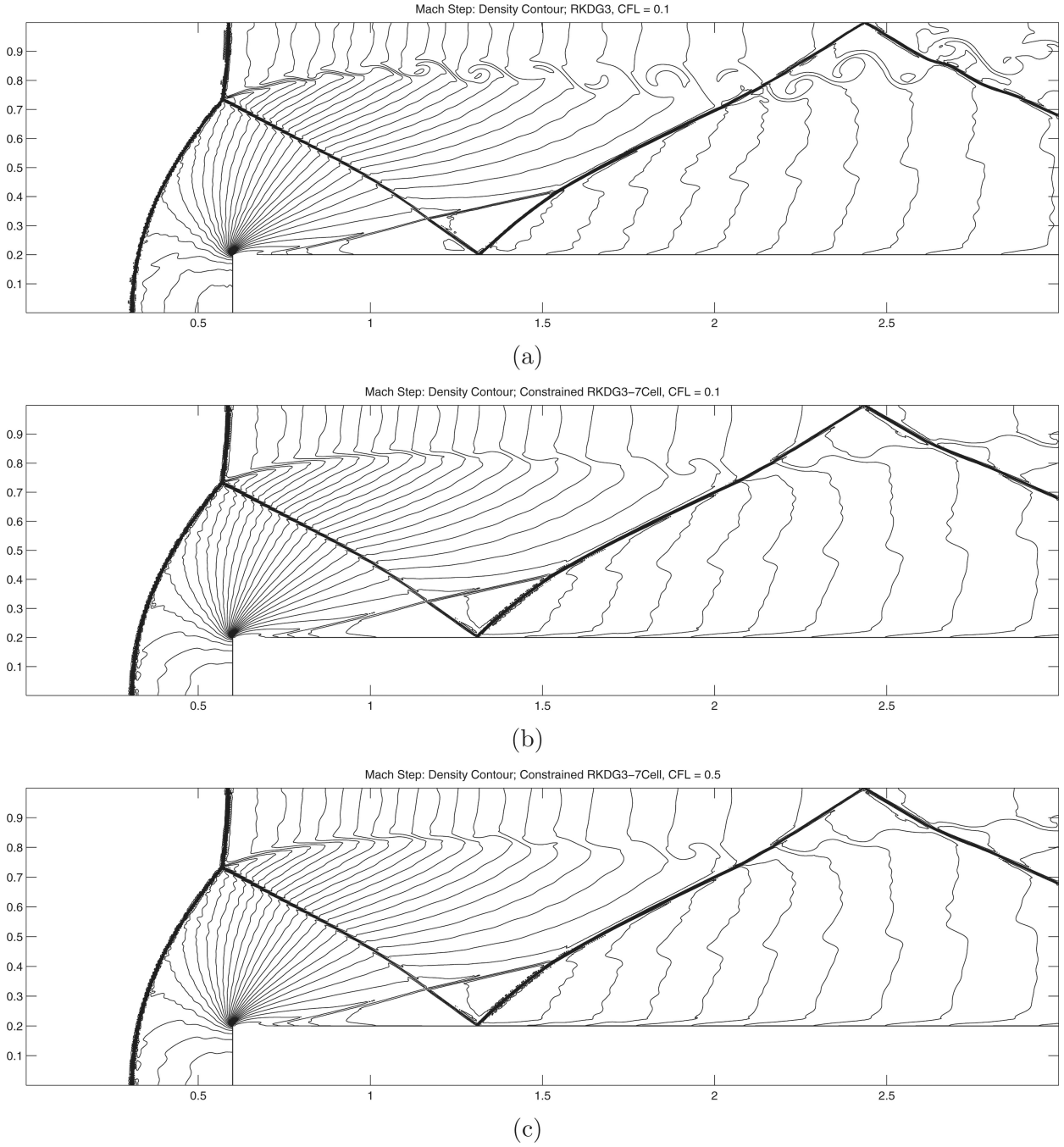

Figure 7.

Solutions to the forward-facing step problem by third-order accurate schemes. (a) The third-order accurate RKDG solution with CFL = 0.1; (b) The third-order accurate constrained RKDG3-7Cell solution with CFL = 0.1; (c) The third-order accurate constrained RKDG3-7Cell solution with CFL = 0.5.

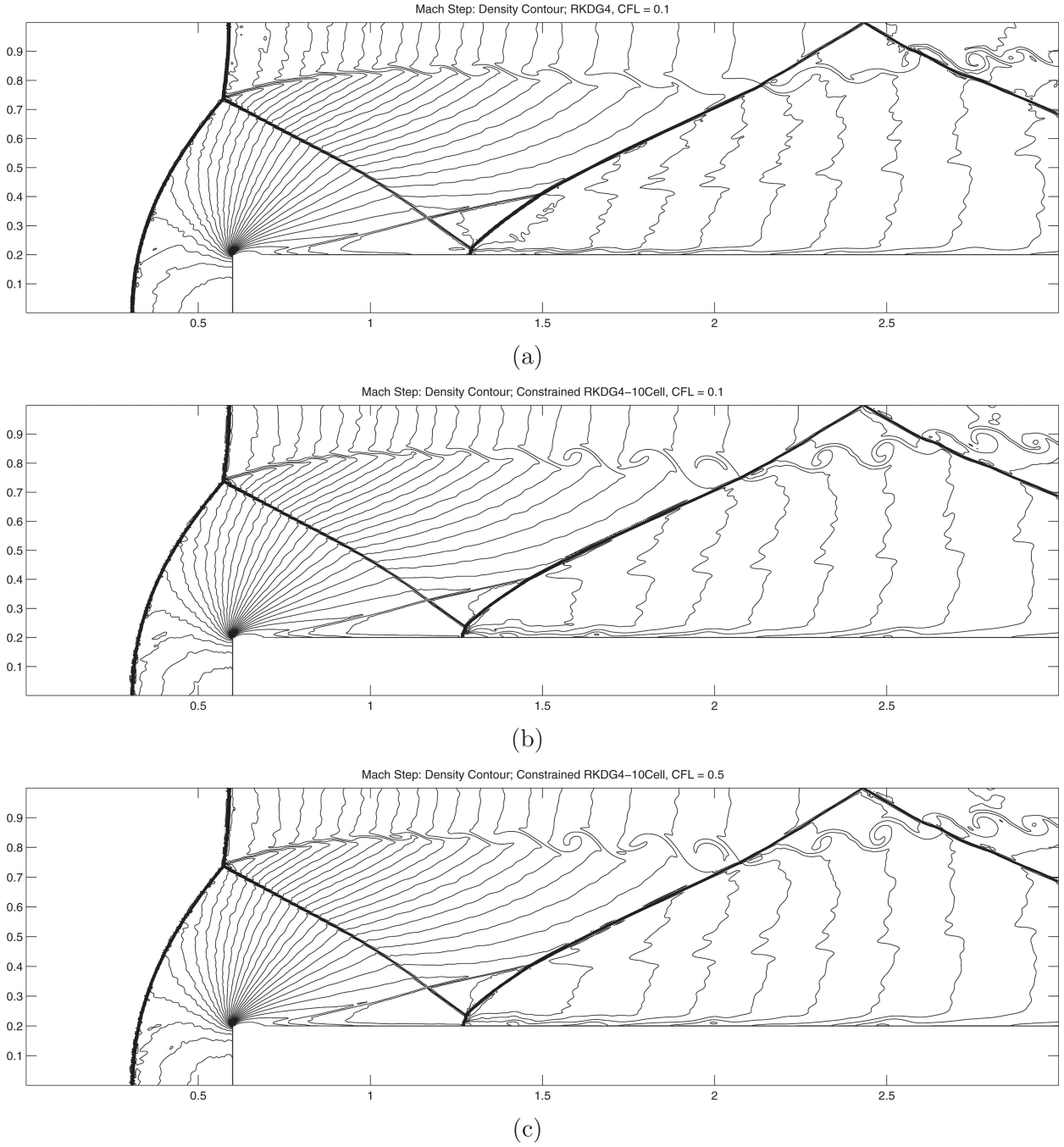

Figure 8.

Solutions to the forward-facing step problem computed by fourth-order accurate schemes. (a) The fourth-order accurate RKDG solution with CFL = 0.1; (b) The fourth-order accurate constrained RKDG4-10Cell solution with CFL = 0.1; (c) The fourth-order accurate constrained RKDG4-10Cell solution with CFL = 0.5.

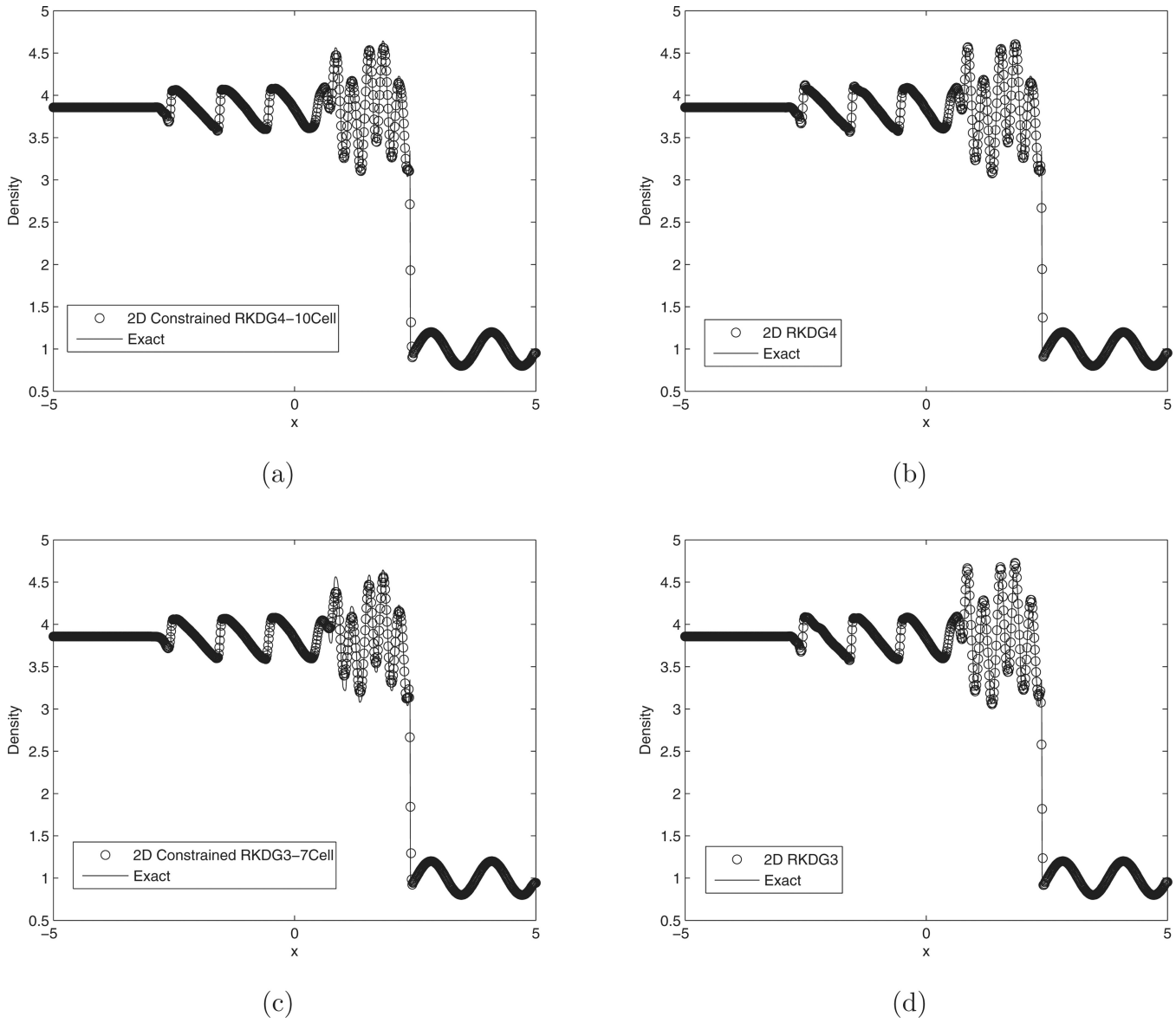

Figure 9.

Cross section plots of solutions of the 2D Shu-Osher problem computed on a rectangular domain of [−5, 5] × [0, 0.1]. (a) The 2D fourth-order accurate constrained RKDG4- 10Cell solution compared with the “exact” solution, CFL = 0.5; (b) the 2D fourth-order accurate RKDG solution compared with the “exact” solution, CFL = 0.1; (c) the 2D thirdorder accurate constrained RKDG3-7Cell solution compared with the “exact” solution, CFL = 0.5; (d) the 2D third-order accurate RKDG solution compared with the “exact” solution, CFL = 0.1.

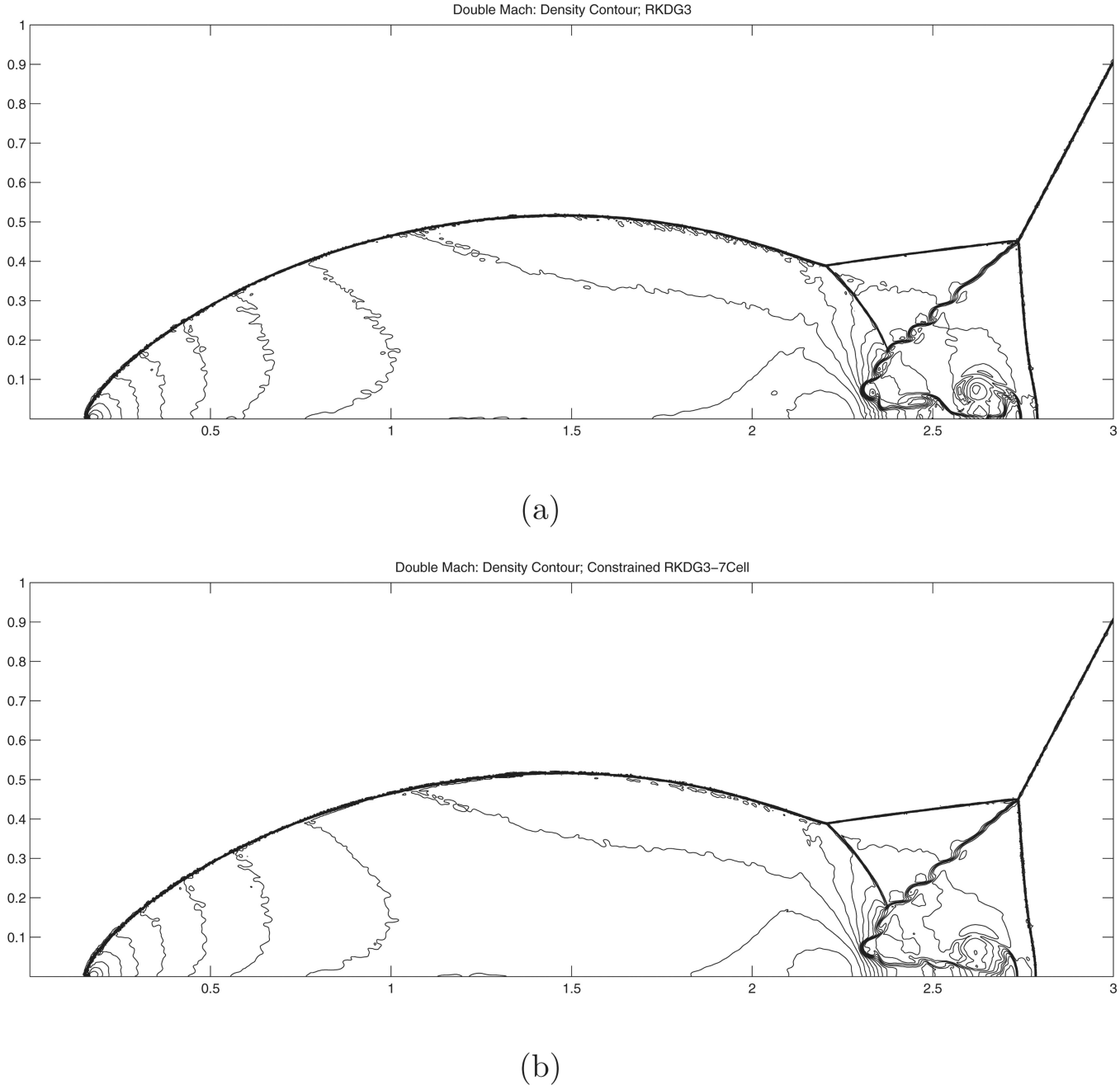

Figure 10.

Solutions to the double Mach reflection problem computed by third-order accurate schemes. T = 2.0. (a) The third-order accurate RKDG solution, CFL = 0.02; (b) The third-order accurate constrained RKDG3-7Cell solution, CFL = 0.1.

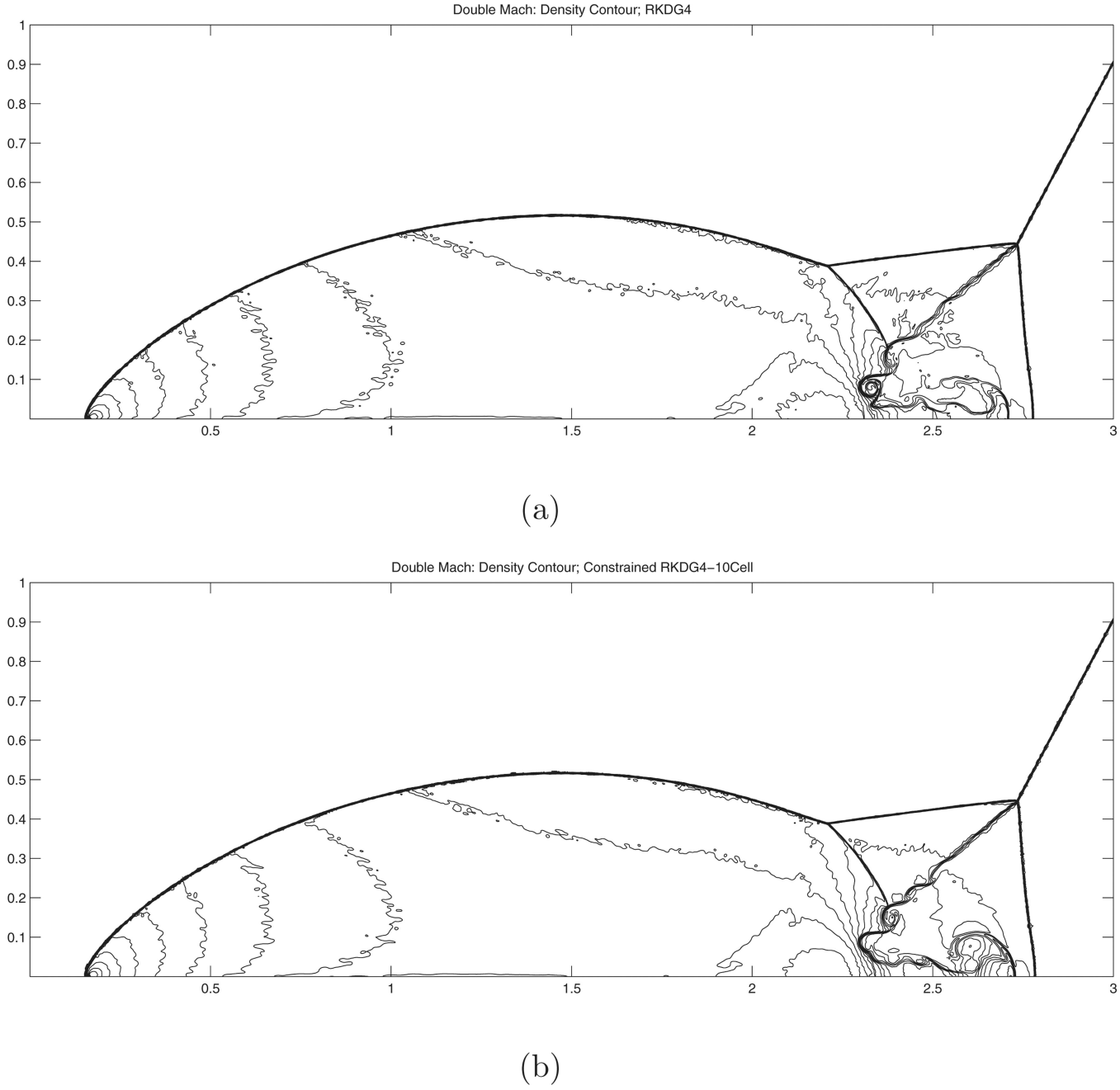

Figure 11.

Solutions to the double Mach reflection problem by fourth-order accurate schemes. T = 2.0. (a) The fourth-order accurate RKDG solution, CFL = 0.02; (b) The fourth-order accurate constrained RKDG4-10Cell solution, CFL = 0.1.

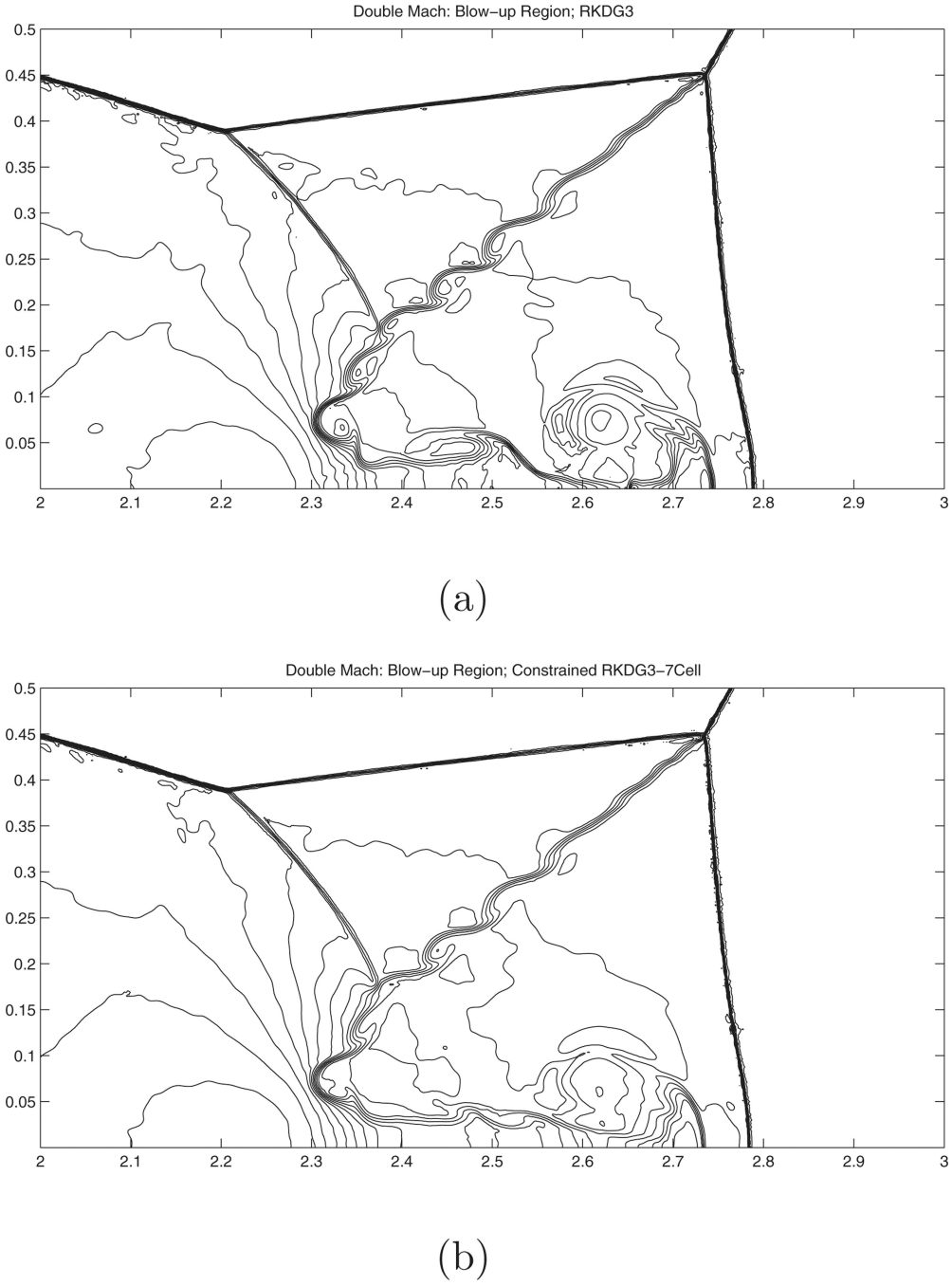

Figure 12.

Solutions to the double Mach reflection problem computed by third-order schemes. Blown-up region around the double Mach stems. (a) The third-order accurate RKDG solution; (b) The third-order accurate constrained RKDG3-7Cell solution.

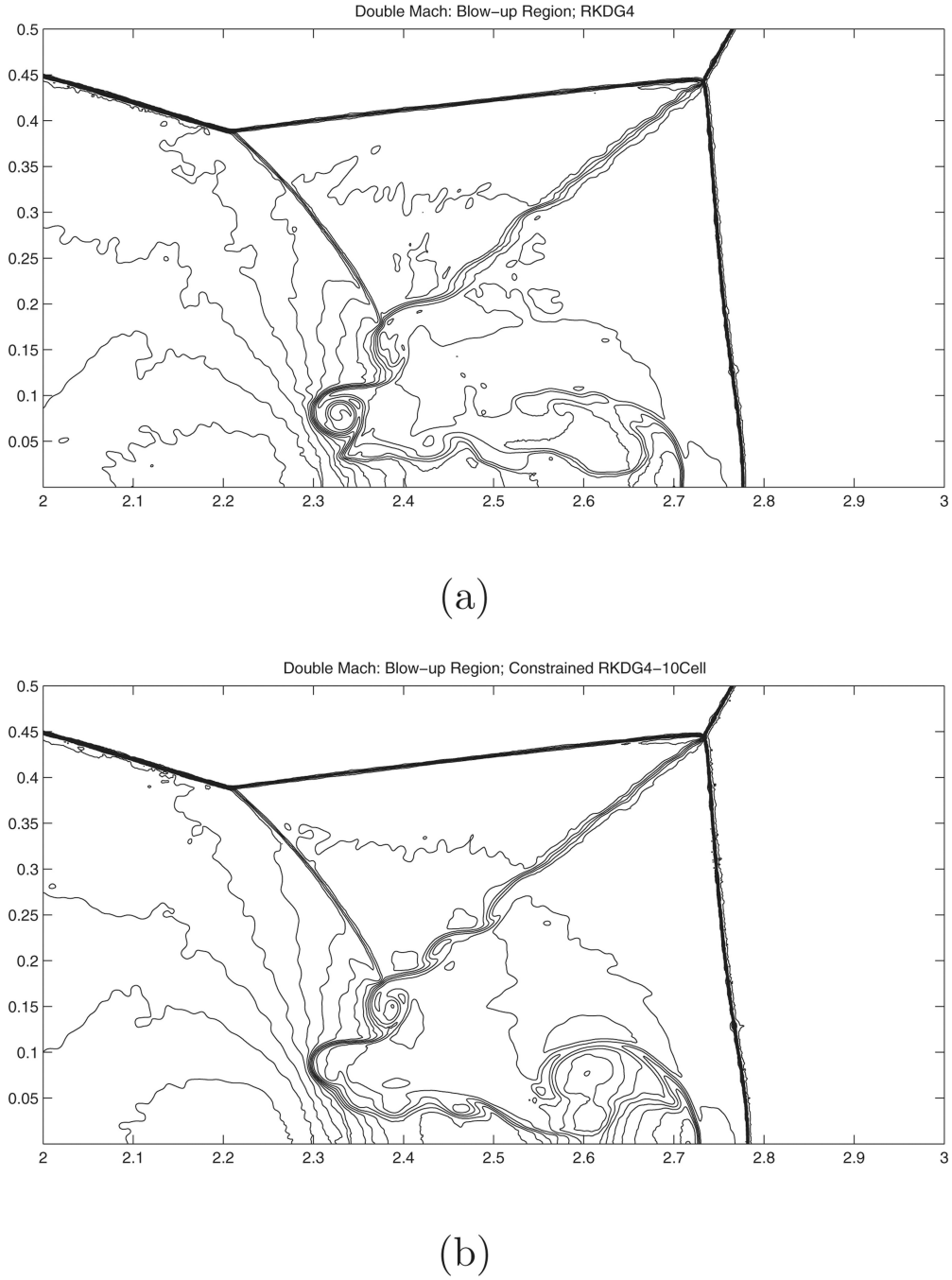

Figure 13.

Solutions to the double Mach reflection problem computed by fourth-order schemes. Blown-up region around the double Mach stems. (a) The fourth-order accurate RKDG solution; (b) The fourth-order accurate constrained RKDG4-10Cell solution.

Table 1

CFL numbers with $\mu = 0$ (no constraint).

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	0.33	0.11	0.05
3rd	0.40	0.20	0.13
4th	–	–	0.14

Table 2

CFL numbers with $\mu = 0.01$.

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	0.41	0.47	0.17
3rd	0.53	0.70	0.60
4th	–	–	0.47

Table 3

CFL numbers with $\mu = 0.5$.

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	0.95	0.83	0.18
3rd	1.2	1.6	0.49
4th	–	–	0.57

Table 4

CFL numbers with $\mu = 100$.

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	1.0	0.88	0.18
3rd	1.1	1.6	0.49
4th	–	–	0.56

Table 5

CFL numbers with $\mu = 500$.

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	1.0	0.88	0.18
3rd	1.1	1.6	0.49
4th	–	–	0.56

Table 6

CFL numbers with $\mu = 0.5$, without right constraint.

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	0.70	0.36	0.10
3rd	0.78	0.43	0.21

Table 7

CFL numbers with $\mu = 0.5$, without left constraint.

Temporal order	p-w linear	p-w quadratic	p-w cubic
2nd	0.45	0.36	0.088
3rd	1.43	0.45	0.22

Table 8

Accuracy test results of solving 1D linear advection equation (3.1) by using the third-order accurate schemes. L^1 and L^∞ errors. 3 cells are used for conservation penalty. T = 2.0.

Δx	1D Constrained RKDG3-3Cell, μ = 0.5, CFL = 1.6				1D RKDG3, CFL = 0.2.			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{400}$	4.74E-7	–	7.45E-7	–	1.48E-9	-	3.84E-9	–
$\frac{1}{800}$	5.92E-8	3.00	9.31E-8	3.00	1.85E-10	3.64	4.79E-10	3.00
$\frac{1}{1600}$	7.40E-9	3.00	1.16E-8	3.00	2.31E-11	3.00	6.00E-11	3.00
$\frac{1}{3200}$	9.25E-10	3.00	1.45E-9	3.00	3.29E-12	2.81	9.21E-12	2.70
$\frac{1}{6400}$	1.16E-10	3.00	1.82E-10	2.99	3.42E-13	3.27	1.08E-12	3.09
$\frac{1}{12800}$	1.45E-11	3.00	2.28E-11	3.00	–	–	–	–

Table 9

Accuracy test results of solving 1D linear advection equation (3.1) by using 1D Constrained RKDG3-3Cell scheme. L^1 and L^∞ errors. T = 2.0. CFL = 0.2.

Δx	1D Constrained RKDG3-3Cell, μ = 0.5			
	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{400}$	1.60E-7	–	2.53E-7	–
$\frac{1}{800}$	2.00E-8	3.00	3.16E-8	3.00
$\frac{1}{1600}$	2.50E-9	3.00	3.95E-9	3.00
$\frac{1}{3200}$	3.13E-10	3.00	4.95E-10	3.00
$\frac{1}{6400}$	3.91E-11	3.00	6.20E-11	3.00
$\frac{1}{12800}$	8.40E-12	2.20	1.54E-11	2.00

Table 10

Accuracy test results of solving 1D linear advection equation (3.1) by using the fourth-order accurate schemes. L^1 and L^∞ errors. 3 cells are used for conservation penalty. T = 2.0.

Δx	1D Constrained RKDG4-3Cell, μ = 0.5, CFL = 0.6				1D RKDG4, CFL = 0.1			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{50}$	1.34E-7	–	2.65E-7	–	2.06E-9	-	5.33E-9	–
$\frac{1}{100}$	8.30E-9	4.01	1.63E-8	4.02	1.28E-10	4.01	3.33E-10	4.00
$\frac{1}{200}$	5.23E-10	3.99	1.05E-9	3.96	8.02E-12	4.00	2.08E-11	4.00
$\frac{1}{400}$	3.25E-11	4.01	6.42E-11	4.03	7.52E-13	3.41	1.66E-12	3.65
$\frac{1}{800}$	2.13E-12	3.93	4.23E-12	3.92	–	–	–	–

Table 11

Accuracy test results of solving 1D linear advection equation (3.1) by using 1D Constrained RKDG4-3Cell scheme. L^1 and L^∞ errors. T = 2.0. CFL = 0.1.

Δx	1D Constrained RKDG4-3Cell, μ = 0.5			
	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{50}$	7.50E-8	–	1.99E-7	–
$\frac{1}{100}$	4.67E-9	4.00	1.26E-8	3.99
$\frac{1}{200}$	2.92E-10	4.00	7.95E-10	3.99
$\frac{1}{400}$	1.86E-11	3.97	5.01E-11	3.99
$\frac{1}{800}$	1.15E-12	4.03	3.07E-12	4.03

Table 12

Accuracy test results of solving 1D linear advection equation (3.1) by using the 1D third-order accurate constrained RKDG (1D Constrained RKDG3-3Cell) scheme with different μ values. L^1 and L^∞ errors. 3 cells are used for conservation penalty. CFL = 1.6. T = 2.0.

Δx	1D Constrained RKDG3-3Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 error	L^∞ error	L^1 error	L^∞ error
$\frac{1}{1600}$	7.40E-9	1.16E-8	7.67E-9	1.20E-8	7.69E-9	1.21E-8
$\frac{1}{3200}$	9.25E-10	1.45E-9	9.58E-10	1.51E-9	9.62E-10	1.51E-9
$\frac{1}{6400}$	1.16E-10	1.82E-10	1.20E-10	1.88E-10	1.20E-10	1.89E-10
$\frac{1}{12800}$	1.45E-11	2.28E-11	1.50E-11	2.36E-11	1.50E-11	2.38E-11

Table 13

Accuracy test results of solving 1D linear advection equation (3.1) by using the 1D fourth-order accurate constrained RKDG (1D Constrained RKDG4-3Cell) scheme with different μ values. L^1 and L^∞ errors. 3 cells are used for conservation penalty. CFL = 0.6. T = 2.0.

Δx	1D Constrained RKDG4-3Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 error	L^∞ error	L^1 error	L^∞ error
$\frac{1}{100}$	8.30E-9	1.63E-8	8.31E-9	1.63E-8	8.31E-9	1.63E-8
$\frac{1}{200}$	5.23E-10	1.05E-9	5.24E-10	1.05E-9	5.24E-10	1.05E-9
$\frac{1}{400}$	3.25E-11	6.42E-11	3.25E-11	6.42E-11	3.25E-11	6.60E-11
$\frac{1}{800}$	2.13E-12	4.23E-12	2.13E-12	4.24E-12	2.32E-12	9.66E-12

Table 14

Accuracy test results of solving 1D Burgers' equation (3.3) by using the third-order accurate schemes. L^1 and L^∞ errors. 3 cells are used for conservation penalty. T = 0.5/π.

Δx	1D Constrained RKDG3-3Cell, μ = 0.5, CFL = 1.6				1D RKDG3, CFL = 0.2			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{400}$	9.32E-8	–	6.17E-7	–	3.51E-9	–	5.78E-8	–
$\frac{1}{800}$	1.18E-8	2.98	8.11E-8	2.93	4.36E-10	3.01	7.30E-9	2.99
$\frac{1}{1600}$	1.47E-9	3.00	1.00E-8	3.02	5.44E-11	3.00	9.20E-10	2.99
$\frac{1}{3200}$	1.85E-10	2.99	1.29E-9	2.95	6.84E-12	2.99	1.17E-10	2.98
$\frac{1}{6400}$	2.30E-11	3.01	1.60E-10	3.01	1.11E-12	2.62	1.81E-11	2.69
$\frac{1}{12800}$	3.00E-12	2.94	2.01E-11	2.99	–	–	–	–

Table 15

Accuracy test results of solving 1D Burgers' equation (3.3) by using the fourth-order accurate schemes. L^1 and L^∞ errors. 3 cells are used for conservation penalty. T = 0.5/π.

Δx	1D Constrained RKDG4-3Cell, μ = 0.5, CFL = 0.6				1D RKDG4, CFL = 0.1			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{50}$	4.56E-7	–	1.18E-5	–	2.27E-8	–	3.64E-7	–
$\frac{1}{100}$	2.99E-8	3.93	7.67E-7	3.94	1.42E-9	4.00	2.27E-8	4.00
$\frac{1}{200}$	1.93E-9	3.95	4.90E-8	3.97	8.89E-11	4.00	1.44E-9	3.98
$\frac{1}{400}$	1.21E-10	4.00	3.08E-9	3.99	5.81E-12	3.94	9.27E-11	3.96
$\frac{1}{800}$	7.87E-12	3.94	1.95E-10	3.98	–	–	–	–

Table 16

Accuracy test results of solving 1D Burgers' equation (3.3) by using the third-order accurate constrained RKDG (1D Constrained RKDG3-3Cell) scheme with different μ values. L^1 and L^∞ errors. 3 cells are used for conservation penalty. CFL = 1.6. T = 0.5/π.

Δx	1D Constrained RKDG3-3Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 error	L^∞ error	L^1 error	L^∞ error
$\frac{1}{1600}$	1.47E-9	1.00E-8	1.49E-9	1.05E-9	1.49E-9	1.06E-8
$\frac{1}{3200}$	1.85E-10	1.29E-9	1.86E-10	1.33E-10	1.87E-10	1.34E-9
$\frac{1}{6400}$	2.30E-11	1.60E-10	2.34E-11	1.70E-11	2.34E-11	1.71E-10
$\frac{1}{12800}$	3.00E-12	2.01E-11	3.04E-12	2.13E-12	3.05E-12	2.15E-11

Table 17

Accuracy test results of solving 1D Burgers' equation (3.3) by using the fourth-order accurate constrained RKDG (1D Constrained RKDG4-3Cell) scheme with different μ values. L^1 and L^∞ errors. 3 cells are used for conservation penalty. CFL = 0.6. T = 0.5/π.

Δx	1D Constrained RKDG4-3Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 error	L^∞ error	L^1 error	L^∞ error
$\frac{1}{200}$	1.93E-9	4.90E-8	1.94E-9	4.91E-08	1.94E-9	4.91E-08
$\frac{1}{400}$	1.21E-10	3.08E-9	1.21E-10	3.08E-09	1.21E-10	3.08E-09
$\frac{1}{800}$	7.87E-12	1.95E-10	7.87E-12	1.95E-10	8.03E-12	1.97E-10

Table 18

Accuracy test results of solving 2D linear advection equation (3.5).by the third-order accurate schemes. L^1 and L^∞ errors. 7 cells are used for conservation penalty. T = 2.0.

h	2D Constrained RKDG3-7Cell, μ = 0.5, CFL = 0.8				RKDG3, CFL = 0.22			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{320}$	2.69E-6	–	1.06E-6	–	1.83E-8	–	1.73E-8	–
$\frac{1}{640}$	3.36E-7	3.00	1.32E-7	3.01	2.29E-9	3.00	2.22E-9	2.96
$\frac{1}{1280}$	4.20E-8	3.00	1.66E-8	2.99	2.88E-10	2.99	2.82E-10	2.98
$\frac{1}{2560}$	5.26E-9	3.00	2.07E-9	3.00	4.01E-11	2.84	4.82E-11	2.55

Table 19

Accuracy test results of solving 2D linear advection equation (3.5) by the fourth-order accurate schemes. L^1 and L^∞ errors. 10 cells are used for conservation penalty. T = 2.0.

h	2D Constrained RKDG4-10 Cell, μ = 0.5, CFL=0.9				RKDG4, CFL = 0.2			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{80}$	2.40E-7	–	2.05E-7	–	5.43E-9	–	4.80E-9	–
$\frac{1}{160}$	1.54E-8	3.96	1.51E-8	3.76	3.35E-10	4.02	3.34E-10	3.85
$\frac{1}{320}$	9.55E-10	4.01	1.06E-9	3.83	2.13E-11	3.98	2.09E-11	4.00
$\frac{1}{640}$	5.92E-11	4.01	7.54E-11	3.81	2.72E-12	2.97	3.57E-12	2.55

Table 20

Accuracy test results of solving 2D linear advection equation (3.5) by the third-order accurate constrained RKDG (2D Constrained RKDG3-7 Cell) scheme with different μ values. L^1 and L^∞ errors. 7 cells are used for conservation penalty. CFL = 0.8. T = 2.0.

h	2D Constrained RKDG3-7Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 error	L^∞ error	L^1 error	L^∞ error
$\frac{1}{320}$	2.69E-6	1.06E-6	2.72E-6	1.07E-06	2.73E-6	1.07E-6
$\frac{1}{640}$	3.36E-7	1.32E-7	3.41E-7	1.34E-7	3.41E-7	1.34E-7
$\frac{1}{1280}$	4.20E-8	1.66E-8	4.26E-8	1.68E-8	4.27E-8	1.68E-8
$\frac{1}{2560}$	5.26E-9	2.07E-9	5.33E-9	2.10E-9	5.33E-9	2.10E-9

Table 21

Accuracy test results of solving 2D linear advection equation (3.5) by the fourth-order accurate constrained RKDG (2D Constrained RKDG4-10 Cell) scheme with different μ values. L^1 and L^∞ errors. 10 cells are used for conservation penalty. CFL = 0.9. T = 2.0.

h	2D Constrained RKDG4-10 Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 order	L^∞ order	L^1 order	L^∞ error
$\frac{1}{160}$	1.54E-8	1.51E-8	3.31E-8	2.53E-08	3.85E-8	2.90E-8
$\frac{1}{320}$	9.55E-10	1.06E-9	2.08E-9	1.81E-9	2.42E-9	2.02E-9
$\frac{1}{640}$	5.92E-11	7.54E-11	1.35E-10	1.35E-10	1.57E-10	1.50E-10

Table 22

Accuracy test results of solving 2D linear advection equation (3.5) by the fourth-order accurate constrained RKDG (2D Constrained RKDG4) scheme with different numbers of cells used for conservation penalty. L^1 and L^∞ errors. μ = 0.5. T = 2.0.

h	2D Constrained RKDG4-13 Cell, CFL = 1.3				2D Constrained RKDG4-7 Cell, CFL = 0.35			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{80}$	3.27E-7	–	2.62E-7	–	3.09E-7	–	1.51E-7	–
$\frac{1}{160}$	1.93E-8	4.09	1.79E-8	3.87	1.92E-8	4.01	1.04E-8	3.86
$\frac{1}{320}$	1.29E-9	3.90	1.38E-9	3.70	1.20E-9	4.00	6.94E-10	3.91
$\frac{1}{640}$	8.02E-11	4.01	8.76E-11	3.98	7.62E-11	3.98	4.56E-11	3.93

Table 23

Accuracy test results of solving 2D Burgers' equation (3.7) by the third-order accurate schemes. L^1 and L^∞ errors. 7 cells are used for conservation penalty. T = 0.5/π.

h	2D Constrained RKDG3-7Cell, μ = 0.5, CFL=0.8				2D RKDG3, CFL = 0.22			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{80}$	3.51E-5	-	6.87E-5	-	3.06E-6	-	1.26E-5	-
$\frac{1}{160}$	4.40E-6	3.00	9.09E-6	2.92	3.85E-7	2.99	1.67E-6	2.92
$\frac{1}{320}$	5.50E-7	3.00	1.30E-6	2.88	4.85E-8	2.99	2.27E-7	2.88
$\frac{1}{640}$	6.93E-8	2.99	1.78E-7	2.87	6.08E-9	3.00	3.03E-8	2.91
$\frac{1}{1280}$	8.62E-9	3.01	2.43E-8	2.87	7.62E-10	3.00	4.04E-9	2.91
$\frac{1}{2560}$	1.08E-9	3.00	3.14E-9	2.95	9.58E-11	2.99	5.49E-10	2.88

Table 24

Accuracy test results of solving 2D Burgers' equation (3.7) by the third-order accurate Constrained RKDG (2D RKDG3-7 Cell) scheme with different μ values. L^1 and L^∞ errors. 7 cells are used for conservation penalty. CFL = 0.8. T = 0.5/π.

h	2D Constrained RKDG3-7Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 order	L^∞ order	L^1 order	L^∞ error
$\frac{1}{160}$	4.40E-6	9.09E-6	4.44E-6	9.37E-6	4.44E-6	9.37E-6
$\frac{1}{320}$	5.50E-7	1.30E-6	5.55E-7	1.31E-6	5.56E-7	1.31E-6
$\frac{1}{640}$	6.93E-8	1.78E-7	6.95E-8	1.78E-7	6.95E-8	1.78E-7
$\frac{1}{1280}$	8.62E-9	2.43E-8	8.69E-9	2.43E-8	8.70E-9	2.43E-8
$\frac{1}{2560}$	1.08E-9	3.14E-9	1.09E-9	3.14E-9	1.09E-9	3.14E-9

Table 25

Accuracy test results of solving 2D Burgers' equation (3.7) by the third-order accurate constrained RKDG (2D Constrained RKDG3) scheme with different numbers of cells used for conservation penalty. L^1 and L^∞ errors. μ = 0.5. $T = 0.5/\pi$.

h	2D Constrained RKDG3-7Cell, CFL=0.8				2D Constrained RKDG3-4Cell, CFL = 0.3			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{160}$	4.40E-6	–	9.09E-6	–	1.12E-6	–	3.46E-6	–
$\frac{1}{320}$	5.50E-7	3.00	1.30E-6	2.88	1.42E-7	2.98	4.89E-7	2.82
$\frac{1}{640}$	6.93E-8	2.99	1.73E-7	2.87	1.80E-8	2.98	6.50E-8	2.91
$\frac{1}{1280}$	8.62E-9	3.01	2.43E-8	2.87	2.27E-9	2.99	9.23E-9	2.82
$\frac{1}{2560}$	1.08E-9	3.00	3.14E-9	2.95	2.85E-10	2.99	1.35E-9	2.77

Table 26

Accuracy test results of solving 2D Burgers' equation (3.7) by the fourth-order accurate schemes. L^1 and L^∞ errors. 10 cells are used for conservation penalty. T = 0.5/π.

h	2D Constrained RKDG4-10 Cell, μ = 0.5, CFL=0.9				2D RKDG4, CFL = 0.25			
	L^1 error	L^1 order	L^∞ error	L^∞ order	L^1 error	L^1 order	L^∞ error	L^∞ order
$\frac{1}{80}$	2.46E-6	–	9.42E-6	–	5.18E-8	–	1.88E-7	–
$\frac{1}{160}$	1.51E-7	4.03	6.29E-7	3.90	3.29E-9	3.98	1.25E-8	3.91
$\frac{1}{320}$	1.00E-8	3.92	4.25E-8	3.89	2.10E-10	3.97	9.51E-10	3.72
$\frac{1}{640}$	6.01E-10	4.06	3.10E-9	3.78	1.35E-11	3.96	6.61E-11	3.85
$\frac{1}{1280}$	3.83E-11	3.97	2.28E-10	3.77	–	–	–	–

Table 27

Accuracy test results of solving 2D Burgers' equation (3.7) by the fourth-order accurate Constrained RKDG (2D RKDG4-10 Cell) scheme with different μ values. L^1 and L^∞ errors. 10 cells are used for conservation penalty. CFL = 0.9. T = 0.5/π.

h	2D Constrained RKDG4-10 Cell					
	μ = 0.5		μ = 5		μ = 500	
	L^1 error	L^∞ error	L^1 order	L^∞ order	L^1 order	L^∞ error
$\frac{1}{160}$	1.51E-7	6.29E-7	1.63E-7	7.42E-7	1.64E-7	7.67E-7
$\frac{1}{320}$	1.00E-8	4.25E-8	1.03E-8	5.31E-8	1.04E-8	5.48E-8
$\frac{1}{640}$	6.01E-10	3.10E-9	6.49E-10	3.64E-9	6.56E-10	3.85E-9
$\frac{1}{1280}$	3.83E-11	2.28E-10	4.11E-11	2.56E-10	4.16E-11	2.71E-10

Table 28

Accuracy test results of solving 2D Burgers' equation (3.7) by the fourth-order accurate constrained RKDG (2D Constrained RKDG4) scheme with different numbers of cells used for conservation penalty. L^1 and L^∞ errors. μ = 0.5. T = 0.5/π.

h	2D Constrained RKDG4-13 Cell, CFL = 1.4				2D Constrained RKDG4-7 Cell, CFL = 0.35			
	L^1 error	L^1 error	L^∞ error	L^∞ error	L^1 order	L^1 order	L^∞ order	L^∞ error
$\frac{1}{160}$	1.55E-7	–	6.38E-7	–	5.22E-8	–	2.56E-7	–
$\frac{1}{320}$	9.36E-9	4.05	5.19E-8	3.62	3.41E-9	3.94	1.73E-8	3.89
$\frac{1}{640}$	5.89E-10	3.99	3.29E-9	3.98	2.21E-10	3.95	1.18E-9	3.87
$\frac{1}{1280}$	3.73E-11	3.98	2.39E-10	3.78	1.44E-11	3.94	8.41E-11	3.81