Journal of Scientific Computing, Vol. 30, No. 3, March 2007 (© 2006)
DOI: 10.1007/s10915-006-9109-5

A Numerical Comparison of the Lax–Wendroff Discontinuous Galerkin Method Based on Different Numerical Fluxes

Jianxian Qiu[1]

Received January 9, 2006; accepted (in revised form) Feburary 23, 2006; Published online September 19, 2006

Discontinuous Galerkin (DG) method is a spatial discretization procedure, employing useful features from high-resolution finite volume schemes, such as the exact or approximate Riemann solvers serving as numerical fluxes and limiters. In [(2005). Comput. Methods Appl. Mech. Eng. **194**, 4528], we developed a Lax–Wendroff time discretization procedure for the DG method (LWDG), an alternative method for time discretization to the popular total variation diminishing (TVD) Runge–Kutta time discretizations. In most of the DG papers in the literature, the Lax–Friedrichs numerical flux is used due to its simplicity, although there are many other numerical fluxes, which could also be used. In this paper, we systematically investigate the performance of the LWDG method based on different numerical fluxes, including the first-order monotone fluxes such as the Godunov flux, the Engquist–Osher flux, etc., the second-order TVD fluxes and generalized Riemann solver, with the objective of obtaining better performance by choosing suitable numerical fluxes. The detailed numerical study is mainly performed for the one-dimensional system case, addressing the issues of CPU cost, accuracy, non-oscillatory property, and resolution of discontinuities. Numerical tests are also performed for two-dimensional systems.

KEY WORDS: Discontinuous Galerkin method; Lax–Wendroff type time discretization; numerical flux; approximate Riemann solver; limiter; WENO scheme; high-order accuracy.

AMS (MOS) SUBJECT CLASSIFICATION: 65M60; 65M99; 35L65.

[1] Department of Mathematics, Nanjing University, Nanjing, Jiangsu, P.R. China 210093. E-mail: jxqiu@nju.edu.cn

1. INTRODUCTION

In this paper, we investigate the performance of the Lax–Wendroff discontinuous Galerkin (LWDG) method [16] based on different numerical fluxes for solving nonlinear hyperbolic conservation laws

$$\begin{aligned} &u_t + \nabla \cdot f(u) = 0, \\ &u(x, 0) = u_0(x) \end{aligned} \tag{1.1}$$

with the objective of obtaining better performance by choosing suitable numerical fluxes.

The Runge–Kutta discontinuous Galerkin (RKDG) method [2–6], for solving hyperbolic conservation laws is a high-order finite element method employing the useful features from high-resolution finite volume schemes, such as the exact or approximate Riemann solvers, TVD Runge–Kutta time discretizations [21, 23], and total variation bounded (TVB) limiters [22].

In RKDG method, DG is a spatial discretization procedure, namely, it is a procedure to approximate the spatial derivative terms in (1.1). The time derivative term there is discretized by explicit, nonlinearly stable high-order Runge–Kutta time discretizations [21, 23]. An alternative approach could be using a Lax–Wendroff type time discretization procedure, which is also called the Taylor type referring to a Taylor expansion in time or the Cauchy–Kowalewski type referring to the similar Cauchy–Kowalewski procedure in partial differential equations (PDEs) [24]. This approach is based on the idea of the classical Lax–Wendroff scheme [14], and it relies on converting all the time derivatives in a temporal Taylor expansion into spatial derivatives by repeatedly using the PDE and its differentiated versions. The spatial derivatives are then discretized by, e.g., the DG approximations. The methods are termed as LWDG methods [16]. Lax–Wendroff type time discretization procedure is also used by Dumbser and Munz [7], in which they developed the ADER (Arbitrary high-order schemes using DERivatives, see [25]) discontinuous Galerkin method using generalized Riemann solvers [25]. The ADER methods also use the Lax–Wendroff procedure to convert time derivatives to spatial derivatives. The Lax–Wendroff type time discretization was also used in high-order finite volume schemes [10, 25] and finite difference schemes [18].

As pointed out in [16], the LWDG is a one step, explicit, high-order finite element method, the limiter is performed once every time step. As a result, LWDG is more compact than RKDG and the Lax–Wendroff time discretization procedure is more cost effective than the Runge–Kutta time discretizations for certain problems including

two-dimensional Euler systems of compressible gas dynamics when nonlinear limiters are applied.

An important component of the DG methods for solving conservation laws (1.1) is the numerical flux, based on exact or approximate Riemann solvers, which is borrowed from finite difference and finite volume methodologies. In most of the DG papers in the literature, the two point, first-order monotone Lax–Friedrichs (LF) numerical flux is used due to its simplicity. However, there exist many other numerical fluxes based on various approximate Riemann solvers in the literature, such as other two point, first-order monotone fluxes and essentially two point TVD flux, which could also be used in the context of DG methods. The Godunov flux [9], the Engquist–Osher (EO) flux [8, 15] for the scalar case and its extension to systems (often referred to as the Osher–Solomon flux [15]), the HLL flux [11] and a modification of the HLL flux, often referred to as the HLLC flux [27] are based on the approximate Riemann solver, these fluxes are two point, first-order monotone fluxes. One of the essentially two point TVD fluxes is the flux limiter centered (FLIC) flux [26] with the following *essentially two point* property: $\hat{f}(u^l, u, u, u^r) = f(u)$ for any u^l and u^r, which combines a low-order monotone flux and a high-order flux with a flux limiter to guarantee the TVD property. The other fluxes such as generalized Riemann solvers [1, 25] can also be used as numerical flux for DG methods.

In [17], we have systematically studied and compared the performance of the RKDG method based on different numerical fluxes, with the objective of obtaining better performance by choosing suitable numerical fluxes. In [16], the simple Lax–Friedrichs flux is used to design the LWDG schemes. Although, the schemes work well for both one- and two-dimensional cases in [16], there could be room for improvement by using different fluxes along the line of [17]. In this paper, based on [16, 17], we study and compare the performance of the LWDG method based on different numerical fluxes, with the objective of obtaining better performance by choosing suitable numerical fluxes. We review and describe the details of the fluxes under consideration in Sect. 2, and present extensive numerical experiments in Sect. 3 to compare their performance. Concluding remarks are given in Sect. 4.

2. REVIEW AND IMPLEMENTATION OF THE NUMERICAL FLUXES FOR THE LWDG METHODS

In this section, we review the numerical fluxes under consideration for the LWDG methods. We start with the description of the LWDG method for one and two-dimensional scalar and system conservation laws.

2.1. One Dimensional Case

Consider the one-dimensional scalar conservation laws:

$$u_t + f(u)_x = 0, \quad u(x,0) = u_0(x). \tag{2.1}$$

We denote the cells by $I_i = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$, the cell centers by $x_i = \frac{1}{2}\left(x_{i-\frac{1}{2}} + x_{i+\frac{1}{2}}\right)$ and the cell sizes by $\Delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$. Let Δt be the time step, $t^{n+1} = t^n + \Delta t$. By a temporal Taylor expansion we obtain

$$u(x, t+\Delta t) = u(x,t) + \Delta t u_t + \frac{\Delta t^2}{2} u_{tt} + \frac{\Delta t^3}{6} u_{ttt} + \cdots. \tag{2.2}$$

If we would like to obtain (k+1)th order accuracy in time, we would need to approximate the first $k+1$ time derivatives: $u_t, \ldots, \frac{\partial^{(k+1)} u}{\partial t^{k+1}}$. We will proceed up to third-order in time in this paper, although the procedure can be naturally extended to any higher orders.

The temporal derivative terms in (2.2) can be replaced with the spatial ones using the governing equation (2.1):

$$\begin{aligned} u_t &= -f(u)_x = -f'(u)u_x, \\ u_{tt} &= -(f'(u)u_t)_x = -f''(u)u_x u_t - f'(u)u_{xt}, \\ u_{xt} &= -f''(u)(u_x)^2 - f'(u)u_{xx}, \\ u_{ttt} &= -(f''(u)(u_t)^2 + f'(u)u_{tt})_x. \end{aligned}$$

Then, we can rewrite the approximation to (2.2) up to third-order as:

$$u(x, t+\Delta t) = u(x,t) - \Delta t F_x \tag{2.3}$$

with

$$F = f + f^*, \quad f^* = \tfrac{\Delta t}{2} f'(u)u_t + \tfrac{\Delta t^2}{6}(f''(u)(u_t)^2 + f'(u)u_{tt}). \tag{2.4}$$

The standard discontinuous Galerkin method is then used to discretize F_x in (2.3), described in detail below.

The DG solution as well as the test function space is given by $V_h^k = \{p : p|_{I_i} \in P^k(I_i)\}$, where $P^k(I_i)$ is the space of polynomials of degree $\leqslant k$ on the cell I_i. We adopt a local orthogonal basis over I_i, $\{v_l^{(i)}(x), l = 0, 1, \ldots, k\}$, namely the scaled Legendre polynomials

$$v_0^{(i)}(x) = 1, \qquad v_1^{(i)}(x) = \frac{x - x_i}{\Delta x_i}, \qquad v_2^{(i)}(x) = \left(\frac{x - x_i}{\Delta x_i}\right)^2 - \frac{1}{12}, \ldots.$$

Other basis functions can be used as well, without changing the numerical method, since the finite element DG method depends only on the choice of space V_h^k, not on the choice of its basis functions.

The numerical solution $u^h(x,t)$ in the space V_h^k can be written as:

$$u^h(x,t) = \sum_{l=0}^{k} u_i^{(l)}(t) v_l^{(i)}(x) \quad \text{for } x \in I_i \tag{2.5}$$

and the degrees of freedom $u_i^{(l)}(t)$ are the moments defined by

$$u_i^{(l)}(t) = \frac{1}{a_l} \int_{I_i} u^h(x,t) v_l^{(i)}(x) dx, \quad l = 0, 1, \ldots, k,$$

where $a_l = \int_{I_i} (v_l^{(i)}(x))^2 dx$ are the normalization constants since the basis is not orthonormal. In order to determine the approximate solution, we evolve the degrees of freedom $u_i^{(l)}$:

$$u_i^{(l)}(t^{n+1}) = u_i^{(l)}(t^n) + \frac{1}{a_l} \left(- \int_{I_i} F \frac{d}{dx} v_l^{(i)}(x) dx + \hat{F}_{i+1/2} v_l^{(i)}(x_{i+1/2}) - \hat{F}_{i-1/2} v_l^{(i)}(x_{i-1/2}) \right) = 0, \qquad l = 0, 1, \ldots, k, \tag{2.6}$$

where $\hat{F}_{i+1/2}$ is a numerical flux which depends on the values of the numerical solution u^h and its spatial derivatives at the cell interface $x_{i+1/2}$, both from the left and from the right. This numerical flux is related to the so-called generalized Riemann solvers [25]. In [16] we use the following simple Lax–Friedrichs flux

$$\begin{aligned} \hat{F}_{i+1/2} &= \frac{1}{2} \left(F_{i+1/2}^- + F_{i+1/2}^+ - \alpha \left(u_{i+1/2}^+ - u_{i+1/2}^- \right) \right), \\ &= \frac{1}{2} \left(f_{i+1/2}^- + f_{i+1/2}^+ - \alpha \left(u_{i+1/2}^+ - u_{i+1/2}^- \right) \right) \\ &\quad + \frac{1}{2} \left(f_{i+1/2}^{*-} + f_{i+1/2}^{*+} \right), \\ &= \hat{f}_{i+1/2} + \frac{1}{2} \left(f_{i+1/2}^{*-} + f_{i+1/2}^{*+} \right), \end{aligned} \tag{2.7}$$

where $u_{i+1/2}^{\pm}$, $F_{i+1/2}^{\pm}$, $f_{i+1/2}^{\pm}$, and $f_{i+1/2}^{*\pm}$ are the left and right limits of the discontinuous solution u^h and F, f, and f^* at the cell interface $x_{i+1/2}$, respectively, and $\alpha = \max_u |f'(u)|$. For the system case, the maximum is taken for the eigenvalues of the Jacobian $f'(u)$. The other numerical fluxes will be presented in detail in Sect. 2.3. The integral term in (2.6) can be

computed either exactly or by a suitable numerical quadrature accurate to at least $O(\Delta x^{k+l+2})$. In this paper, we use two and three point Gaussian quadratures for $k=1$ and $k=2$, respectively.

For systems of conservation laws (2.1), $u(x,t)=(u^1(x,t),\ldots,u^m(x,t))^T$ is a vector and $f(u)=(f^1(u^1,\ldots,u^m),\ldots,f^m(u^1,\ldots,u^m))^T$ is a vector function of u. As before, the time derivatives in (2.2) are replaced by the spatial derivatives using the PDE. The DG discretization is then performed on each component.

2.2. Two-Dimensional Cases

Consider the two-dimensional conservation laws:

$$u_t + f(u)_x + g(u)_y = 0, \quad u(x,y,0)=u_0(x,y). \tag{2.8}$$

By a temporal Taylor expansion we obtain

$$u(x,y,t+\Delta t)=u(x,y,t)+\Delta t u_t+\frac{\Delta t^2}{2}u_{tt}+\frac{\Delta t^3}{6}u_{ttt}+\cdots.$$

For example, for third-order accuracy in time we would need to reconstruct three time derivatives: u_t, u_{tt}, u_{ttt}.

We again use the PDE (2.8) to replace time derivatives by spatial derivatives.

$$\begin{aligned}
u_t &= -f(u)_x - g(u)_y = -f'(u)u_x - g'(u)u_y,\\
u_{tt} &= -(f'(u)u_t)_x - (g'(u)u_t)_y = -(f''(u)u_x u_t\\
&\quad + f'(u)u_{xt} + g''(u)u_y u_t + g'(u)u_{yt}),\\
u_{xt} &= -(f''(u)(u_x)^2 + f'(u)u_{xx} + g''(u)u_x u_y + g'(u)u_{xy}),\\
u_{yt} &= -(f''(u)u_y u_x + f'(u)u_{xy} + g''(u)(u_y)^2 + g'(u)u_{yy}),\\
u_{ttt} &= -(f''(u)(u_t)^2 + f'(u)u_{tt})_x - (g''(u)(u_t)^2 + g'(u)u_{tt})_y.
\end{aligned}$$

Then we rewrite the approximation to (2.8) up to third-order as:

$$u(x,t+\Delta t)=u(x,t)-\Delta t(F_x+G_y) \tag{2.9}$$

with

$$\begin{aligned}
F &= f+\frac{\Delta t}{2}f'(u)u_t+\frac{\Delta t^2}{6}(f''(u)(u_t)^2+f'(u)u_{tt}),\\
G &= g+\frac{\Delta t}{2}g'(u)u_t+\frac{\Delta t^2}{6}(g''(u)(u_t)^2+g'(u)u_{tt}).
\end{aligned}$$

The standard discontinuous Galerkin method is then used to discretize F_x and G_y in (2.9).

For systems of conservation laws (2.8), the time derivatives are replaced by the spatial derivatives using the PDE. The DG discretization is then performed on each component.

In order to maintain stability and non-oscillatory property of the DG method for solving conservation laws (1.1) with strong shocks, a nonlinear limiter must be applied. In the numerical experiments in this paper, we will use the shock detection technique by Krivodonova *et al.* [13] to detect troubled-cells (we refer to [20] for a detailed investigation of various troubled-cell indicators), where a WENO limiter developed in [19] will be used for the reconstruction of first and higher order moments of the polynomials inside those troubled cells. We refer to [19] for the details of this WENO reconstruction and will not repeat it here. For the case of hyperbolic systems, to identify the troubled cells, we could either use a component-wise indicator or a characteristic one. In this paper, we will use the component-wise indicator. For both the one and the two-dimensional Euler equations, we use only the components of density and energy as indicator variables. We emphasize that the component-wise strategy is used only to identity troubled cells. Once these cells are identified, the WENO reconstructions in them are performed in local characteristic directions. We again refer to [19] for more details of the reconstruction.

2.3. Description of Fluxes for LWDG

We now review the two point or essentially two point numerical fluxes under consideration. Numerical experiments to compare their performance for the LWDG method will be given in next section.

For the one-dimensional system case, we will consider Euler equations of compressible gas dynamics, namely (2.1) with

$$u = (\rho, \rho v, E)^T, \quad f(u) = (\rho v, \rho v^2 + p, v(E + p))^T, \tag{2.10}$$

where ρ is the density, v the velocity, E the total energy, p the pressure, which is related to the total energy by $E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho v^2$ with $\gamma = 1.4$ for air. We will also use the sound speed $c = \sqrt{\gamma p/\rho}$ in the definition of some of the numerical fluxes as follows:

1. *The LF flux and the local LF (LLF) flux.*

The LF flux is one of the simplest and most widely used building blocks for the DG method and high-order finite volume methods such as the ENO and WENO schemes. However, the numerical viscosity of the LF

flux is also the largest among monotone fluxes for scalar problems. The LF or the LLF flux is defined by (2.7), where for the LF flux, α is taken as an upper bound over the whole line for $|f'(u)|$ in the scalar case, or for the absolute value of eigenvalues of the Jacobian for the system case, and for the LLF flux α is taken as an upper bound between u^- and u^+.

2. *The Godunov flux.*

The Godunov flux [9, 26] is based on the *exact* Riemann solver, which has the smallest numerical viscosity among all monotone fluxes for the scalar case but could be very costly to evaluate in the system case, as it often lacks explicit formulas and relies on iterative procedures for its evaluation. The Godunov flux is defined as

$$\hat{f}^G(u^-, u^+) = f(u(0)),$$

where $u(0)$ is the solution of the local Riemann problem at $x/t = 0$ (the solution of the local Riemann problem is a function of the single variable x/t only due to self-similarity), i.e. the exact solution to the conservation law (2.1) with the initial condition:

$$u(x, 0) = \begin{cases} u^- & \text{for } x \leqslant 0, \\ u^+ & \text{for } x > 0. \end{cases}$$

For the scalar case, the Godunov flux can be expressed in a closed form as

$$\hat{f}^G(u^-, u^+) = \begin{cases} \min_{u^- \leqslant u \leqslant u^+} f(u), & \text{if } u^- \leqslant u^+, \\ \max_{u^+ \leqslant u \leqslant u^-} f(u), & \text{if } u^- > u^+. \end{cases} \tag{2.11}$$

However, for most nonlinear systems, the Godunov flux cannot be expressed in a closed form. Its evaluation would in general require an iterative procedure. We refer to [26] and the references therein for the details of the exact Riemann solver for systems in applications, such as the Euler equations (2.10), which are needed for the evaluation of the Godunov flux for such systems. Then the final Godunov flux for LWDG is:

$$\hat{F}^G(u^-, u^+) = \hat{f}^G(u^-, u^+) + \frac{1}{2}(f^*(u^-) + f^*(u^+)). \tag{2.12}$$

3. *The EO flux and the Osher–Solomon flux* [8, 15].

The EO flux [8] for the scalar case and its extension to systems (often referred to as the Osher–Solomon flux [15]) are smoother than the Godunov flux with an almost as small numerical viscosity, and have the advantage of explicit formulas for the scalar case and for some well

known physical systems, such as the Euler equations of compressible gas dynamics.

For the scalar case the EO flux is given by:

$$\hat{f}^{EO}(u^-, u^+) = \frac{1}{2}\left(f(u^-) + f(u^+) - \int_{u^-}^{u^+} |f'(u)| du \right), \tag{2.13}$$

For the system case, the explicit formulas for the Osher–Solomon flux for the Euler equations (2.10) refers to [15, 17], we do not repeat it here to save space. Then the final EO flux for LWDG is:

$$\hat{F}^{EO}(u^-, u^+) = \hat{f}^{EO}(u^-, u^+) + \frac{1}{2}(f^*(u^-) + f^*(u^+)). \tag{2.14}$$

4. *The Harten–Lax–van Leer (HLL) flux* [11, 26].

The HLL flux [11] is based on the approximate Riemann solver with only three constant states separated by two waves. The HLL flux for the Euler equations (2.10) is given by:

$$\hat{f}^{HLL}(u^-, u^+) = \begin{cases} f(u^-), & \text{if } 0 \leqslant s^-, \\ \frac{s^+ f(u^-) - s^- f(u^+) + s^- s^+ (u^+ - u^-)}{s^+ - s^-}, & \text{if } s^- \leqslant 0 \leqslant s^+, \\ f(u^+), & \text{if } s^+ \leqslant 0. \end{cases} \tag{2.15}$$

where the lower and upper bounds of the wave speed, s^- and s^+, must be estimated. We use the pressure-velocity estimates [26]

$$s^- = v^- - c^- q^-, \quad s^* = v^*, \quad s^+ = v^+ + c^+ q^+, \tag{2.16}$$

where, for $K = \pm$,

$$q^K = \begin{cases} 1, & \text{if } p^* \leqslant p^K, \\ (1 + \frac{\gamma+1}{2\gamma}(p^*/p^K - 1))^{1/2}, & \text{if } p^* > p^K \end{cases}$$

with

$$p^* = \frac{1}{2}(p^- + p^+) - \frac{1}{2}(v^+ - v^-)\overline{\rho}\,\overline{c}, \qquad v^* = \frac{1}{2}(v^- + v^+) - \frac{p^+ - p^-}{2\overline{\rho}\,\overline{c}},$$

and

$$\overline{\rho} = \frac{1}{2}(\rho^- + \rho^+), \qquad \overline{c} = \frac{1}{2}(c^- + c^+).$$

Then the final HLL flux for LWDG is:

$$\hat{F}^{\mathrm{HLL}}(u^-, u^+) = \hat{f}^{\mathrm{HLL}}(u^-, u^+) + \frac{1}{2}(f^*(u^-) + f^*(u^+)). \tag{2.17}$$

5. *The HLLC flux—a modification of the HLL flux* [27, 26].

The HLLC flux is a modification of the HLL flux, whereby the missing contact and shear waves are restored. The HLLC flux for the Euler equations (2.10) is given by:

$$\hat{f}^{\mathrm{HLLC}}(u^-, u^+) = \begin{cases} f(u^-), & \text{if } 0 \leqslant s^-, \\ f(u^-) + s^-(u^{*-} - u^-), & \text{if } s^- \leq 0 \leqslant s^*, \\ f(u^+) + s^+(u^{*+} - u^+), & \text{if } s^* \leq 0 \leqslant s^+, \\ f(u^+), & \text{if } s^+ \leqslant 0, \end{cases} \tag{2.18}$$

where, for $K = \pm$,

$$u^{*K} = \rho^K \frac{s^K - v^K}{s^K - s^*} \begin{bmatrix} 1 \\ s^* \\ \frac{E^K}{\rho^K} + (s^* - v^K)\left[s^* + \frac{p^K}{\rho^K (s^K - v^K)}\right] \end{bmatrix} \tag{2.19}$$

The definitions of s^-, s^*, and s^+ are given in (2.16).

Then the final HLLC flux for LWDG is:

$$\hat{F}^{\mathrm{HLLC}}(u^-, u^+) = \hat{f}^{\mathrm{HLLC}}(u^-, u^+) + \frac{1}{2}(f^*(u^-) + f^*(u^+)). \tag{2.20}$$

6. *The first-order centered (FORCE) flux* [26].

The FORCE flux is given by:

$$\hat{f}^{\mathrm{FORCE}}(u^-, u^+) = \frac{1}{2}\left(\hat{f}^{LF}(u^-, u^+) + \hat{f}^R(u^-, u^+)\right), \tag{2.21}$$

where $\hat{f}^R$ is the second order Richtmyer flux given by

$$\hat{f}^R(u^-, u^+) = f(u^*), \quad u^* = \frac{1}{2}\left(u^- + u^+ - \frac{\Delta t}{\Delta x}(f(u^+) - f(u^-))\right). \tag{2.22}$$

This flux is the average of the LF flux and the second-order Richtmyer flux, hence its viscosity is smaller than that of the LF flux.

Then the final FORCE flux for LWDG is:

$$\hat{F}^{\mathrm{FORCE}}(u^-, u^+) = \hat{f}^{\mathrm{FORCE}}(u^-, u^+) + \frac{1}{2}(f^*(u^-) + f^*(u^+)). \tag{2.23}$$

7. *A flux limiter centered (FLIC) flux* [26].

The general flux limiter approach combines a low order monotone flux and a high-order flux. The FLIC flux, we use has the FORCE flux as the low order flux and the Richtmyer flux as the high-order flux:

$$\hat{f}^{\mathrm{FLIC}}(u^-, u^+) = \hat{f}^{\mathrm{FORCE}}(u^-, u^+) + \phi_{i+1/2}[\hat{f}^R(u^-, u^+) - \hat{f}^{\mathrm{FORCE}}(u^-, u^+)]. \tag{2.24}$$

where $\phi_{i+1/2}$ is a flux limiter. There are several possible choices for the flux limiter such as the superbee, van Leer and the minbee flux limiters. Following [26], for the Euler equation, we use the following procedure: we first define $q = E$ (total energy) and set

$$r^-_{i+1/2} = \frac{\Delta q_{i-1/2}}{\Delta q_{i+1/2}}, \quad r^+_{i+1/2} = \frac{\Delta q_{i+3/2}}{\Delta q_{i+1/2}}$$

where $\Delta q_{i-1/2} = \overline{q}_i - \overline{q}_{i-1}$, and $\overline{q}_i$ is the cell average of q on the cell I_i. We then compute a single flux limiter

$$\phi_{i+1/2} = \min(\phi(r^-_{i+1/2}), \quad \phi(r^+_{i+1/2}))$$

and apply it to all components of the flux. In this paper we use the minbee limiter:

$$\phi(r) = \begin{cases} 0, & r \leqslant 0, \\ r, & 0 \leqslant r \leqslant 1, \\ 1, & r \geqslant 1. \end{cases}$$

Clearly, if $u^- = u^+ = u$, then $\hat{f}^{\mathrm{FLIC}}(u, u) = f(u)$. Hence, even if the FLIC flux depends on more than the two points u^- and u^+ through the limiter $\phi_{i+1/2}$ and we are abusing notations when we denote it by $\hat{f}^{\mathrm{FLIC}}(u^-, u^+)$, it is indeed an essentially two point flux as defined before, hence can be used as a flux for the DG method.

Then the final FLIC flux for LWDG is:

$$\hat{F}^{\mathrm{FLIC}}(u^-, u^+) = \hat{f}^{\mathrm{FLIC}}(u^-, u^+) + \frac{1}{2}(f^*(u^-) + f^*(u^+)). \tag{2.25}$$

8. *GRP flux based on generalized Riemann solver* [25].

The GRP flux at $x_{i+1/2}$ is given by:

$$\hat{F}^{\mathrm{GRP}}(u^-, u^+) = \sum_{\alpha=0}^{K_\alpha} f(u(x_{i+1/2}, t^n + \gamma_\alpha \Delta t))\omega_\alpha, \tag{2.26}$$

where γ_α and ω_α are properly scaled nodes and weights of K_α points Gaussian quadrature, for example, $K_\alpha = 1$ for $k = 1$ and $K_\alpha = 2$ for $k = 2$ cases.

The $u(x_{i+1/2}, t^n + \gamma_\alpha \Delta t)$ in (2.26) can be obtained by:

$$\begin{aligned} u(x_{i+1/2}, t^n + \gamma_\alpha \Delta t) = {} & u(x_{i+1/2}, t^n) + \gamma_\alpha \Delta t u_t(x_{i+1/2}, t^n) \\ & + \frac{(\gamma_\alpha \Delta t)^2}{2} u_{tt}(x_{i+1/2}, t^n) \\ & + \cdots + \frac{(\gamma_\alpha \Delta t)^{k+1}}{(k+1)!} u^{(k+1)}(x_{i+1/2}, t^n). \end{aligned} \tag{2.27}$$

All time derivatives can be replaced by space derivatives using the Lax–Wendroff procedure described in the previous section. $u(x_{i+1/2}, t^n)$ is the solution of equation (1.1) at $(x - x_{i+1/2})/t = 0$ with the initial condition:

$$u(x, t^n) = u^- \quad \text{for} \quad x < x_{i+1/2}, \quad u(x, t^n) = u^+ \quad \text{for} \quad x > x_{i+1/2}.$$

and the space derivatives $\partial_x^q u(x_{i+1/2}, t^n)$, $q = 1, \ldots, k+1$, can be evaluated by the following linearised equations at $(x - x_{i+1/2})/t = 0$:

$$(\partial_x^q u)_t + A(u(x_{i+1/2}, t^n))(\partial_x^q u)_x = 0,$$
$$\partial_x^q u = \partial_x^q u(x_{i+1/2}^-, t^n), \quad \text{for } x < x_{i+1/2},$$
$$\partial_x^q u = \partial_x^q u(x_{i+1/2}^+, t^n), \quad \text{for } x > x_{i+1/2},$$

where $A = df/du$ is Jacobian.

In next section, we will use these fluxes to perform numerical experiments.

3. NUMERICAL RESULTS

In this section, we perform extensive numerical experiments to compare the performance of the LWDG method based on the nine different fluxes outlined in the previous section. The detailed numerical study is mainly performed for the one-dimensional system case, addressing the issues of CPU cost, accuracy, non-oscillatory property, and resolution of discontinuities. Numerical tests are also performed for two-dimensional systems. In all the figures, we plot only the cell averages of the numerical solution. For CPU time comparison, all the computations are performed on a Dell Precision Workstation 370, P4-2.8 with 2GB ram. We denote the LWDG scheme with the flux "X" as LWDG-X, such as LWDG-LF for the LWDG scheme with the LF flux. In our numerical experiments, the CFL numbers are taken as 0.16 and 0.1 for $k = 1$ and $k = 2$ (second and third-order accuracy), respectively.

Example 3.1. We solve the one-dimensional nonlinear system of Euler equations (2.10). The initial condition is set to be $\rho(x, 0) = 1 + 0.2\sin(\pi x)$, $v(x, 0) = 1$, $p(x, 0) = 1$, with a 2-periodic boundary condition. The exact solution is $\rho(x, t) = 1 + 0.2\sin(\pi(x - t))$, $v(x, t) = 1$, $p(x, t) = 1$. We compute the solution up to $t = 2$. In Table I, we provide a CPU time comparison for the LWDG schemes with different fluxes. The ratios of the numerical errors for comparison with the LWDG-LF scheme for the density ρ are shown in Figs. 1 and 2, the coordinate $x = 1, 2, \ldots, 6$ are corresponding to 10, 20, ..., 320 cells, respectively. All schemes achieve their designed orders of accuracy, as expected, and we do not show them to save space.

Table I. CPU time (in Seconds) for the LWDG Methods with Different Fluxes, for the Accuracy Test Problem. Total CPU time for $N = 10$, 20, 40, 80, 160, and 320 cells is Recorded

Flux	LF	LLF	G	EO	HLL	HLLC	FORCE	FLIC	GRP
$k=1$	3.95	3.98	9.44	8.70	5.03	5.06	4.27	4.32	10.45
$k=2$	14.89	14.99	23.06	22.20	16.90	16.97	14.80	15.14	24.49

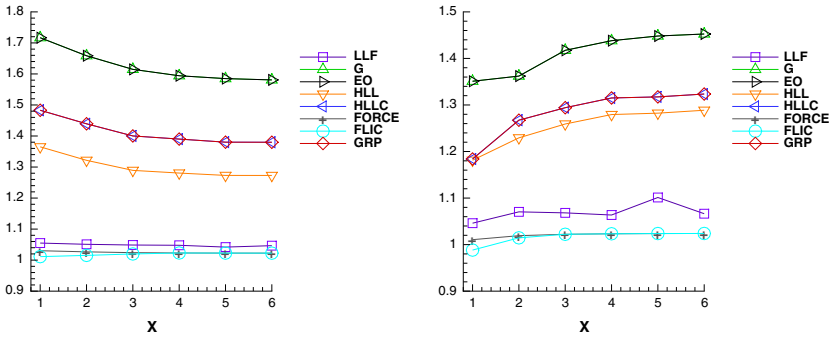

Fig. 1. Euler equations. $\rho(x,0)=1+0.2\sin(\pi x)$, $v(x,0)=1$, $p(x,0)=1$, $t=2$. The ratios of the numerical errors by LWDG with different fluxes compare with those by the LWDG-LF scheme for the density ρ, $k=1$. L_1 (left) and L_∞ (right).

We can see that the LWDG-LF scheme costs the least CPU time for each of the cases $k=1$ and 2, the LWDG-GRP, LWDG-G, and LWDG-EO schemes cost 120% more than that of the LWDG-LF scheme for $k=1$ and 50% for $k=2$, the LWDG-HLL and LWDG-HLLC schemes cost about 30% more than that of the LWDG-LF scheme for $k=1$ and 10% for $k=2$, the LWDG-LLF, LWDG-FORCE, and LWDG-FLIC schemes cost a little more than that of the LWDG-LF scheme for both $k=1$ and 2. Of course, this CPU time comparison depends on our specific implementation of these fluxes and also on the specific test case (for the Godunov flux which has an iteration procedure and may converge with different number of steps for different solutions), but it does give the correct ball-park of the relative CPU costs of the LWDG method using these different fluxes.

On the numerical errors, for the case of $k=1$, the L_1, and L_∞ errors of LWDG-LF are smallest for the same meshes among all schemes, but for the case of $k=2$, errors of LWDG-LF are largest.

For the case of $k=1$, the L_1 errors of LWDG-G, LWDG-EO, LWDG-HLL, LWDG-HLLC, and LWDG-GRP schemes are about 60, 60, 30, 40,

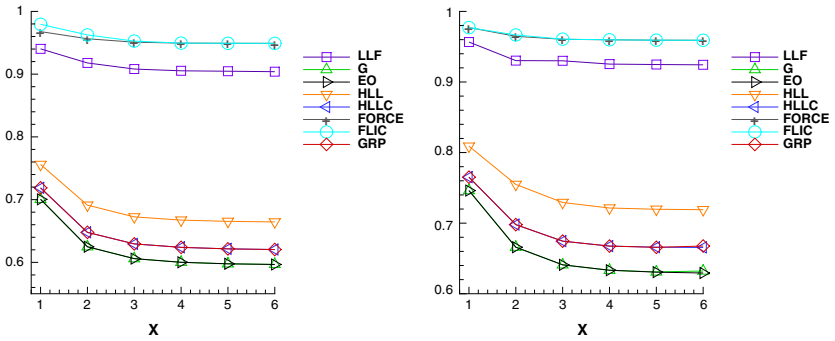

Fig. 2. Euler equations. $\rho(x,0)=1+0.2\sin(\pi x)$, $v(x,0)=1$, $p(x,0)=1$, $t=2$. The ratios of the numerical errors by LWDG with different fluxes compare with those by the LWDG-LF scheme for the density ρ, $k=2$. L_1 (left) and L_∞ (right).

and 40% larger than those by the LWDG-LF scheme for the same meshes, respectively. The L_∞ errors of LWDG-G, LWDG-EO, LWDG-HLL, LWDG-HLLC, and LWDG-GRP schemes are about 40, 40, 20, 30, and 30% larger than those by the LWDG-LF scheme for the same meshes, respectively. For the case of $k=2$, the L_1, and L_∞ errors of LWDG-G, LWDG-EO, LWDG-HLL, LWDG-HLLC, and LWDG-GRP schemes are about 70% of those by the LWDG-LF scheme for the same meshes. The L_1 and L_∞ errors of the LWDG-LLF, LWDG-FORCE, and LWDG-FLIC schemes are similar to those of the LWDG-LF scheme for both $k=1$ and 2 cases. This indicates that, we have to be cautious when discussing about the accuracy advantage of various fluxes as this may depend on the order of accuracy of the scheme.

Example 3.2. We repeat the numerical experiments of the previous example using the following Riemann initial condition for the Lax problem:

$$(\rho, v, p) = (0.445, 0.698, 3.528) \quad \text{for } x \leqslant 0;$$
$$(\rho, v, p) = (0.5, 0, 0.571) \quad \text{for } x > 0.$$

This is a demanding test case in terms of controlling spurious oscillations. The computational domain is $[-5, 5]$ and the final time is $t=1.3$. In Figs. 3 and 4, the computed densities ρ with 200 cells are plotted against the exact solution and against the numerical solution computed by the LWDG-LF scheme on the same mesh, zoomed at the region $1 \leq x \leq 4$ which contains the contact discontinuity and the shock.

The relation of CPU times by the LWDG with different fluxes is similar to that in Example 3.1, we do not show it to save the space. From Figs. 3 and 4,

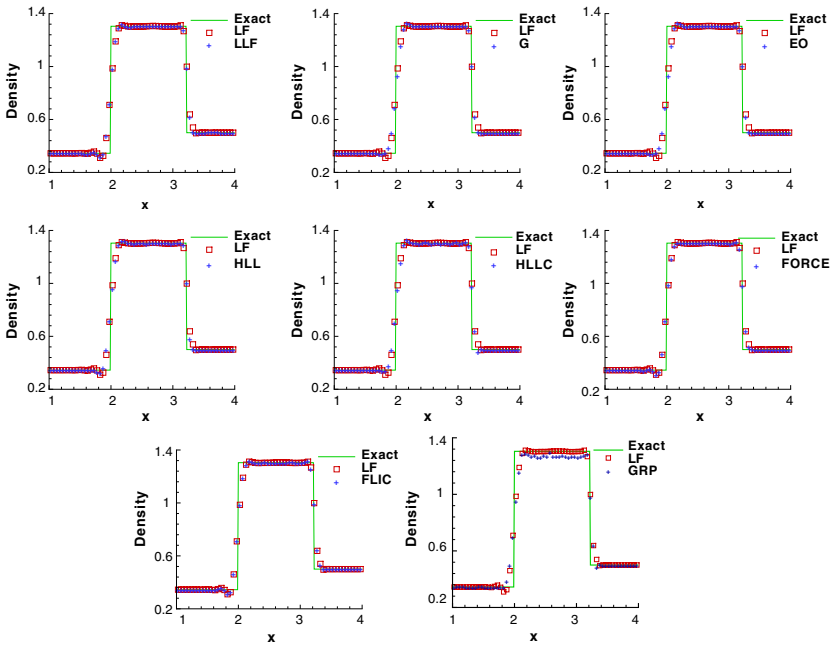

Fig. 3. Lax problem. $t = 1.3$. LWDG with different fluxes, $k = 1$, 200 cells. Density. Solid lines: the exact solution; hollow squares: the results computed by the LWDG-LF scheme; plus symbols: results computed by the LWDG-LLF (top left) LWDG-G (top middle), LWDG-EO (top right), LWDG-HLL (middle left), LWDG-HLLC (middle middle), LWDG-FORCE (middle right), LWDG-FLIC (bottom left), LWDG-GRP (bottom right) schemes.

we can see that the results computed by the LWDG-G, LWDG-EO, LWDG-HLLC, and LWDG-GRP schemes are slightly better than that computed by the LWDG-LF scheme, in terms of the resolution of the discontinuities, and the results computed by all other schemes are similar to that computed by the LWDG-LF scheme.

Example 3.3. We consider the interaction of blast waves of the Euler equation (2.10) with the initial condition:

$$(\rho, v, p) = \begin{cases} (1, 0, 1000) & \text{for } 0 \leqslant x < 0.1 \\ (1, 0, 0.01) & \text{for } 0.1 \leqslant x < 0.9 \\ (1, 0, 100) & \text{for } 0.9 \leqslant x. \end{cases}$$

A reflecting boundary condition is applied to both ends. See [28, 10]. The computational domain is [0, 1], the final time is $t = 0.038$. In Figs. 5 and 6, the computed densities ρ with 400 cells are plotted against the reference

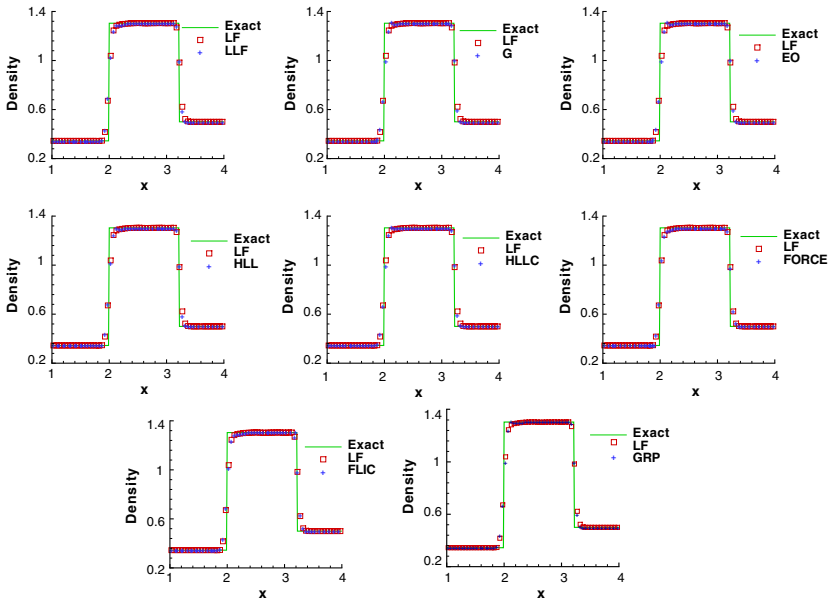

Fig. 4. Lax problem. $t = 1.3$. LWDG with different fluxes, $k = 2$, 200 cells. Density. Solid lines: the exact solution; hollow squares: the results computed by the LWDG-LF scheme; plus symbols: results computed by the LWDG-LLF (top left) LWDG-G (top middle), LWDG-EO (top right), LWDG-HLL (middle left), LWDG-HLLC (middle middle), LWDG-FORCE (middle right), LWDG-FLIC (bottom left), LWDG-GRP (bottom right) schemes.

"exact" solution, computed using a fifth order WENO scheme [12] using 2,000 grid points, and against the solution computed by the LWDG-LF scheme on the same mesh, zoomed at the region $0.53 \leq x \leq 0.88$, which contains the contact discontinuities and shocks in the solution.

The relation of CPU times by the LWDG with different fluxes is also similar to those in the previous examples. Similar to Example 3.2, the resolution of the LWDG-LF scheme is the worst among all schemes. For the case of $k = 1$, the resolution of the LWDG-G, LWDG-EO, LWDG-HLLC, and LWDG-GRP schemes is the best, followed closely by that of the LWDG-HLL scheme, and the resolution of these four schemes is much better than that of the other four schemes. The resolution of the LWDG-LLF, LWDG-FORCE, and LWDG-FLIC schemes is similar to that of the LWDG-LF scheme. For the case $k = 2$, we also observe that the resolution of the LWDG-G, LWDG-EO, LWDG-HLLC, and LWDG-GRP schemes is the best, followed closely by that of the LWDG-HLL and LWDG-LLF schemes, and the resolution of these six schemes is much better than that

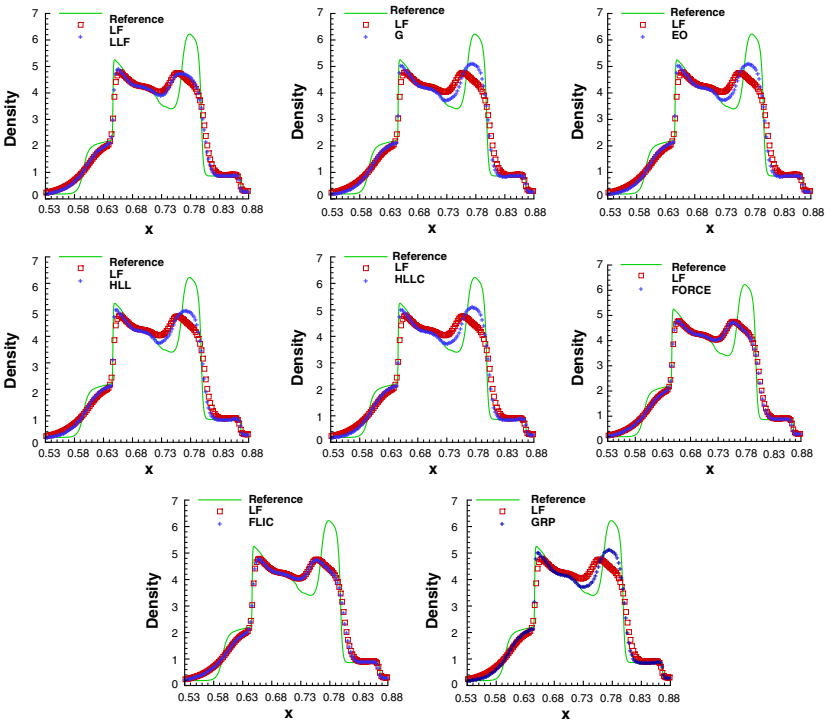

Fig. 5. Blast wave problem. $t = 0.038$. LWDG with different fluxes, $k = 1$, 400 cells. Density. Solid lines: the "exact" reference solution; hollow squares: the results computed by the LWDG-LF scheme; plus symbols: results computed by the LWDG-LLF (top left) LWDG-G (top middle), LWDG-EO (top right), LWDG-HLL (middle left), LWDG-HLLC (middle middle), LWDG-FORCE (middle right), LWDG-FLIC (bottom left), LWDG-GRP (bottom right) schemes.

of the other three schemes. The resolution of the LWDG-FORCE and LWDG-FLIC schemes is similar to that of the LWDG-LF scheme.

Example 3.4. Double Mach reflection. This problem is originally from [28]. The computational domain for this problem is $[0, 4] \times [0, 1]$. The reflecting wall lies at the bottom, starting from $x = 1/6$. Initially a right-moving Mach 10 shock is positioned at $x = 1/6, y = 0$ and makes a 60° angle with the x-axis. For the bottom boundary, the exact post-shock condition is imposed for the part from $x = 0$ to $x = 1/6$ and a reflective boundary condition is used for the rest. At the top boundary, the flow values are set to describe the exact motion of a Mach 10 shock. We compute the solution up to $t = 0.2$. Based on our numerical experimental results for the

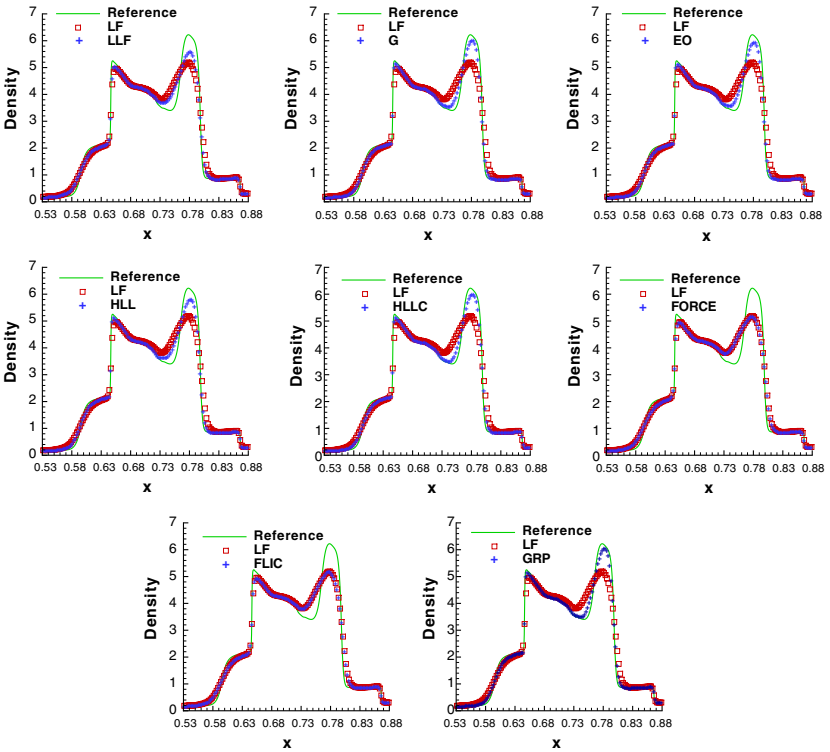

Fig. 6. Blast wave problem. $t = 0.038$. LWDG with different fluxes, $k = 2$, 400 cells. Density. Solid lines: the "exact" reference solution; hollow squares: the results computed by the LWDG-LF scheme; plus symbols: results computed by the LWDG-LLF (top left) LWDG-G (top middle), LWDG-EO (top right), LWDG-HLL (middle left), LWDG-HLLC (middle middle), LWDG-FORCE (middle right), LWDG-FLIC (bottom left), LWDG-GRP (bottom right) schemes.

one-dimensional case, we test all the schemes except LWDG-G, LWDG-EO, and LWDG-GRP methods, which cost significantly more CPU time than the LWDG-LF method.

The LWDG methods with WENO limiters, for two uniform meshes, with 480×120 and 960×240 cells, and two-different orders of accuracy (for $k = 1$ and $k = 2$, second and third-order), are used in the numerical experiments. In Table II we again document the CPU time by the LWDG schemes with different fluxes. We can see that the LWDG-HLL scheme costs about 7–11% more CPU time than the LWDG-LF scheme for the same accuracy order and same mesh, the LWDG-HLLC scheme costs about 11–15% more than the LWDG-

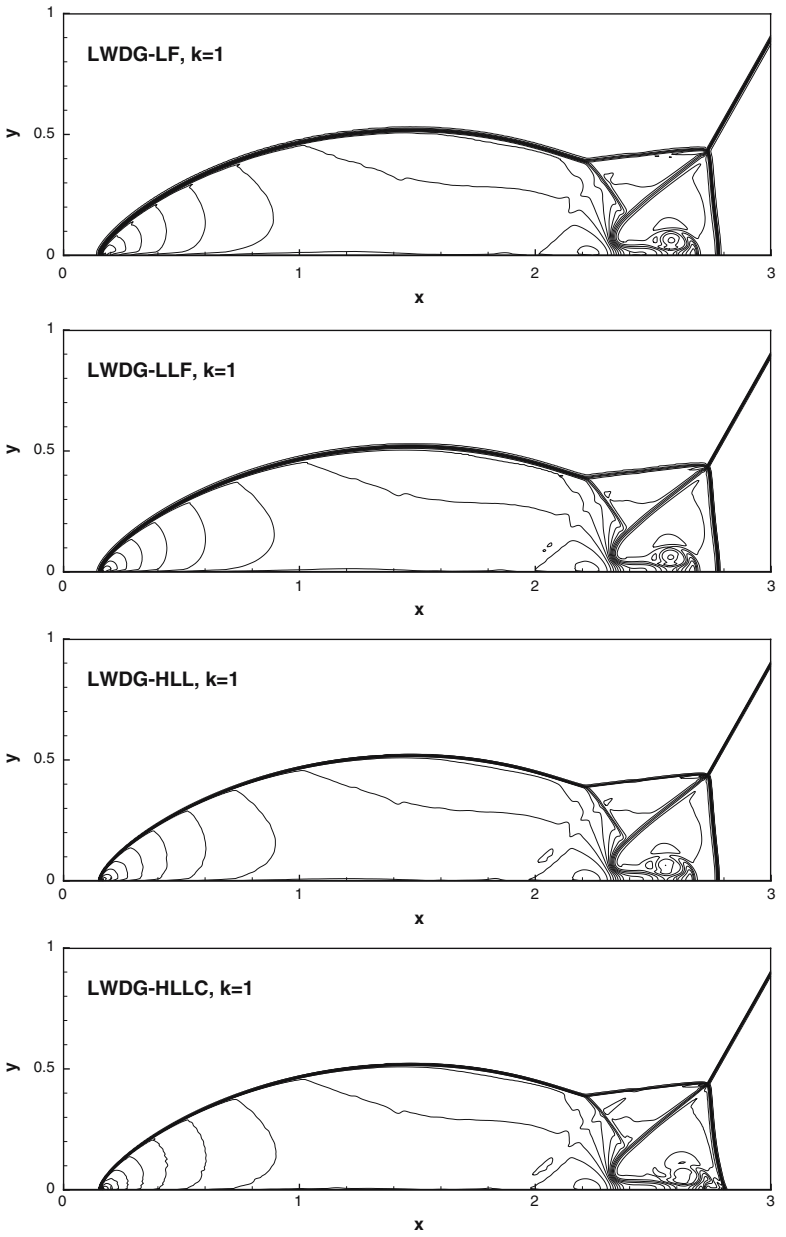

Fig. 7. Double Mach reflection problem. 960×240 cells. 30 equally spaced density contours from 1.5 to 22.7. $k = 1$. From top to bottom are LWDG with LF, LLF, HLL, and HLLC fluxes, respectively.

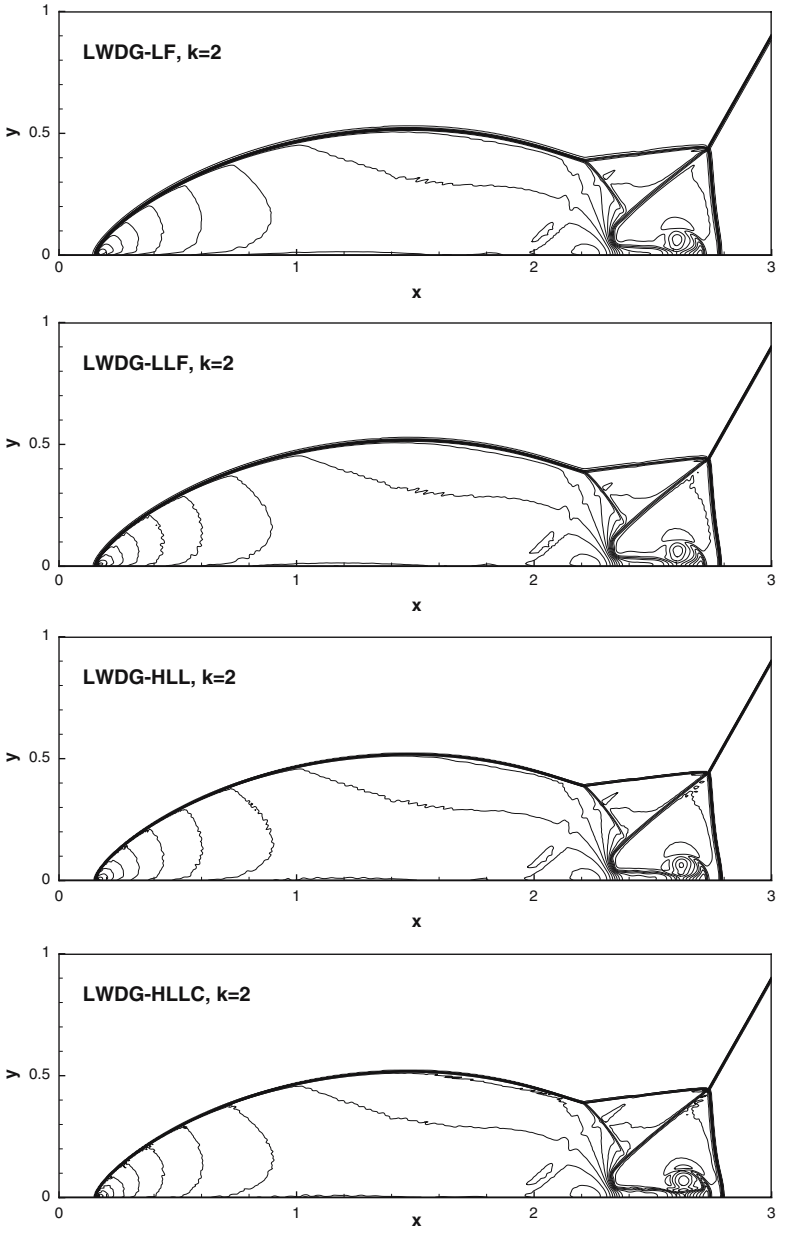

Fig. 8. Double Mach reflection problem. 960×240 cells. 30 equally spaced density contours from 1.5 to 22.7. $k = 2$. From top to bottom are LWDG with LF, LLF, HLL, and HLLC fluxes, respectively.

Table II. CPU time (in Hours) for the LWDG Methods to Compute the Double Mach reflection Problem for the Two Meshes of 480×120 and 960×240 Cells

$N_x \times N_y$	480×120				960×240			
Flux	$k=1$	ratio	$k=2$	ratio	$k=1$	ratio	$k=2$	ratio
LF	0.8076	1.0000	4.7262	1.0000	6.7642	1.0000	38.4635	1.0000
LLF	0.8129	1.0067	4.7555	1.0062	6.8089	1.0066	38.9594	1.0129
HLL	0.8991	1.1133	5.0400	1.0664	7.5333	1.1137	41.1024	1.0686
HLLC	0.9240	1.1441	5.3240	1.1265	7.5885	1.1219	42.9384	1.1163
FORCE	0.8658	1.0721	4.8554	1.0273	7.1543	1.0577	39.7088	1.0324
FLIC	0.8757	1.0844	4.9118	1.0393	7.2417	1.0706	39.8898	1.0371

LF scheme, and LWDG-LLF, LWDG-FORCE, and LWDG-FLIC cost a little more CPU time than LWDG-LF scheme. Since, the result of LWDG-FORCE and LWDG-FLIC is similar to that of LWDG-LF, we plot only the simulation results on the most refined mesh with 960×240 cells by the LWDG-LF, LWDG-LLF, LWDG-HLL and LWDG-HLLC schemes in Figs. 7 and 8, to save space. All the figures are showing 30 equally spaced density contours from 1.5 to 22.7. It seems that all schemes perform similarly well for this test case.

4. CONCLUDING REMARKS

In this paper, we have systematically studied and compared a few different fluxes for the LWDG methods. Extensive one and two-dimensional simulations on the hyperbolic systems of Euler equations indicate that LWDG methods with the LF flux cost the least CPU time among all, but the resolution of solutions on the discontinuities are also the worst among all. The numerical errors of the LWDG method with the LF flux for a smooth problem seem to be the smallest for $k=1$ and largest for $k=2$ among all LWDG schemes. The LWDG methods with the GRP, Godunov or EO fluxes seem to cost significantly more CPU time than the LWDG-LF method. The HLL, HLLC, and LLF fluxes might be good choices as fluxes for the LWDG method when all factors such as the cost of CPU time, numerical errors and resolution of discontinuities in the solution are considered. We also tested the Sod and shock interaction with entropy waves problems [23], the relation of CPU times and the performance of resolution by the LWDG with different fluxes are similar to these of the Lax and blast wave problems, hence they are not shown in the paper to save space.

We have also tested the LWDG methods based on a few other numerical fluxes, such as the second-order Lax–Wendroff (LW) flux and the

Warming–Beam (WB) flux. Our numerical tests indicate that spurious oscillations appear for the Lax shock tube problem for the LWDG-LW and LWDG-WB schemes, and the codes are unstable (they blow up) for the blast wave test case.

ACKNOWLEDGMENTS

Jianxian Qiu Research partially supported by NNSFC grant 10371118 and Nanjing University Talent Development Foundation.

REFERENCES

1. Ben-Artzi, M., and Falcovitz, J. (1984). A second-order Godunov-type scheme for compressible fluid dynamics. *J. Comput. Phys*. **55**, 1–32.
2. Cockburn, B., Hou, S., and Shu, C.-W. (1990). The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case. *Math. Comput*. **54**, 545–581.
3. Cockburn, B., Lin, S.-Y., and Shu, C.-W. (1989). TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one-dimensional systems. *J. Comput. Phys.* **84**, 90–113.
4. Cockburn, B., and Shu, C.-W. (1989). TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework. *Math. Comput*. **52**, 411–435.
5. Cockburn, B., and Shu, C.-W. (1991). The Runge–Kutta local projection P1-discontinuous Galerkin finite element method for scalar conservation laws. *Math. Model. Numer. Anal*. (M^2AN), **25**, 337–361.
6. Cockburn, B., and Shu, C.-W. (1998). The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems. *J. Comput. Phys.* **141**, 199–224.
7. Dumbser, M., and Munz, C. D. (2005). Arbitrary high-order discontinuous Galerkin schemes. In Cordier, S., Goudon, T., Sonnendrcker, E. (eds.), *Numerical Methods for Hyperbolic and Kinetic Problems*, EMS Publishing House, Zurich, Switzerland, pp. 295–333.
8. Engquist, B., and Osher, S. (1981). One sided difference approximation for nonlinear conservation laws. *Math. Comp.* **36**, 321–351.
9. Godunov, S. K. (1959). Finite difference methods for the computation of discontinuous solutions of the equations of fluid dynamics. *Math. Sbornik*, **47**, 271–306.
10. Harten, A., Engquist, B., Osher, S., and Chakravathy, S. (1987). Uniformly high-order accurate essentially non-oscillatory schemes, III. *J. Comput. Phys.* **71**, 231–303.
11. Harten, A., Lax, P. D., and van Leer, B. (1983). On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Rev.* **25**, 35–61.
12. Jiang, G., and Shu, C. W. (1996). Efficient implementation of weighted ENO schemes. *J. Comput. Phys.* **126**, 202–228.
13. Krivodonova, L., Xin, J., Remacle, J.-F., Chevaugeon, N., and Flaherty, J.E. (2004). Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. *Appl. Numer. Math.* **48**, 323–338.
14. Lax, P. D., and Wendroff, B. (1960). Systems of conservation laws. *Commun. Pure Appl. Math.* **13**, 217–237.

15. Osher, S., and Solomon, F. (1982). Upwind difference schemes for hyperbolic conservation laws. *Math. Comput.* **38** (1982), 339–374.
16. Qiu, J., Dumbser, M., and Shu, C.-W. (2005). The discontinuous Galerkin method with Lax–Wendroff type time discretizations. *Comput. Methods Appl. Mech. Eng.* **194**, 4528–4543.
17. Qiu, J., Khoo, B. C., and Shu, C.-W. (2006). A Numerical study for the performance of the Runge–Kutta discontinuous Galerkin method based on different numerical fluxes. *J. Comput. Phys*. **212**, 540–565.
18. Qiu, J., and Shu, C.-W. (2003). Finite difference WENO schemes with Lax–Wendroff-type time discretizations. *SIAM J. Sci. Comput.* **24**, 2185–2198.
19. Qiu, J., and Shu, C.-W., (2005). Runge–Kutta discontinuous Galerkin method using WENO limiters. *SIAM J. Sci. Comput*. **26**, 907–929.
20. Qiu, J., and Shu, C.-W. (2005). A comparison of troubled-cell indicators for Runge–Kutta discontinuous Galerkin methods using WENO limiters. *SIAM J. Sci. Comput.* **27**, 995–1013.
21. Shu, C.-W. (1988). Total-variation-diminishing time discretizations. *SIAM J. Sci. Stat. Comput.* **9**, 1073–1084.
22. Shu, C.-W. (1987). TVB uniformly high-order schemes for conservation laws. *Math. Comput.* **49**, 105–121.
23. Shu, C.-W., and Osher, S., (1989). Efficient implementation of essentially non-oscillatory shock capturing schemes II. *J. Comput. Phys.* **83**, 32–78.
24. Taylor, M. E. (1996). *Partial Differential Equation I: Basic Theory*, Applied Mathematical Sciences, Vol. 115, Springer, Berlin.
25. Titarev, V. A., and Toro, E. F. (2002). ADER: arbitrary high-order Godunov approach. *J. Sci. Comput.* **17**, 609–618.
26. Toro, E. F. (1997). *Riemann Solvers and Numerical Methods for Fluid Dynamics, A Practical Introduction*, Springer, Berlin.
27. Toro, E.F., Spruce, M., and Speares, W. (1994). Restoration of the contact surface in the Harten-Lax-van Leer Riemann solver. *J. Shock Waves* **4**, 25–34.
28. Woodward, P., and Colella, P. (1984). The numerical simulation of two-dimensional fluid flow with strong shocks. *J. Comput. Phys.* **54**, 115–173.