A Study on CFL Conditions for the DG Solution of Conservation Laws on Adaptive Moving Meshes

Min Zhang[1], Weizhang Huang[2] and Jianxian Qiu[3,*]

[1] *School of Mathematical Sciences, Peking University, Beijing 100871, China*
[2] *Department of Mathematics, University of Kansas, Lawrence, Kansas 66045, USA*
[3] *School of Mathematical Sciences, Fujian Provincial Key Laboratory of Mathematical Modeling, High-Performance Scientific Computing, Xiamen University, Xiamen, Fujian 361005, China*

Abstract. The selection of time step plays a crucial role in improving stability and efficiency in the Discontinuous Galerkin (DG) solution of hyperbolic conservation laws on adaptive moving meshes that typically employs explicit stepping. A commonly used selection of time step is a direct extension based on Courant-Friedrichs-Levy (CFL) conditions established for fixed and uniform meshes. In this work, we provide a mathematical justification for those time step selection strategies used in practical adaptive DG computations. A stability analysis is presented for a moving mesh DG method for linear scalar conservation laws. Based on the analysis, a new selection strategy of the time step is proposed, which takes into consideration the coupling of the α-function (that is related to the eigenvalues of the Jacobian matrix of the flux and the mesh movement velocity) and the heights of the mesh elements. The analysis also suggests several stable combinations of the choices of the α-function in the numerical scheme and in the time step selection. Numerical results obtained with a moving mesh DG method for Burgers' and Euler equations are presented. For comparison purpose, numerical results obtained with an error-based time step-size selection strategy are also given.

AMS subject classifications: 65M50, 65M60

Key words: Discontinuous Galerkin method, adaptive mesh, moving mesh, CFL condition, stability.

1. Introduction

We are concerned with the stability of the discontinuous Galerkin (DG) solution of

---

*Corresponding author. *Email addresses:* `minzhang@math.pku.edu.cn` (M. Zhang), `jxqiu@xmu.edu.cn` (J. Qiu), `whuang@ku.edu` (W. Huang)

conservation laws on adaptive moving meshes. The DG method is a powerful numerical tool for use in the simulation of hyperbolic problems. It was first used by Reed and Hill [31] for the steady radiation transport equation and studied theoretically by Lesaint and Raviart [25]. The method was extended to conservation laws by Cockburn and Shu (and their coworkers) in a series of papers [6–11]. The DG method has the advantages of high-order accuracy, geometric flexibility, easy use with mesh adaptivity, local data structure, high parallel efficiency, and a good foundation for theoretical analysis. The DG method has been used widely in scientific and engineering computation. Meanwhile, conservation laws typically exhibit discontinuous structures such as shock waves, rarefaction waves, and contact discontinuities and are amenable to mesh adaptation in their numerical solution to enhance numerical resolution and computational efficiency. It is natural to combine the DG method with mesh adaptation method for the solution of conservation laws.

A large amount of work has been done in this area. For example, Bey and Oden [3] combined the hp-method with the DG method for conservation laws and Li and Tang [26] solved two-dimensional conservation laws using a rezoning moving mesh DG method where the physical variables are interpolated from the old mesh to the new one using conservative interpolation schemes. Mackenzie and Nicola [29] solved the Hamilton-Jacobi equation by the DG method using a moving mesh method based on the moving mesh partial differential equation (MMPDE) strategy [21, 22]. Vilar *et al.* [36] studied a DG discretization for solving the two-dimensional gas dynamics equations in Lagrangian formulation. More recently, Uzunca *et al.* [35] employed a moving mesh symmetric interior penalty Galerkin method (SIPG) to solve PDEs with traveling waves. Luo *et al.* considered a quasi-Lagrange moving mesh DG method (MMDG) for conservation laws [27] and multi-component flows [28]. Zhang *et al.* studied the MMDG solution for the radiative transfer equation [38, 39] and shallow water equations (SWEs) [40, 41]. Zhang *et al.* [43] develop a arbitrary Lagrangian-Eulerian discontinuous Galerkin (ALE-DG) methods for the SWEs. Wang *et al.* [37] developed a reconstructed DG Method for compressible flows in Lagrangian formulation.

In principle, any marching scheme (e.g., see Hairer and Wanner [16]) can be used for the time integration of DG computations of hyperbolic conservation laws, including explicit and implicit Runge-Kutta methods [13, 14] and multi-step methods [32]. Nevertheless, explicit schemes have been widely used in these computations. There are at least two considerations for this. First, as we can see later, the stability condition for explicit schemes when applied to hyperbolic equations typically requires the time step-size to be proportional to the minimum mesh element size, which is considered acceptable in practical computations with a uniform mesh. Second, due to the highly nonlinear and hyperbolic nature of conservation laws, there exists hardly any efficient solver for nonlinear algebraic systems (whose linearization is typically non-symmetric and non-definite, and whose solution contains discontinuity, such as shock wave) resulting from the implicit temporal discretization. As such, it does not seem worth the trouble to increase the time step-size using implicit schemes when a uniform mesh is

used. However, this can be a different situation when an adaptive mesh is employed where some mesh elements can become very small. While implicit schemes for DG computations on adaptive meshes deserve further investigations, in this work we focus on explicit schemes and their stability on adaptive moving meshes.

Consider the conservation laws in the form

$$U_t + \nabla \cdot \mathbf{F}(U, \boldsymbol{x}) = 0, \quad \forall \boldsymbol{x} \in \Omega, \tag{1.1}$$

where Ω is a polygonal/polyhedral domain in $\mathbb{R}^d$ ($d \geq 1$), $U = (u_1, \ldots, u_m)^T$ ($m \geq 1$) is the unknown function, and the flux $\mathbf{F}(U, \boldsymbol{x})$ is an m-by-d matrix-valued function of U and $\boldsymbol{x}$. A commonly used selection of time step in adaptive DG computations (e.g., see [27, 40, 41]) is

$$\Delta t = \frac{C_{cfl}\sigma_{h,\min}}{\alpha_h}, \tag{1.2}$$

where $\sigma_{h,\min}$ is the minimum height or diameter of the mesh elements, C_{cfl} is a positive parameter, α_h is the maximum absolute value of the eigenvalues of the Jacobian matrix of $(\mathbf{F} - U\dot{\boldsymbol{X}}) \cdot \boldsymbol{n}$ (with respect to U) taken over all of the edges, $\boldsymbol{n}$ is the unit outward normal vector of the edges, and $\dot{\boldsymbol{X}}$ is the piecewise linear mesh velocity function. The choice (1.2) is a direct extension of the CFL conditions studied and used in DG computation for fixed meshes (e.g., see [9, 12]). Some researchers have used a different yet mathematical equivalent form,

$$\Delta t = \frac{C_{cfl}}{\alpha_h \max\limits_{K \in \mathcal{T}_h} \left(\frac{1}{|K|} \sum\limits_{e \in \partial K} |e| \right)}, \tag{1.3}$$

where K is an element of the mesh $\mathcal{T}_h$, ∂K is the boundary of K, e is an edge of K, and $|K|$ and $|e|$ denote the area of K and the length of e, respectively. This condition has also been established by Zhang *et al.* [42] for positivity preservation for fixed unstructured triangular meshes. To show the equivalence between (1.2) and (1.3), we notice that, for a simplex K,

$$\max_{e \in \partial K} |e| \leq \sum_{e \in \partial K} |e| \leq (d+1) \max_{e \in \partial K} |e|,$$

which implies

$$\frac{d}{\sigma_{K,\min}} \leq \frac{1}{|K|} \sum_{e \in \partial K} |e| \leq \frac{d(d+1)}{\sigma_{K,\min}},$$

where $\sigma_{K,\min}$ denotes the minimum height of K and we have used a geometric property of simplexes, $|K| = (1/d)\sigma_{K,\min} \max_{e \in \partial K} |e|$. Taking maximum over all elements, we get

$$\frac{\sigma_{h,\min}}{d(d+1)} \leq \frac{1}{\max\limits_{K} \left(\frac{1}{|K|} \sum\limits_{e \in \partial K} |e| \right)} \leq \frac{\sigma_{h,\min}}{d}, \tag{1.4}$$

which gives the equivalence of (1.2) and (1.3).

While the condition (1.2) or (1.3) appears to work well in existing adaptive DG computation for conservation laws, there lacks a theoretical justification of them for non-uniform and moving meshes. One may also wonder if the coupling between physical quantities and mesh elements can be taken into consideration for time step selection. The objective of this work is to study these issues. To be specific, we consider a quasi-Lagrange MMDG method [27, 38] with the Lax-Friedrichs (LF) flux for (1.1). A CFL condition for the L^1 stability of the MMDG method with P^0 elements is then established and analyzed for scalar linear equations (cf. Proposition 3.1), which provides a theoretical justification for the stability of the method. Moreover, based on this analysis, for the MMDG method with P^k elements ($k \geq 1$) and general conservation laws we propose to choose Δt as

$$\Delta t = \frac{C_{cfl}}{\max\limits_{K \in \mathcal{T}_h} \left(\frac{1}{|K|} \sum\limits_{e \in \partial K} \alpha_e |e| \right)} = \frac{C_{cfl}}{\max\limits_{K \in \mathcal{T}_h} \left(\sum\limits_{e \in \partial K} \alpha_e \frac{|e|}{|K|} \right)}, \tag{1.5}$$

where $\alpha_e = \alpha_e(U, \dot{\boldsymbol{X}})$ is the maximum absolute value of the eigenvalues of the Jacobian matrix of $(\mathbf{F} - U\dot{\boldsymbol{X}}) \cdot \boldsymbol{n}$ (with respect to U) taken over edge e. This choice is very similar to (1.3). Indeed, it reduces to (1.3) when α_e is replaced with the global α_h. However, unlike (1.3), the condition (1.5) takes into consideration the spatial variation of α and its coupling with the element height (i.e., $|K|/|e|$). Moreover, it shows that Δt can be increased if the mesh velocity $\dot{\boldsymbol{X}}$ can be chosen to minimize α_e in regions where the element height is relatively small. An example of this is Lagrangian-type methods (e.g. [17, 18, 24]) where the mesh velocity is taken as the fluid velocity. On the other hand, the mesh movement can be determined by other considerations. For example, in the current work we consider the use of the MMPDE moving mesh method to determine the mesh movement based on solution-Hessian. In this case, the mesh velocity does not necessarily minimize α_e. If α_e does not change significantly over the domain, then (1.5) is mathematically equivalent to (1.2) and Δt is determined essentially by the minimum height of the mesh elements.

The CFL condition such as (1.5) provides a selection strategy for time step-size based on stability. This CFL-condition-based strategy is widely used in the DG computation of hyperbolic conservation laws. On the other hand, it is common practice to use an error-based time step-size selection strategy in the computation of ordinary differential equations (e.g., see Hairer and Wanner [16]). It is worth studying how error-based time step-size selection strategies fare in the DG computation of hyperbolic conservation laws. To this end, we use the third-order explicit Strong Stability Preserving (SSP) embedded RK (SSP-ERK(4,3)) pair of [13] and the standard PI controller step-size selection strategy (e.g., see Hairer and Wanner [16]) and present numerical examples to demonstrate the feasibility of this approach.

The MMPDE moving mesh method [21, 22] is used to generate adaptive moving meshes for the numerical examples presented in this work. A key idea of the MM-PDE method is to view any nonuniform mesh as a uniform one in some Riemannian metric specified by a tensor $\mathbb{M} = \mathbb{M}(\boldsymbol{x})$, a symmetric and uniformly positive

definite matrix-valued function that provides the information needed for determining the size, shape, and orientation of the mesh elements throughout the domain. It has been shown analytically and numerically in [20] that the moving mesh generated by the MMPDE method stays nonsingular (free of tangling) if the metric tensor is bounded and the initial mesh is nonsingular. It is worth pointing out that other adaptive moving mesh methods (such as Lagrangian-type methods) can also be used; e.g., see [1–3, 5, 15, 17, 18, 22, 24, 30, 34, 36, 37] and references therein.

An outline of the paper is as follows. The MMDG method is described in Section 2 and the L^1 stability analysis of the method with P^0 elements is carried out for scalar linear equations in Section 3. Numerical examples are presented in Section 4. In these examples, the MMPDE moving mesh method is used to generate adaptive moving meshes. The section also contains the descriptions of the procedure of the MMDG method and the definition of the metric tensor that is used to control mesh concentration. The conclusions are given in the final Section 5.

2. The moving mesh DG method

In this section we describe a quasi-Lagrange MMDG method [27, 38, 41] for solving hyperbolic conservation laws in the form (1.1).

To start with, we assume that a sequence of simplicial meshes, $\mathcal{T}_h^0, \mathcal{T}_h^1, \ldots$, have been given for Ω at time instants $t_0, t_1, \ldots$ and these meshes have the same number of elements and vertices and the same connectivity. For numerical results presented in this work, we use the MMPDE moving mesh method [19, 21, 22] to generate these meshes (cf. Section 4). For any $n \geq 0$ and $t \in [t_n, t_{n+1}]$, we define $\mathcal{T}_h(t)$ as the mesh with the same number of elements (N) and vertices (N_v) and the same connectivity as $\mathcal{T}_h^n$, and having the vertices given by

$$\boldsymbol{x}_i(t) = \frac{t - t_n}{\Delta t_n}\boldsymbol{x}_i^n + \frac{t_{n+1} - t}{\Delta t_n}\boldsymbol{x}_i^{n+1}, \quad i = 1, \ldots, N_v, \quad \Delta t_n = t_{n+1} - t_n. \tag{2.1}$$

Define the piecewise linear mesh velocity function as

$$\dot{\boldsymbol{X}}(\boldsymbol{x}, t) = \sum_{i=1}^{N_v} \dot{\boldsymbol{x}}_i \phi_i(\boldsymbol{x}, t) = \sum_{i=1}^{N_v} \frac{\boldsymbol{x}_i^{n+1} - \boldsymbol{x}_i^n}{\Delta t_n}\phi_i(\boldsymbol{x}, t), \quad t \in [t_n, t_{n+1}], \tag{2.2}$$

where $\phi_i(\boldsymbol{x}, t)$ is the linear basis function at $\boldsymbol{x}_i$ and $\dot{\boldsymbol{x}}_i$ is the nodal velocity. For any element $K \in \mathcal{T}_h(t)$, let $P^k(K)$ be the set of polynomials of degree at most $k \geq 1$ on K. The DG finite element space is defined as

$$\mathcal{V}_h^k(t) = \left\{ u \in L^2(\Omega) : u|_K \in P^k(K), \ \forall K \in \mathcal{T}_h(t) \right\}. \tag{2.3}$$

We now are ready to describe the DG discretization of (1.1). Multiplying it with an arbitrary test function $\phi \in \mathcal{V}_h^k(t)$, integrating the resulting equation over K, and using

the Reynolds transport theorem, we get

$$\frac{d}{dt} \int_K U_h \phi d\boldsymbol{x} - \int_K \mathbf{H}(U_h, \boldsymbol{x}) \cdot \nabla \phi d\boldsymbol{x} + \sum_{e \in \partial K} \int_e \phi \hat{\mathbf{H}}(U_h, \boldsymbol{x}) ds = 0, \qquad (2.4)$$

where $U_h = U_h(\boldsymbol{x}, t)$, $\mathbf{H}(U_h, \boldsymbol{x}) = \mathbf{F}(U_h, \boldsymbol{x}) - U_h \dot{\boldsymbol{X}}$ is the new flux accounting for the effect of mesh movement, $\hat{\mathbf{H}}(U_h, \boldsymbol{x}) \approx \mathbf{H}(U_h, \boldsymbol{x}) \cdot \boldsymbol{n}$ is a numerical flux, and $\boldsymbol{n}$ is the unit outward normal to edge e. Let λ_m's be the eigenvalues of the Jacobian matrix of $\mathbf{H}(U_h, \boldsymbol{x}) \cdot \boldsymbol{n}$ with respect to U_h and $U_{h,K}^{int}$ and $U_{h,K}^{ext}$ be the values of U_h in K and K', respectively, where K' is the element sharing e with K. Define the α-function as

$$\alpha(U_h, \boldsymbol{x}) = \max_m \Big(\max \big(\big| \lambda_m(U_{h,K}^{int}, \boldsymbol{x}) \big|, \big| \lambda_m(U_{h,K}^{ext}, \boldsymbol{x}) \big| \big) \Big). \qquad (2.5)$$

In this work we consider the global/local Lax-Friedrichs (LF) numerical flux,

$$\hat{\mathbf{H}}(U_h, \boldsymbol{x}) = \frac{1}{2} \Big(\big(\mathbf{H}(U_{h,K}^{int}, \boldsymbol{x}) + \mathbf{H}(U_{h,K}^{ext}, \boldsymbol{x}) \big) \cdot \boldsymbol{n} - \alpha_{LF} \big(U_{h,K}^{ext} - U_{h,K}^{int} \big) \Big), \quad \boldsymbol{x} \in e \subset \partial K,$$

where α_{LF} denotes a choice of the α-function in this numerical flux. In practical computation, the second and third terms in (2.4) are calculated typically by Gaussian quadrature rules. We denote those by

$$\int_K \mathbf{H}(U_h, \boldsymbol{x}) \cdot \nabla \phi d\boldsymbol{x} \approx |K| \sum_{\boldsymbol{x}_G^K} w_G^K \big(\mathbf{H}(U_h, \boldsymbol{x}) \cdot \nabla \phi \big) \big|_{\boldsymbol{x}_G^K} \quad \text{with} \quad \sum w_G^K = 1,$$

$$\int_e \phi \hat{\mathbf{H}}(U_h, \boldsymbol{x}) ds \approx |e| \Big(\sum_{\boldsymbol{x}_G^e} w_G^e \big(\phi \hat{\mathbf{H}}(U_h, \boldsymbol{x}) \big) \big|_{\boldsymbol{x}_G^e} \Big) \qquad \text{with} \quad \sum w_G^e = 1, \qquad (2.6)$$

where $\boldsymbol{x}_G^K$'s and $\boldsymbol{x}_G^e$'s are the Gauss points on K and e, respectively. For the analytical analysis in the next section, we assume that the weights w_G^K's and w_G^e's are nonnegative. Combining the above with (2.4) we obtain the semi-discrete MMDG scheme as

$$\frac{d}{dt} \int_K U_h \phi d\boldsymbol{x} - |K| \sum_{\boldsymbol{x}_G^K} w_G^K \big(\mathbf{H}(U_h, \boldsymbol{x}) \cdot \nabla \phi \big) \big|_{\boldsymbol{x}_G^K}$$

$$+ \sum_{e \in \partial K} |e| \Big(\sum_{\boldsymbol{x}_G^e} w_G^e \big(\phi \hat{\mathbf{H}}(U_h, \boldsymbol{x}) \big) \big|_{\boldsymbol{x}_G^e} \Big) = 0. \qquad (2.7)$$

One choice of α_{LF} is the point-wise value of $\alpha(U_h, \boldsymbol{x})$, i.e.,

$$\alpha_p = \alpha(U_h, \boldsymbol{x})|_{\boldsymbol{x}_G^e}$$

$$= \max_m \Big(\max \big(\big| \lambda_m(U_{h,K}^{int}, \boldsymbol{x})|_{\boldsymbol{x}_G^e} \big|, \big| \lambda_m(U_{h,K}^{ext}, \boldsymbol{x})|_{\boldsymbol{x}_G^e} \big| \big) \Big), \quad \boldsymbol{x}_G^e \in e \subset \partial K. \qquad (2.8)$$

It is worth pointing out that (2.8) is calculated pointwise. We can choose it differently, for example, by taking the maximum value over each edge (denoted as α_e), or all edges in the mesh (denoted as α_h), i.e.,

$$\alpha_e = \max_{\boldsymbol{x}_G^e} \alpha(U_h, \boldsymbol{x})|_{\boldsymbol{x}_G^e}$$

$$= \max_{\boldsymbol{x}_G^e, m} \Big(\max \big(\big|\lambda^m(U_{h,K}^{int}, \boldsymbol{x})|_{\boldsymbol{x}_G^e}\big|, \big|\lambda^m(U_{h,K}^{ext}, \boldsymbol{x})|_{\boldsymbol{x}_G^e}\big| \big) \Big), \tag{2.9}$$

$$\alpha_h = \max_{K, e, \boldsymbol{x}_G^e} \alpha(U_h, \boldsymbol{x})|_{\boldsymbol{x}_G^e}$$

$$= \max_{K, e, \boldsymbol{x}_G^e, m} \Big(\max \big(\big|\lambda_m(U_{h,K}^{int}, \boldsymbol{x})|_{\boldsymbol{x}_G^e}\big|, \big|\lambda_m(U_{h,K}^{ext}, \boldsymbol{x})|_{\boldsymbol{x}_G^e}\big| \big) \Big). \tag{2.10}$$

It is the remark that one choice of the α_{LF} in local LF in the computation (e.g., see [12, 27]) is taken based on cell average, i.e.,

$$\tilde{\alpha}_e = \max_{\boldsymbol{x}_G^e, m} \Big(\max \big(\big|\lambda_m(\bar{U}_{h,K}, \boldsymbol{x}_G^e)\big|, \big|\lambda_m(\bar{U}_{h,K'}, \boldsymbol{x}_G^e)\big| \big) \Big), \tag{2.11}$$

where $\bar{U}_{h,K}$ is the average of U_h on K, $\bar{U}_{h,K'}$ is the average of U_h on K', and K and K' are sharing $e \in \partial K$.

A fully discrete MMDG scheme can be obtained by discretizing (2.7) in time. A stability analysis is presented in the next section for a simple case with the explicit Euler scheme and numerical results are presented in Section 4 with a third-order strong-stability-preserving (SSP) Runge-Kutta scheme.

3. CFL conditions on adaptive moving meshes

Generally speaking, it is difficult, if not impossible, to obtain a CFL condition rigorously for a fully discrete version of the MMDG scheme (2.7) for general conservation law (1.1). To gain insight into the stability, we consider a simple situation with P^0-DG for spatial discretization, the explicit Euler scheme for time integration, and for linear scalar conservation laws with the flux $\mathbf{F} = \boldsymbol{a}(\boldsymbol{x}, t)U$. We also assume that Ω is cubical, periodic boundary conditions are used, and $\boldsymbol{a}(\boldsymbol{x}, t)$ is periodic in each coordinate direction.

Under these assumptions, we have

$$\mathbf{H}(U_h, \boldsymbol{x}) = \big(\boldsymbol{a}(\boldsymbol{x}, t) - \dot{\boldsymbol{X}} \big) U_h, \quad \lambda(\boldsymbol{x}, t) = \big(\boldsymbol{a}(\boldsymbol{x}, t) - \dot{\boldsymbol{X}} \big) \cdot \boldsymbol{n}, \quad \alpha(\boldsymbol{x}, t) = |\lambda(\boldsymbol{x}, t)|.$$

Moreover,

$$\hat{\mathbf{H}}(U_h, \boldsymbol{x}) = \frac{1}{2} \Big(\big(\mathbf{H}(U_{h,K}^{int}, \boldsymbol{x}) + \mathbf{H}(U_{h,K}^{ext}, \boldsymbol{x}) \big) \cdot \boldsymbol{n} - \alpha_{LF}(\boldsymbol{x}, t) \big(U_{h,K}^{ext} - U_{h,K}^{int} \big) \Big)$$

$$= \frac{1}{2} \Big(\lambda(\boldsymbol{x}, t) \big(U_{h,K}^{int} + U_{h,K}^{ext} \big) - \alpha_{LF}(\boldsymbol{x}, t) \big(U_{h,K}^{ext} - U_{h,K}^{int} \big) \Big)$$

$$= \frac{\alpha_{LF}(\boldsymbol{x}, t) + \lambda(\boldsymbol{x}, t)}{2} U_{h,K}^{int} - \frac{\alpha_{LF}(\boldsymbol{x}, t) - \lambda(\boldsymbol{x}, t)}{2} U_{h,K}^{ext}.$$

Applying the explicit Euler scheme to (2.7) and taking $\phi = 1$, we get

$$|K^{n+1}|\bar{U}_{h,K}^{n+1} = |K^n|\bar{U}_{h,K}^n - \Delta t_n \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e, \quad \hat{\mathbf{H}}(U_h, \boldsymbol{x})\big|_{\boldsymbol{x}_G^e} = 0, \qquad (3.1)$$

where $\bar{U}_{h,K}^n$ is the average of U_h^n on K^n, $\bar{U}_{h,K}^{n+1}$ is the average of U_h^{n+1} on K^{n+1}, and K^n and K^{n+1} are the corresponding elements in $\mathcal{T}_h^n$ and $\mathcal{T}_h^{n+1}$, respectively.

Proposition 3.1. *The MM P^0-DG scheme (3.1) with $\mathbf{F} = \boldsymbol{a}(\boldsymbol{x}, t)U$ is L^1-stable under the CFL condition*

$$\Delta t_n \leq \frac{1}{\max\limits_{K^n} \left(\frac{1}{|K^n|} \sum\limits_{e \in \partial K^n} |e| \sum\limits_{\boldsymbol{x}_G^e} w_G^e \, \alpha_{CFL}^n(\boldsymbol{x}_G^e) \right)}, \qquad (3.2)$$

where $\alpha_{CFL}^n(\boldsymbol{x}) \geq \alpha_{LF}^n(\boldsymbol{x})$ and the subscripts LF and CFL stand for the LF numerical flux and CFL condition, respectively.

Proof. From the assumption of P^0-DG, for any $\boldsymbol{x} \in e$ of K^n, we have

$$U_{h,K}^{int}|_{\boldsymbol{x}_G^e} = \bar{U}_{h,K}^{int} = \bar{U}_{h,K}, \quad U_{h,K}^{ext}|_{\boldsymbol{x}_G^e} = \bar{U}_{h,K}^{ext} = \bar{U}_{h,K'},$$

where K' is the element sharing e with K. With this, (3.1) can be rewritten as

$$|K^{n+1}|\bar{U}_{h,K}^{n+1} = |K^n|\bar{U}_{h,K}^n - \Delta t_n \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n + \lambda^n}{2} U_{h,K}^{n,int} - \frac{\alpha_{LF}^n - \lambda^n}{2} U_{h,K}^{n,ext} \right)\Bigg|_{\boldsymbol{x}_G^e}$$

$$= |K^n|\bar{U}_{h,K}^n - \Delta t_n \sum_{e \in \partial K^n} |e|\bar{U}_{h,K}^n \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right)$$

$$+ \Delta t_n \sum_{e \in \partial K^n} |e|\bar{U}_{h,K}^{n,ext} \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) - \lambda^n(\boldsymbol{x}_G^e)}{2} \right)$$

$$= |K^n|\bar{U}_{h,K}^n \left[1 - \frac{\Delta t_n}{|K^n|} \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right) \right]$$

$$+ \Delta t_n \sum_{e \in \partial K^n} |e|\bar{U}_{h,K}^{n,ext} \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) - \lambda^n(\boldsymbol{x}_G^e)}{2} \right). \qquad (3.3)$$

Notice that, for any $\boldsymbol{x}_G^e \in e \subset \partial K^n$, we have

$$0 \leq \alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e) \leq 2\alpha_{CFL}^n(\boldsymbol{x}_G^e), \quad 0 \leq \alpha_{LF}^n(\boldsymbol{x}_G^e) - \lambda^n(\boldsymbol{x}_G^e).$$

From the CFL condition (3.2), we have

$$1 - \frac{\Delta t_n}{|K^n|} \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right) \geq 0.$$

From this, taking the absolute value on both side of (3.3) gives

$$|K^{n+1}||\bar{U}_{h,K}^{n+1}| \leq |K^n||\bar{U}_{h,K}^n| \left[1 - \frac{\Delta t_n}{|K^n|} \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right) \right]$$
$$+ \Delta t_n \sum_{e \in \partial K^n} |e||\bar{U}_{h,K}^{n,ext}| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) - \lambda^n(\boldsymbol{x}_G^e)}{2} \right).$$

Summing this over all elements, we get

$$\sum_{K^{n+1}} |K^{n+1}||\bar{U}_{h,K}^{n+1}| \leq \sum_{K^n} |K^n||\bar{U}_{h,K}^n| \left[1 - \frac{\Delta t_n}{|K^n|} \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right) \right]$$
$$+ \Delta t_n \sum_{K^n} \sum_{e \in \partial K^n} |e||\bar{U}_{h,K}^{n,ext}| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) - \lambda^n(\boldsymbol{x}_G^e)}{2} \right).$$

We notice that $\sum_{e \in \partial K^n} |e|\bar{U}_{h,K}^{n,ext}(\cdots)$ goes over all neighboring elements (K') of K^n and each term can be considered to be associated with K' but with $\boldsymbol{n}$ being changed to $-\boldsymbol{n}$ (because the unit outward normal of e in view of K^n is opposite to the unit outward normal of e in view of K'). From the periodicity assumption on the boundary conditions and $a(\boldsymbol{x},t)$, we can rewrite the second term on the right-hand side of the above equation as

$$\Delta t_n \sum_{K^n} \sum_{e \in \partial K^n} |e||\bar{U}_{h,K}^{n,ext}| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) - \lambda^n(\boldsymbol{x}_G^e)}{2} \right)$$
$$= \Delta t_n \sum_{K^n} \sum_{e \in \partial K^n} |e||\bar{U}_{h,K}^n| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right).$$

Combining these, we have

$$\sum_{K^{n+1}} |K^{n+1}||\bar{U}_{h,K}^{n+1}| \leq \sum_{K^n} |K^n||\bar{U}_{h,K}^n| \left[1 - \frac{\Delta t_n}{|K^n|} \sum_{e \in \partial K^n} |e| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right) \right]$$
$$+ \Delta t_n \sum_{K^n} \sum_{e \in \partial K^n} |e||\bar{U}_{h,K}^n| \sum_{\boldsymbol{x}_G^e} w_G^e \left(\frac{\alpha_{LF}^n(\boldsymbol{x}_G^e) + \lambda^n(\boldsymbol{x}_G^e)}{2} \right)$$
$$= \sum_K |K^n||\bar{U}_{h,K}^n|.$$

Hence, the scheme is L^1-stable. $\qquad\qquad\square$

It should be pointed out that the CFL condition (3.2) is only a sufficient condition. Nevertheless, it offers several insights on the maximum time step allowed by stability. We elaborate these in the following remarks.

Remark 3.1. Condition (3.2) involves an important factor $|e|/|K|$. It is known that

$$\frac{|e|}{|K|} = \frac{d}{\sigma_e}, \quad \forall e \in \partial K, \tag{3.4}$$

where σ_e is an element height defined as the distance between e and the vertex of K opposite to e. Using this, we can rewrite (3.2) into

$$\Delta t \leq \frac{1}{d \max\limits_{K} \left(\sum\limits_{e \in \partial K} \frac{1}{\sigma_e} \sum\limits_{\boldsymbol{x}_G^e} w_G^e \, \alpha_{CFL}(\boldsymbol{x}_G^e) \right)}. \tag{3.5}$$

This indicates that the allowed maximum time step depends on the coupling between σ_e and $\alpha_{CFL}(\boldsymbol{x}_G^e)$. Recall that α (cf. (2.5)) is defined as the maximum of the eigenvalue of the Jacobian matrix of $(\mathbf{F}(U_h, \boldsymbol{x}) - U_h \dot{\boldsymbol{X}}) \cdot \boldsymbol{n}$. Thus, if the mesh velocity can be chosen to minimize α_{CFL} (such as in Lagrangian-type methods; e.g., see [17, 18, 24]), a larger time step can be used. On the other hand, the mesh movement can be determined by other considerations. For example, the meshes in the examples of Section 4 are moved via the MMPDE method using solution-Hessian based mesh adaptation. In this case, the mesh velocity does not necessarily minimize α_{CFL}. The condition (3.5) shows that, if $\alpha_{CFL}(\boldsymbol{x}_G^e)$ does not change significantly over the domain, Δt is determined by the minimum height $\sigma_{h,\min} = \min_{K,e} \sigma_e$ of the mesh elements.

Remark 3.2. From the above proof we can see that the choice of α in the CFL condition (3.2) can be different from that in the DG scheme (2.8) as long as

$$\alpha_{CFL}(\boldsymbol{x}) \geq \alpha_{LF}(\boldsymbol{x}). \tag{3.6}$$

For example, we can use $\alpha_{LF} = \alpha_p$ (pointwise, cf. (2.8)) for the scheme (denoted as $\alpha_{LF,p}$) and $\alpha_{CFL} = \alpha_h$ (global, cf. (2.10)) for the CFL condition (denoted as $\alpha_{CFL,h}$), which has been expressed for the general form (1.1) of conservation laws. This works since (3.6) is satisfied. On the other hand, the proof of Proposition 3.1 will not hold in general for the choice with $\alpha_{LF} = \alpha_h$ and $\alpha_{CFL} = \alpha_p$ since it violates (3.6). As a consequence, it is unclear if there is a theoretical guarantee that the scheme is L^1-stable for this choice.

Remark 3.3. For the choice $\alpha_{CFL} = \alpha_h$ (denoted as $\alpha_{CFL,h}$), the CFL condition (3.2) becomes

$$\Delta t \leq \frac{1}{\alpha_{CFL,h} \max\limits_{K} \left(\frac{1}{|K|} \sum\limits_{e \in \partial K} |e| \right)}. \tag{3.7}$$

This corresponds to the CFL condition (1.3) that has been commonly used in existing adaptive DG computation and is a direct extension of CFL conditions used for fixed, uniform meshes.

Remark 3.4. We can use something in between the very local α_p (2.8) and the global one in (2.10). For example, we take the maximum value of α over the Gauss points on edge e (cf. (2.9)). For this choice ($\alpha_{CFL} = \alpha_e$, denoted as $\alpha_{CFL,e}$), the CFL condition (3.2) becomes

$$\Delta t \leq \frac{1}{\max\limits_{K} \left(\frac{1}{|K|} \sum\limits_{e \in \partial K} |e| \alpha_{CFL,e} \right)}. \tag{3.8}$$

Remark 3.5. Since an explicit Runge-Kutta scheme can be expressed as a combination of the explicit Euler scheme with different time stepsize, we expect that the above analysis applies to explicit Runge-Kutta schemes as well. Moreover, for general P^k-DG ($k \geq 0$) and general systems of conservation laws, based on [6, 9], we suggest to use (1.5), where α_e is defined in (2.9). A choice of C_{cfl} is $C_{cfl} \leq 1/(2k+1)$ [9].

4. Numerical results

In this section we present numerical results obtained with the MMDG method described in the previous sections with the third-order explicit SSP Runge-Kutta scheme (SSP RK3) [14] for one- and two-dimensional Burgers' equations and Euler equations. The moving mesh is generated by the MMPDE moving mesh method; e.g., see [38, Section 4] or [39, Section 3] for a brief yet complete description of the method and [19–23] for a more detailed description and a development history. A key idea of the MMPDE method is to view any nonuniform mesh as a uniform one in some Riemannian metric specified by a tensor $\mathbb{M} = \mathbb{M}(\boldsymbol{x})$, a symmetric and uniformly positive definite matrix-valued function that provides the information needed for determining the size, shape, and orientation of the mesh elements throughout the domain.

In this work we use an optimal metric tensor based on the L^2-norm of piece linear interpolation error [22, 23]. To be specific, we consider a physical variable u and its finite element approximation u_h. Let H_K be a recovered Hessian of u_h on $K \in \mathcal{T}_h$ such as one obtained using least squares fitting. Assuming that the eigen-decomposition of H_K is given by

$$H_K = Q\text{diag}(\lambda_1, \cdots, \lambda_d)Q^T,$$

where Q is an orthogonal matrix, we define

$$|H_K| = Q\text{diag}(|\lambda_1|, \ldots, |\lambda_d|)Q^T.$$

The metric tensor is defined as

$$\mathbb{M}_K = \det \left(\beta_h \mathbb{I} + |H_K| \right)^{-\frac{1}{d+4}} \left(\beta_h \mathbb{I} + |H_K| \right), \quad \forall K \in \mathcal{T}_h, \tag{4.1}$$

where $\mathbb{I}$ is the identity matrix, $\det(\cdot)$ is the determinant of a matrix, and β_h is a regularization parameter defined through the algebraic equation

$$\sum_{K \in \mathcal{T}_h} |K| \det \left(\beta_h \mathbb{I} + |H_K| \right)^{\frac{2}{d+4}} = 2 \sum_{K \in \mathcal{T}_h} |K| \det \left(|H_K| \right)^{\frac{2}{d+4}}.$$

Roughly speaking, the choice of (4.1) is to concentrate mesh points in regions where the determinant of the Hessian is large.

In our numerical results, we use the physical solution u to compute the metric tensor for Burgers' equation and the density ρ and the entropy $\mathcal{S} = \ln(P\rho^{-\gamma})$ for the Euler equations, unless otherwise stated. To explain the latter, we first compute $\mathbb{M}_K^\rho$ and $\mathbb{M}_K^{\mathcal{S}}$ using (4.1) with $u = \rho$ and $\mathcal{S}$, respectively. Then, a new metric tensor is obtained through matrix intersection as

$$\tilde{\mathbb{M}}_K = \frac{\mathbb{M}_K^{\mathcal{S}}}{|||\mathbb{M}_K^{\mathcal{S}}|||} \cap \frac{\mathbb{M}_K^\rho}{|||\mathbb{M}_K^\rho|||}, \tag{4.2}$$

where $||| \cdot |||$ denotes the maximum absolute value of the entries of a matrix and "$\cap$" stands for matrix intersection. The reader is referred to [38] for the definition and geometric interpretation of matrix intersection.

The procedure of the MMDG method is presented in Algorithm 4.1.

Algorithm 4.1 The MMDG method for hyperbolic conservation laws

---

1: **Initialization.** For a given initial mesh $\mathcal{T}_h^0$, project the initial physical variables into the DG space $\mathcal{V}_h^{k,0}$ to obtain U_h^0.

2: **for** $n = 0, 1, \ldots,$ **do**

3:     **Mesh adaptation.**

4:     Compute the time step $\tilde{\Delta} t_n$ according to (1.5) based on $\mathcal{T}_h^n = \{\boldsymbol{x}_i^n\}$ and U_h^n, i.e.,

$$\tilde{\Delta} t_n = \frac{C_{cfl}}{\max\limits_{K^n} \left(\frac{1}{|K^n|} \sum\limits_{e \in \partial K^n} |e| \tilde{\alpha}_{CFL}^n \right)}, \tag{4.3}$$

    where $\tilde{\alpha}_{CFL}^n$ is chosen based on the eigenvalues of the Jacobian matrix of $\mathbf{F} \cdot \boldsymbol{n}$ (with respect to U) evaluated on edge e.

5:     Compute the metric tensor $\mathbb{M}$ based on $\mathcal{T}_h^n$ and U_h^n.

6:     Generate the new mesh $\tilde{\mathcal{T}}_h^{n+1} = \{\tilde{\boldsymbol{x}}_i^{n+1}\}$ using the MMPDE moving mesh method.

7:     Compute the nodal mesh velocity as

$$\dot{\boldsymbol{x}}_i^n = \frac{\tilde{\boldsymbol{x}}_i^{n+1} - \boldsymbol{x}_i^n}{\tilde{\Delta} t_n}, \quad i = 1, \ldots, N_v.$$

8:     Compute the time step Δt_n (using (1.5)) based on $\mathcal{T}_h^n$, $\tilde{\mathcal{T}}_h^{n+1}$ and U_h^n as

$$\Delta t_n = \frac{C_{cfl}}{\max\limits_{K} \left[\max \left(\frac{\sum\limits_{e \in \partial K^n} |e| \alpha_{CFL}^n}{|K^n|}, \frac{\sum\limits_{e \in \partial \tilde{K}^{n+1}} |e| \alpha_{CFL}^n}{|\tilde{K}^{n+1}|} \right) \right]}, \tag{4.4}$$

    where α_{CFL}^n is chosen based on the eigenvalues of the Jacobian matrix of $(\mathbf{F} - U\dot{\boldsymbol{X}}) \cdot \boldsymbol{n}$ (with respect to U) evaluated on edge e.

9:     Finally, the physical mesh $\mathcal{T}_h^{n+1}$ is defined as

$$\boldsymbol{x}_i^{n+1} = \boldsymbol{x}_i^n + \Delta t_n \, \dot{\boldsymbol{x}}_i^n, \quad i = 1, \dots, N_v.$$

10: **end for**
11: **Solution of the physical equations on the moving mesh.** Integrate the physical equations from t_n to t_{n+1} using the MMDG scheme to obtain U_h^{n+1}.

It is remarked that the CFL condition (4.4) has been used in the above algorithm to take the old and new meshes $\mathcal{T}_h^n$ and $\mathcal{T}_h^{n+1}$ into consideration. The rationale behind this is that the SSP RK3 scheme we use for the time integration has three stages that can be viewed roughly as the explicit Euler scheme from t_n to t_{n+1}, t_{n+1} to $t_{n+\frac{1}{2}}$, and $t_{n+\frac{1}{2}}$ to t_{n+1}, respectively. The computation of the right-hand side of (3.1) involves both $\mathcal{T}_h^n$ and $\mathcal{T}_h^{n+1}$ and thus it would be better to take the effects of these meshes into consideration more directly in time step even the mesh velocity is involved in the computation of α_e. In principle, we should also update α_e during the Runge-Kutta stages. However, this can cause changes in the time step-size during the Runge-Kutta stepping, which requires to re-start the stepping with a new time step-size. To avoid this complicity, we choose to freeze α_e at $t = t_n$.

We also note that α_e has been used in the algorithm and it can be replaced by α_h. Unless otherwise stated, α_e is chosen in the LF numerical flux. In fact, in the following numerical examples, we consider two options α_e and α_h for each of the LF numerical flux and the CFL condition. For example, $(\alpha_{CFL,h}, \alpha_{LF,e})$ indicates that α_h is used in the CFL condition and α_e used in the LF numerical flux. Furthermore, we consider moving mesh P^k-DG with $k = 1, 2, 3$. We take the CFL number C_{cfl} as 0.3 for P^1-DG, 0.15 for P^2-DG, and 0.1 for P^3-DG.

The CFL condition such as (4.4) provides a selection strategy for time step-size based on stability. This strategy is widely used in the DG computation of hyperbolic conservation laws. On the other hand, it is common practice to use an error-based time step-size selection strategy in the computation of ordinary differential equations (e.g., see Hairer and Wanner [16]). It is worth studying how error-based time step-size selection strategies fare in the DG computation of hyperbolic conservation laws. Interestingly, Conde *et al.* [13] developed a family of embedded pairs of SSP Runge-Kutta schemes and combined them with WENO5 (for spatial discretization on fixed meshes) for the numerical solution of the one-dimensional Euler equations. Particularly, they compared the performance of the third-order explicit SSP embedded RK (SSP-ERK(4,3)) pair with RK(3,2) of Bogacki and Shampine [4] and showed that both pairs lead to stable computation while SSP-ERK(4,3) offers stronger stability. In this section, we also present numerical results to demonstrate this idea of using error-based time step-size selection strategies for the computation of hyperbolic conservation laws. Our demonstration is more general than that of [13] in the sense that we consider one- and two-dimensional conservation laws and DG discretization on adaptive moving meshes. We use SSP-ERK(4,3) of [13] (see its Butcher tableau in Table 1) and the

Table 1: Butcher tableau for SSP ERK(4,3).

				0
$\frac{1}{2}$				$\frac{1}{2}$
$\frac{1}{2}$	$\frac{1}{2}$			1
$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$		$\frac{1}{2}$
$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{2}$	1
$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	1

standard PI controller for time step-size selection [16], with the relative and absolute tolerances chosen as 1×10^{-6} and 1×10^{-8}, respectively. This selection strategy is compared with that based on the CFL condition.

Example 4.1 (1D Burgers' equation)**.** We first consider Burgers' equation in one dimension,

$$u_t + \left(\frac{u^2}{2}\right)_x = 0, \quad x \in (0, 2) \tag{4.5}$$

subject to the initial condition $u(x,0) = \frac{1}{2} + \sin(\pi x)$ and periodic boundary conditions. The final time is $T = 1$.

The mesh trajectories, solution, and time step-size obtained with the moving mesh P^k-DG method ($k = 1, 2, 3$) and $N = 100$ are shown in Fig. 1. Three combinations of α_e and α_h are used: $(\alpha_{CFL,e}, \alpha_{LF,e})$, $(\alpha_{CFL,h}, \alpha_{LF,e})$, and $(\alpha_{CFL,h}, \alpha_{LF,h})$. They all lead to stable computation and almost identical solutions and mesh trajectories. A close examination on Δt indicates that $(\alpha_{CFL,e}, \alpha_{LF,e})$ gives a slightly larger and less oscillatory time step than those with $(\alpha_{CFL,h}, \alpha_{LF,e})$ and $(\alpha_{CFL,h}, \alpha_{LF,h})$. We may attribute the former to the fact $\alpha_e \leq \alpha_h$ (cf. (4.4)). However, it is unclear to the authors why $(\alpha_{CFL,h}, \alpha_{LF,e})$ and $(\alpha_{CFL,h}, \alpha_{LF,h})$, which appear to be more stable, lead to more oscillatory Δt. This feature is also observed in other examples presented in this section except Example 4.5. The figures also show that the time step behaves similar qualitatively and quantitatively for $(\alpha_{CFL,h}, \alpha_{LF,e})$ and $(\alpha_{CFL,h}, \alpha_{LF,h})$.

The mesh trajectories ($N = 100$) obtained with the P^1-DG method of Δt and a PI-controller time step-size selection strategy (denoted as ERK PI) are plotted in Fig. 2(a). The solution and time step-size are obtained with the P^1-DG method and ERK PI and $(\alpha_{CFL,e}, \alpha_{LF,e})$ step-size selection strategies are shown in Fig. 2(d) and Fig. 2(g), respectively. The results of P^2-DG and P^3-DG are also shown in Fig. 2. From Fig. 2, we see that both time-size selection strategies lead to stable computation and almost identical solutions and mesh trajectories. ERK PI produces slightly larger time step-size than $(\alpha_{CFL,e}, \alpha_{LF,e})$, and this is especially true for higher-order DG. To guarantee the
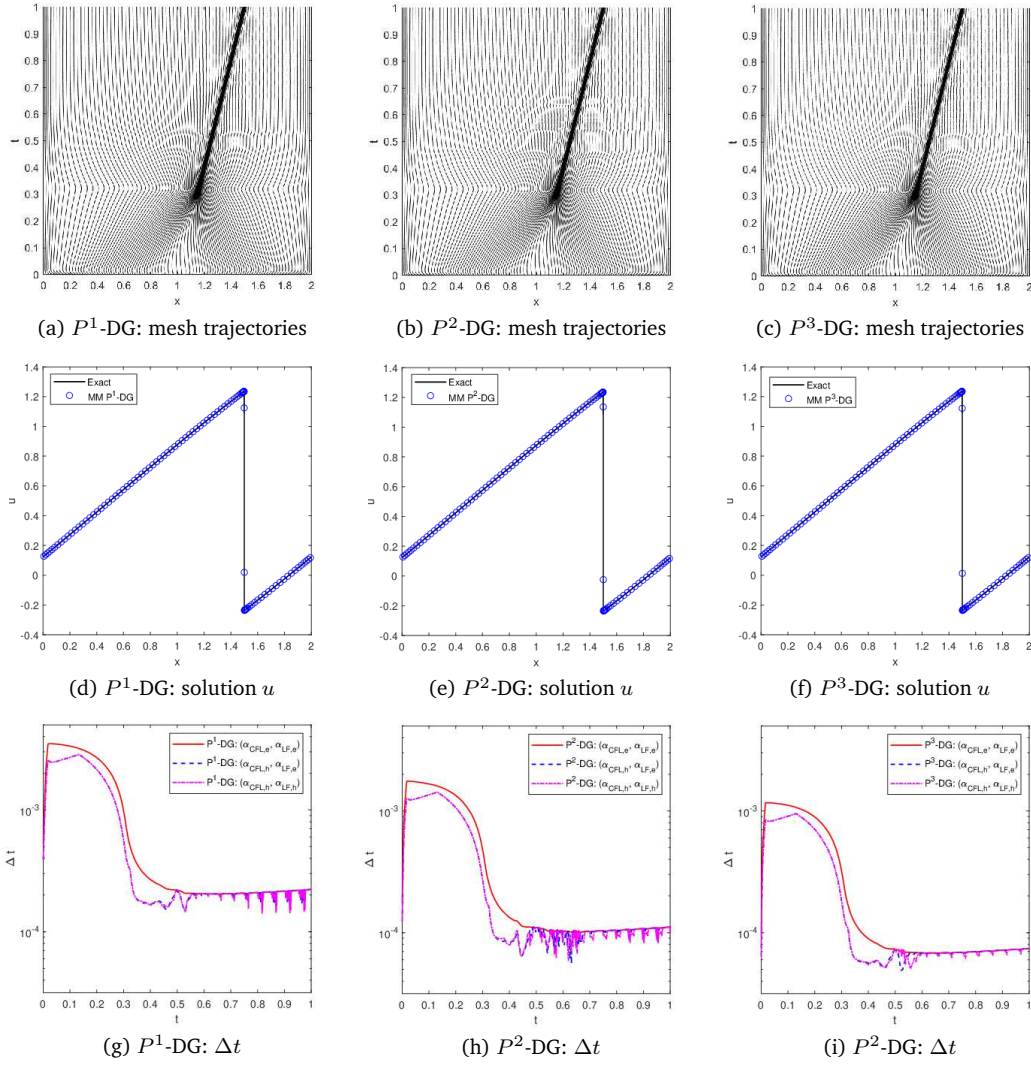
(a) P^1-DG: mesh trajectories   (b) P^2-DG: mesh trajectories   (c) P^3-DG: mesh trajectories

(d) P^1-DG: solution u (e) P^2-DG: solution u (f) P^3-DG: solution u

(g) P^1-DG: Δt (h) P^2-DG: Δt (i) P^2-DG: Δt

Figure 1: Example 4.1. The mesh trajectories, solution, and time step-size are obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 100$.

stability, the CFL number and thus Δt in the CFL condition decrease when k increases. This is reflected in Fig. 2. On the other hand, Δt selected by ERK PI does not seem to have big changes when k increases. As a result, the difference in Δt selected by ERK PI and CFL condition becomes bigger as k increases. Note that ERK PI leads to more oscillatory Δt, which is consistent with observations made in the computation of general stiff equations; e.g., see [16].

The time step-size obtained with $N = 50, 100, 200$ is shown in Fig. 3. The number of time steps obtained with $N = 50, 100, 200$ is shown in Fig. 4. The number of time steps appears to be a linear function of N for all cases. Interestingly, the slope of the straight lines for CFL-based selection is significantly larger than that associated with ERK PI,

(a) P^1-DG: mesh trajectories

(b) P^2-DG: mesh trajectories

(c) P^3-DG: mesh trajectories

(d) P^1-DG: solution u

(e) P^2-DG: solution u

(f) P^3-DG: solution u

(g) P^1-DG: Δt

(h) P^2-DG: Δt
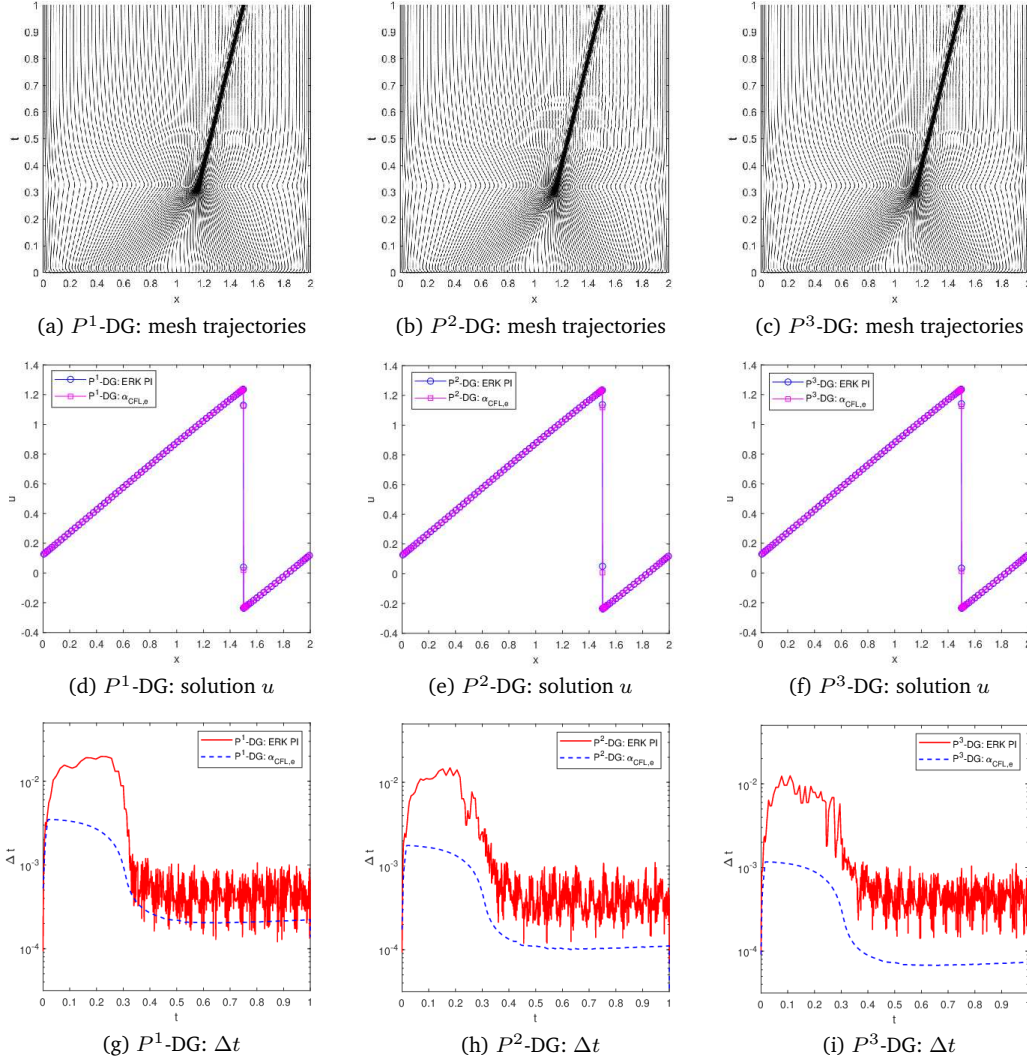
(i) P^3-DG: Δt

Figure 2: Example 4.1. The mesh trajectories, solution, and time step-size are obtained with the P^k-DG method ($k = 1, 2, 3$) and with ERK PI and CFL ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) step-size selection strategies ($N = 100$). The mesh trajectories and solutions are almost indistinguishable for both strategies.

and becomes larger as k increases. On the other hand, the lines associated with ERK PI for $k = 1, 2, 3$ differ only slightly.

Example 4.2 (Sod shock tube problem for 1D Euler equations). We consider the Sod problem of the Euler equations,

$$\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho u \\ \rho u^2 + P \\ u(E + P) \end{pmatrix} = 0, \quad x \in (-5, 5), \tag{4.6}$$

(a) P^1-DG: $(\alpha_{CFL,e}, \alpha_{LF,e})$

(b) P^2-DG: $(\alpha_{CFL,e}, \alpha_{LF,e})$

(c) P^3-DG: $(\alpha_{CFL,e}, \alpha_{LF,e})$

(d) P^1-DG: ERK PI

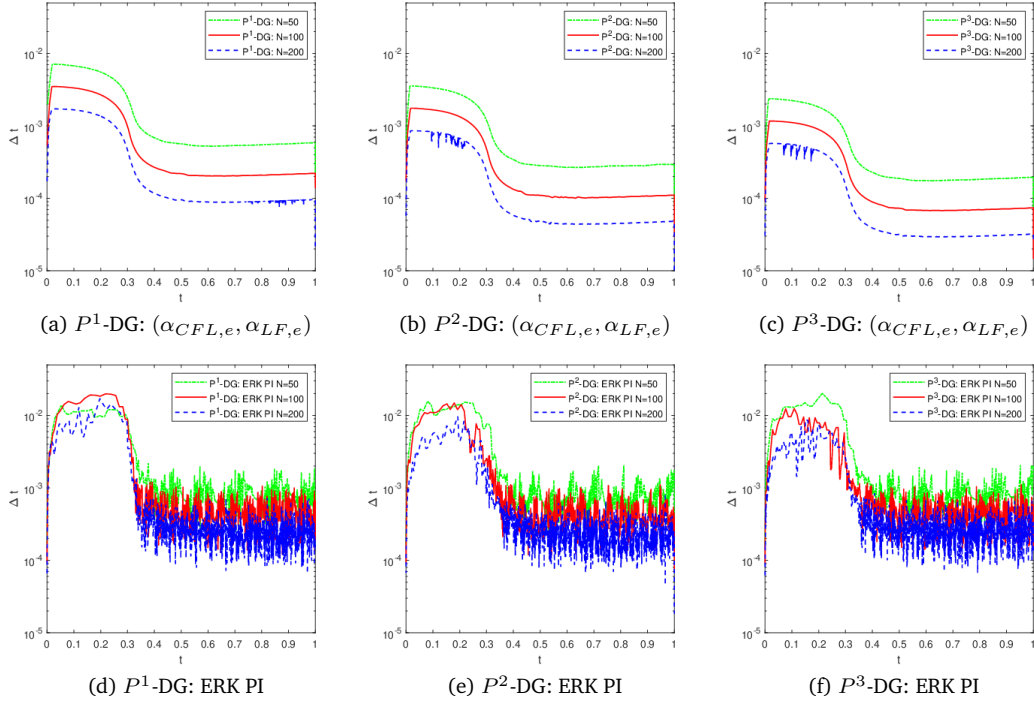(e) P^2-DG: ERK PI

(f) P^3-DG: ERK PI

Figure 3: Example 4.1. The time step-size is obtained with the P^k-DG method ($k = 1, 2, 3$) and the moving meshes of $N = 50, 100, 200$, and with ERK PI and CFL $(\alpha_{CFL,e}, \alpha_{LF,e})$ time step-size selection strategies.

where ρ is the density, u is the velocity, E is the energy density, and P is the pressure. The equation of the state is $E = P/(\gamma - 1) + \rho u^2/2$ with $\gamma = 1.4$. The initial conditions are given by

$$(\rho, u, P) = \begin{cases} (1, \ 0, \ 1), & x \leq 0, \\ (0.125, \ 0, \ 0.1), & x > 0. \end{cases} \tag{4.7}$$

The computation is stopped at $T = 2$. The solution of the problem contains a shock wave, a rarefaction, and a contact discontinuity.

The mesh trajectories, density, and time step-size obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 200$ are shown in Fig. 5. We can see that the mesh points are concentrated correctly around the shock wave, rarefaction, and contact discontinuity. The computation is stable for all three choices of α. Similar observations can be made for Δt as in the previous example except for the case with P^2-DG where $(\alpha_{CFL,h}, \alpha_{LF,e})$ leads to larger Δt for $0.2 < t < 0.8$ than $(\alpha_{CFL,e}, \alpha_{LF,e})$. To explain this, we notice that different CFL conditions can lead to different Δt, which can affect the mesh adaptation and the time integration of the physical equations. These effects do not seem significant in the previous example and in other cases in this example. However, they are slightly more significant in this case. From Fig. 6, we can see that $(\alpha_{CFL,h}, \alpha_{LF,e})$ results in a slightly larger minimum mesh spacing, which in turns gives slightly larger Δt.
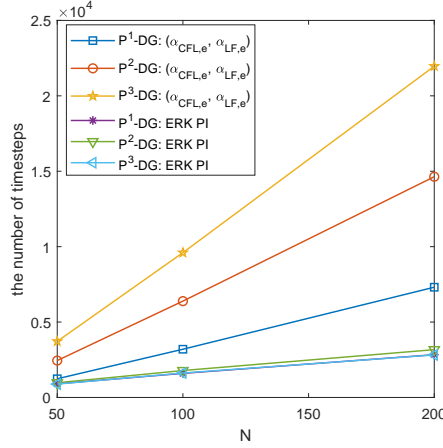
Figure 4: Example 4.1. The number of time steps is obtained with the P^k-DG method ($k = 1, 2, 3$) and the moving meshes of $N = 50, 100, 200$, and with ERK PI and CFL ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) time step-size selection strategies.

Once again, the results show that Δt associated with ($\alpha_{CFL,h}$, $\alpha_{LF,e}$) and ($\alpha_{CFL,h}$, $\alpha_{LF,h}$) has large oscillations than that associated with ($\alpha_{CFL,e}$, $\alpha_{LF,e}$).

It is worth mentioning that we have tried ($\alpha_{CFL,e}$, $\alpha_{LF,h}$), which violates (3.6), in the computation with moving mesh P^1-DG. The computation stops at around $t = 0.0011$ when Δt becomes machine-precision. Similar unstable computations have also been observed for this choice for other P^k elements.

The results obtained with ERK PI (SSP-ERK(4,3)) and CFL ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) time step-size selection strategies are shown in Fig. 7 for comparison purpose. As Example 4.1, we can see that both selection strategies lead to stable computation and almost identical solutions and mesh trajectories. Δt associated with ERK PI is slightly larger than that associated with CFL ($\alpha_{CFL,e}$, $\alpha_{LF,e}$), and this is especially true and the difference in Δt for both is larger for higher-order DG methods.

Example 4.3 (Lax problem for 1D Euler equations)**.** In this example, we consider the Lax problem of the Euler equations (4.6) with the following initial conditions:

$$(\rho, u, P) = \begin{cases} (0.445, \; 0.698, \; 3.528), & x \leq 0, \\ (0.5, \; 0, \; 0.571), & x > 0. \end{cases} \tag{4.8}$$

The final time is $T = 1.3$. The mesh trajectories, density, and time step-size Δt obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 200$ are shown in Fig. 8. In this example (and following examples), we show results only for two choices of α, ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) and ($\alpha_{CFL,h}$, $\alpha_{LF,e}$), since ($\alpha_{CFL,h}$, $\alpha_{LF,e}$) and ($\alpha_{CFL,h}$, $\alpha_{LF,h}$) produce almost identical results. The results show that the computation is stable and Δt associated with ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) is slightly larger and has small oscillations than that associated with ($\alpha_{CFL,h}$, $\alpha_{LF,e}$).

(a) P^1-DG: mesh trajectories   (b) P^2-DG: mesh trajectories   (c) P^3-DG: mesh trajectories

(d) P^1-DG: density ρ (e) P^2-DG: density ρ (f) P^3-DG: density ρ

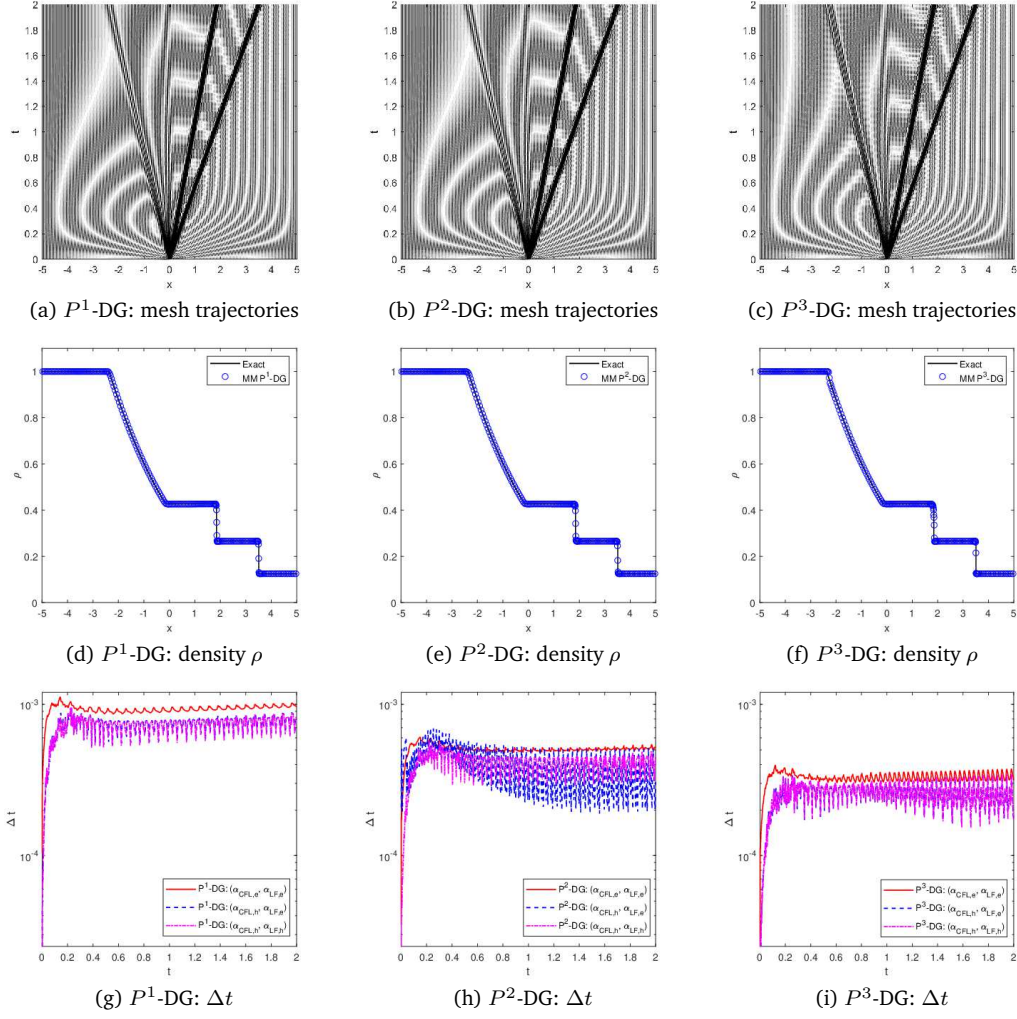(g) P^1-DG: Δt (h) P^2-DG: Δt (i) P^3-DG: Δt

Figure 5: Example 4.2. The mesh trajectories, solution, and time step-size are obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 200$.
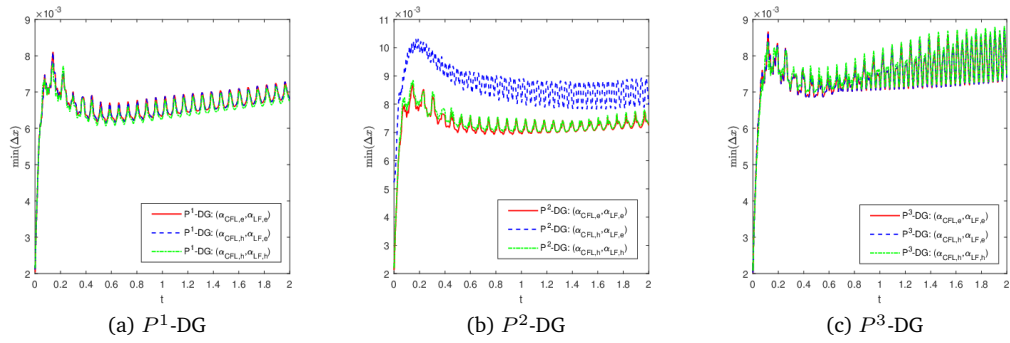
(a) P^1-DG   (b) P^2-DG   (c) P^3-DG

Figure 6: Example 4.2. The evolution of the minimal mesh spacing $\min(\Delta x)$ with the time obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 200$.

(a) P^1-DG: mesh trajectories   (b) P^2-DG: mesh trajectories   (c) P^3-DG: mesh trajectories

(d) P^1-DG: density ρ (e) P^2-DG: density ρ (f) P^3-DG: density ρ

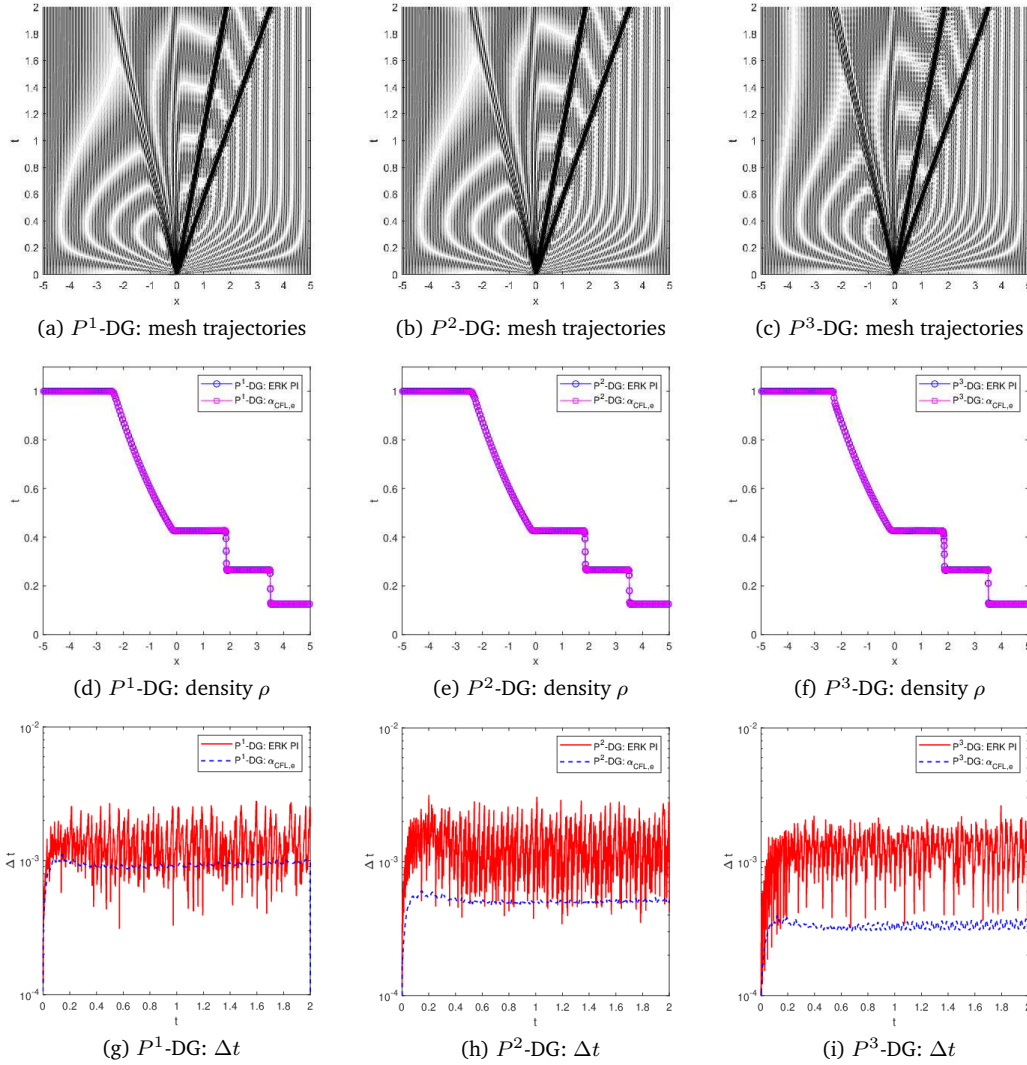(g) P^1-DG: Δt (h) P^2-DG: Δt (i) P^3-DG: Δt

Figure 7: Example 4.2. The mesh trajectories, solution, and time step-size are obtained with the P^k-DG method ($k = 1, 2, 3$) and with ERK PI and CFL ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) step-size selection strategies ($N = 200$). The mesh trajectories and solutions are almost indistinguishable for both strategies.

Example 4.4 (2D Burgers' equation)**.** We now consider Burgers' equation in two dimensions,

$$u_t + \left(\frac{u^2}{2}\right)_x + \left(\frac{u^2}{2}\right)_y = 0, \quad (x, y) \in (0, 2) \times (0, 2) \tag{4.9}$$

subject to the initial condition $u(x, y, 0) = e^{(-c(x^2+y^2))}$, $c = -\ln(10^{-16})$ and periodic boundary conditions. The final time is $T = 2$.

The mesh, solution, and time step-size (associated with three selection strategies, ERK PI, CFL ($\alpha_{CFL,e}, \alpha_{LF,e}$), and CFL ($\alpha_{CFL,h}, \alpha_{LF,e}$)) obtained with the moving mesh

(a) P^1-DG: mesh trajectories   (b) P^2-DG: mesh trajectories   (c) P^3-DG: mesh trajectories

(d) P^1-DG: density ρ (e) P^2-DG: density ρ (f) P^3-DG: density ρ

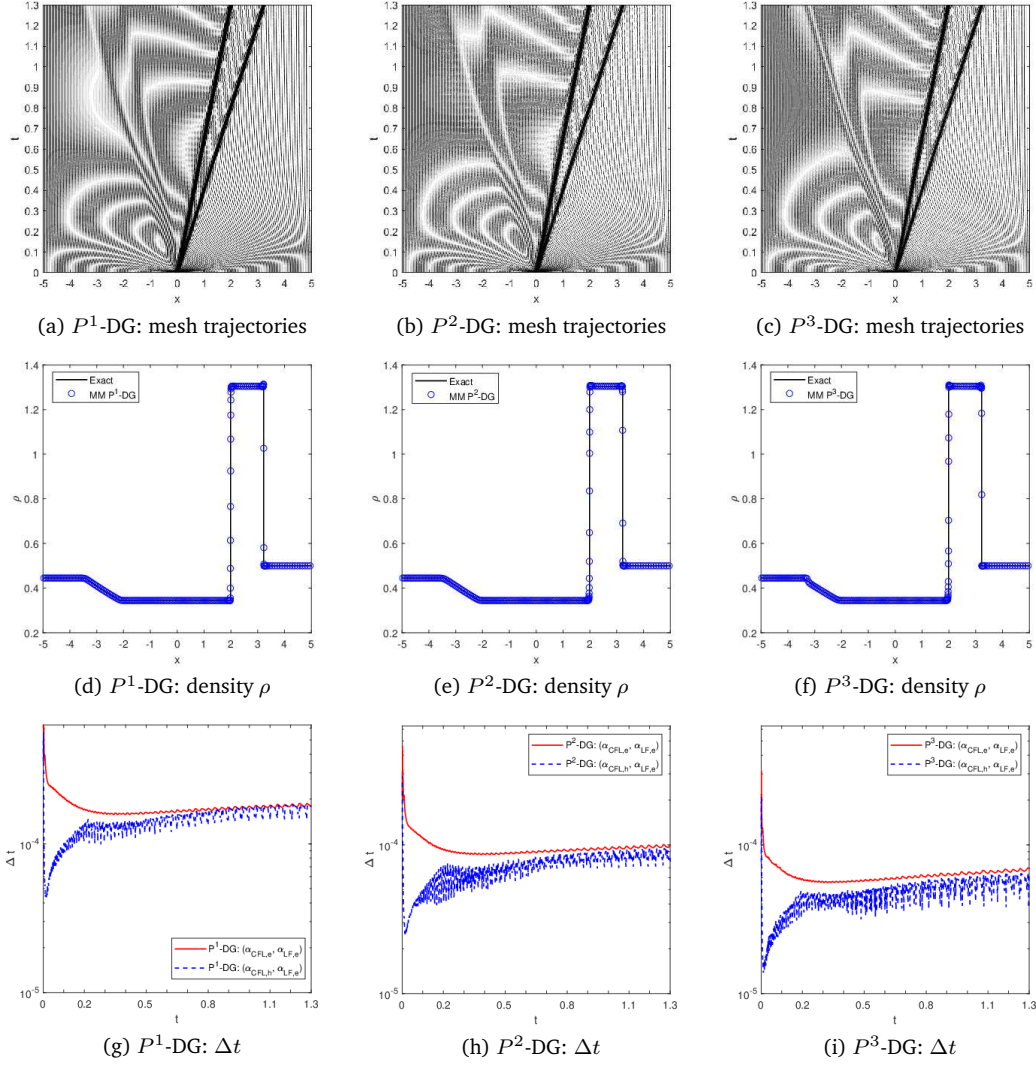(g) P^1-DG: Δt (h) P^2-DG: Δt (i) P^3-DG: Δt

Figure 8: Example 4.3. The mesh trajectories, solution, and time step-size are obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 200$.

P^k-DG method ($k = 1, 2, 3$) and $N = 30 \times 30 \times 4$ are plotted in Fig. 9. The results show that the computation is stable and the mesh points are concentrated in regions with sharp jumps in the solution. Moreover, Δt associated with ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) is larger and has smaller oscillations than that associated with ($\alpha_{CFL,h}$, $\alpha_{LF,e}$) while Δt associated with ERK PI is significantly larger than that associated with ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) (and this is especially true for higher-order DG). The time step-size for the meshes with $N = 10 \times 10 \times 4$, $20 \times 20 \times 4$, and $30 \times 30 \times 4$, is shown in Fig. 10. The number of time steps obtained with $N = 10 \times 10 \times 4$, $20 \times 20 \times 4$, and $30 \times 30 \times 4$, is shown in Fig. 11. The number of time steps for CFL-based time-size selection is significantly larger than that associated with ERK PI. Moreover, it increases more significantly as N increases. On the
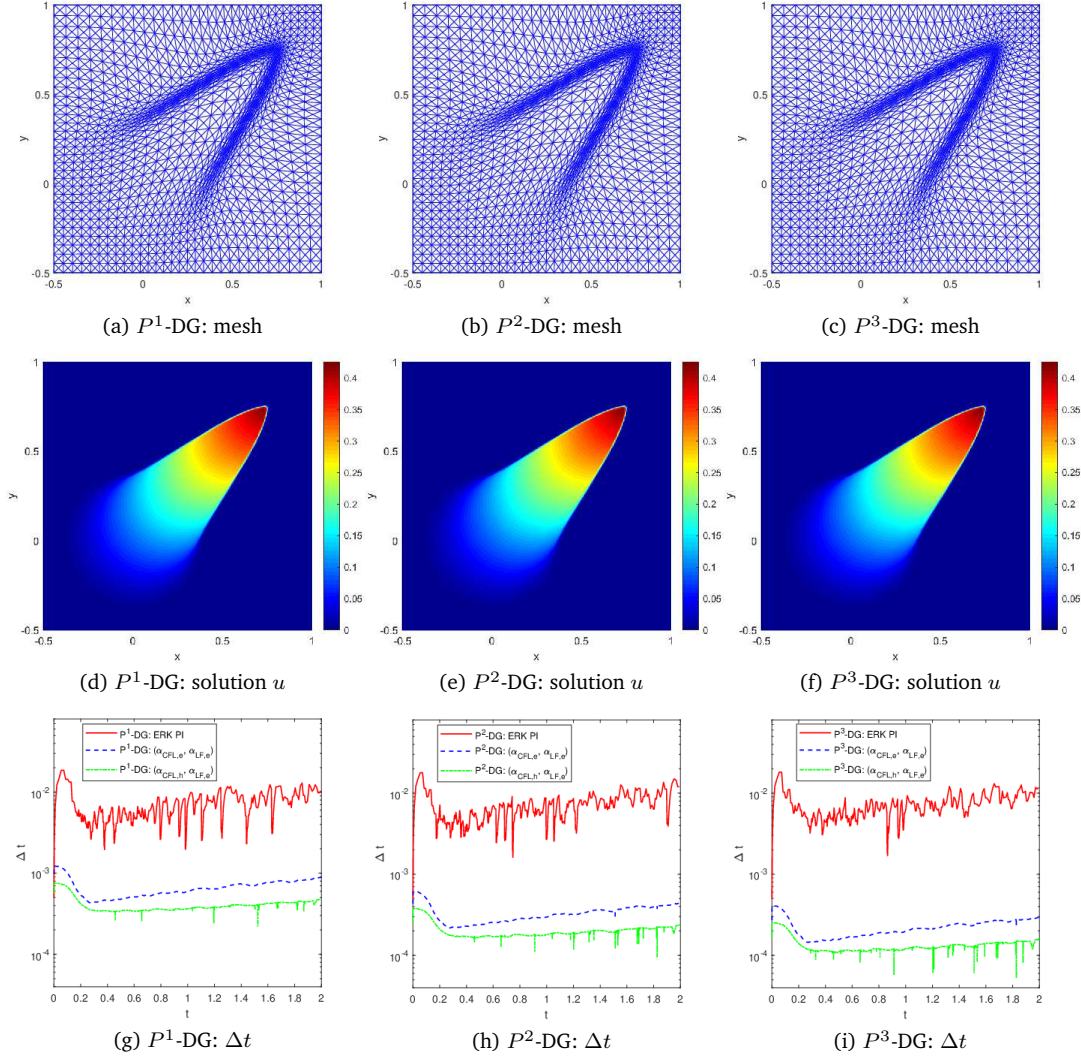
(a) P^1-DG: mesh   (b) P^2-DG: mesh   (c) P^3-DG: mesh

(d) P^1-DG: solution u (e) P^2-DG: solution u (f) P^3-DG: solution u

(g) P^1-DG: Δt (h) P^2-DG: Δt (i) P^3-DG: Δt

Figure 9: Example 4.4. The time step-size Δt results from three selection strategies, ERK PI, CFL $(\alpha_{CFL,h}, \alpha_{LF,e})$, and CFL $(\alpha_{CFL,e}, \alpha_{LF,e})$. The P^1, P^2, and P^3-DG method with a moving mesh of $N = 30 \times 30 \times 4$ is used.

other hand, the number of time steps associated with ERK PI increases almost linearly and more mildly as N increases and does not have much difference for $k = 1, 2, 3$.

Example 4.5 (Riemann problem for the 2D Euler equations)**.** We consider a two-dimensional Riemann problem of the Euler equations

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u(E + P) \end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ u(E + P) \end{pmatrix} = 0. \qquad (4.10)$$
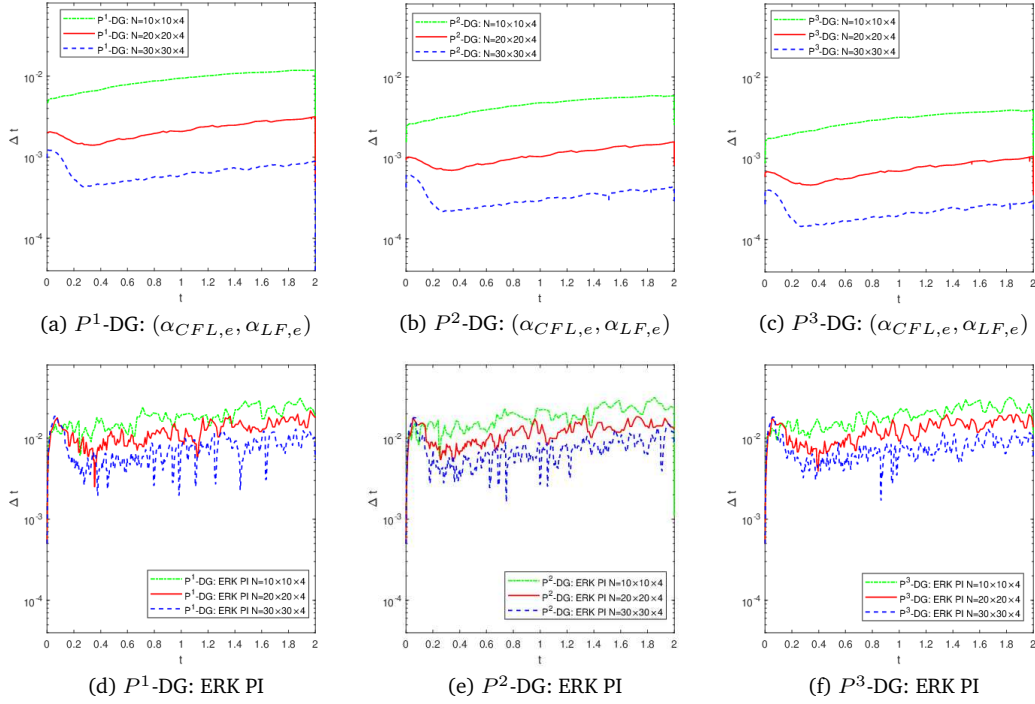
(a) P^1-DG: $(\alpha_{CFL,e}, \alpha_{LF,e})$

(b) P^2-DG: $(\alpha_{CFL,e}, \alpha_{LF,e})$

(c) P^3-DG: $(\alpha_{CFL,e}, \alpha_{LF,e})$

(d) P^1-DG: ERK PI

(e) P^2-DG: ERK PI

(f) P^3-DG: ERK PI

Figure 10: Example 4.4. The time step-size results from ERK PI and CFL $(\alpha_{CFL,e}, \alpha_{LF,e})$ strategies for the P^k-DG method ($k = 1, 2, 3$) and the moving meshes of $N = 10 \times 10 \times 4$, $20 \times 20 \times 4$, and $30 \times 30 \times 4$.

The computational domain is taken as $[0, 1] \times [0, 1]$, and the initial conditions are

$$(\rho, u, v, P)(x, y, 0) = \begin{cases} (1.1,\ 0,\ 0,\ 1.1), & x \geq 0.5, \quad y \geq 0.5, \\ (0.5065,\ 0.8939,\ 0,\ 0.35), & x < 0.5, \quad y \geq 0.5, \\ (1.1,\ 0.8939,\ 0.8939,\ 1.1), & x < 0.5, \quad y < 0.5, \\ (0.5065,\ 0,\ 0.8939,\ 0.35), & x \geq 0.5, \quad y < 0.5. \end{cases} \quad (4.11)$$

The energy density E and the pressure P are related by the equation of the state $E = P/(\gamma - 1) + \rho(u^2 + v^2)/2$ with $\gamma = 1.4$. The problem contains complicated interactions between four initial shocks. This problem has been widely used as a benchmark test for shock capturing methods due to the challenge in resolving the complicated flow features that emerge from those interactions. The final time for the computation is taken as $T = 0.25$.

The mesh and density at the final time, and time step-size obtained with the moving mesh P^k-DG method ($k = 1, 2, 3$) and $N = 50 \times 50 \times 4$ are shown in Fig. 12. Like the previous examples, the computation is stable and the mesh points are concentrated correctly around the shocks. Moreover, Δt associated with $(\alpha_{CFL,e}, \alpha_{LF,e})$ is slightly larger than that associated with $(\alpha_{CFL,h}, \alpha_{LF,e})$. However, unlike the previous examples, Δt has large oscillations before $t < 0.1$ for both cases.
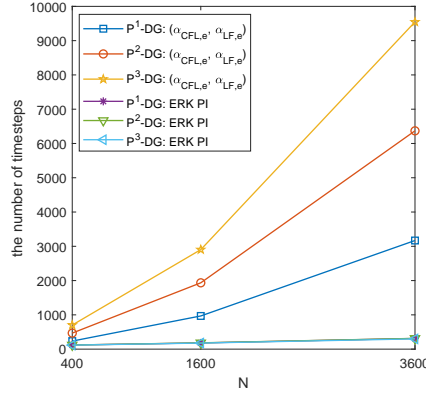
Figure 11: Example 4.4. The number of time step-sizes results from ERK PI and CFL $(\alpha_{CFL,e}, \alpha_{LF,e})$ strategies for the P^k-DG method $(k = 1, 2, 3)$ and the moving meshes of $N = 10 \times 10 \times 4$, $20 \times 20 \times 4$, and $30 \times 30 \times 4$.

Example 4.6 (Isentropic vortex convection problem for the 2D Euler equations)**.** We consider the isentropic vortex problem of Shu [33] for the two-dimensional compressible Euler equations (4.10). The mean flow is $\rho = P = u = v = 1$. We add to this mean flow an isentropic vortex perturbations centered at (x_0, y_0) in (u, v) and the temperature $(T = P/\rho)$ and no perturbation in the entropy $S = P\rho^{-\gamma}$, i.e.,

$$(\delta u, \delta v) = \frac{\varepsilon}{2\pi} e^{0.5(1-r^2)}(-\bar{y}, \bar{x}), \quad \delta T = -\frac{(\gamma - 1)\varepsilon^2}{8\gamma\pi^2} e^{1-r^2}, \tag{4.12}$$

where $(\bar{x}, \bar{y}) = (x - x_0, y - y_0)$, $r^2 = x^2 + y^2$, and the vortex strength $\varepsilon = 5$. This means that the initial conditions are

$$\rho(x, y, 0) = \left(1 - \frac{(\gamma - 1)\varepsilon^2}{8\gamma\pi^2} e^{1-r^2}\right)^{\frac{1}{\gamma-1}}, \quad P(x, y, 0) = \rho^\gamma(x, y, 0),$$

$$u(x, y, 0) = 1 - \frac{\varepsilon}{2\pi} e^{0.5(1-r^2)}\bar{y}, \qquad v(x, y, 0) = 1 + \frac{\varepsilon}{2\pi} e^{0.5(1-r^2)}\bar{x}.$$

The computational domain is taken as $(0, 10) \times (0, 10)$ and $(x_0, y_0) = (5, 5)$. The periodic boundary conditions are used for all unknown variables. The energy density E and the pressure P are related by the equation of the state $E = P/(\gamma - 1) + \rho(u^2 + v^2)/2$ with $\gamma = 1.4$. The final time for the computation is taken as $T = 1$.

   We use the density ρ only to compute the metric tensor in the MMPDE method since entropy $\mathcal{S} = \ln(P\rho^{-\gamma})$ is constant for this example. The mesh $(N = 50 \times 50 \times 4)$, the contours of density and momenta at the final time obtained with P^3-DG method and CFL $(\alpha_{CFL,h}, \alpha_{LF,e})$ step-size selection strategy are shown in Fig. 13. For comparison purpose, we plot Δt for three time step-size strategies, ERK PI, CFL $(\alpha_{CFL,e}, \alpha_{LF,e})$, and CFL $(\alpha_{CFL,h}, \alpha_{LF,e})$ in Fig. 14. Like previous examples, the computation is stable. Moreover, Δt associated with $(\alpha_{CFL,e}, \alpha_{LF,e})$ is slightly larger than that associated with $(\alpha_{CFL,h}, \alpha_{LF,e})$, and Δt associated with ERK PI is significantly larger than that associated with $(\alpha_{CFL,e}, \alpha_{LF,e})$.

(a) P^1-DG: mesh   (b) P^2-DG: mesh   (c) P^3-DG: mesh

(d) P^1-DG: density ρ (e) P^2-DG: density ρ (f) P^3-DG: density ρ

(g) P^1-DG: Δt (h) P^2-DG: Δt (i) P^3-DG: Δt
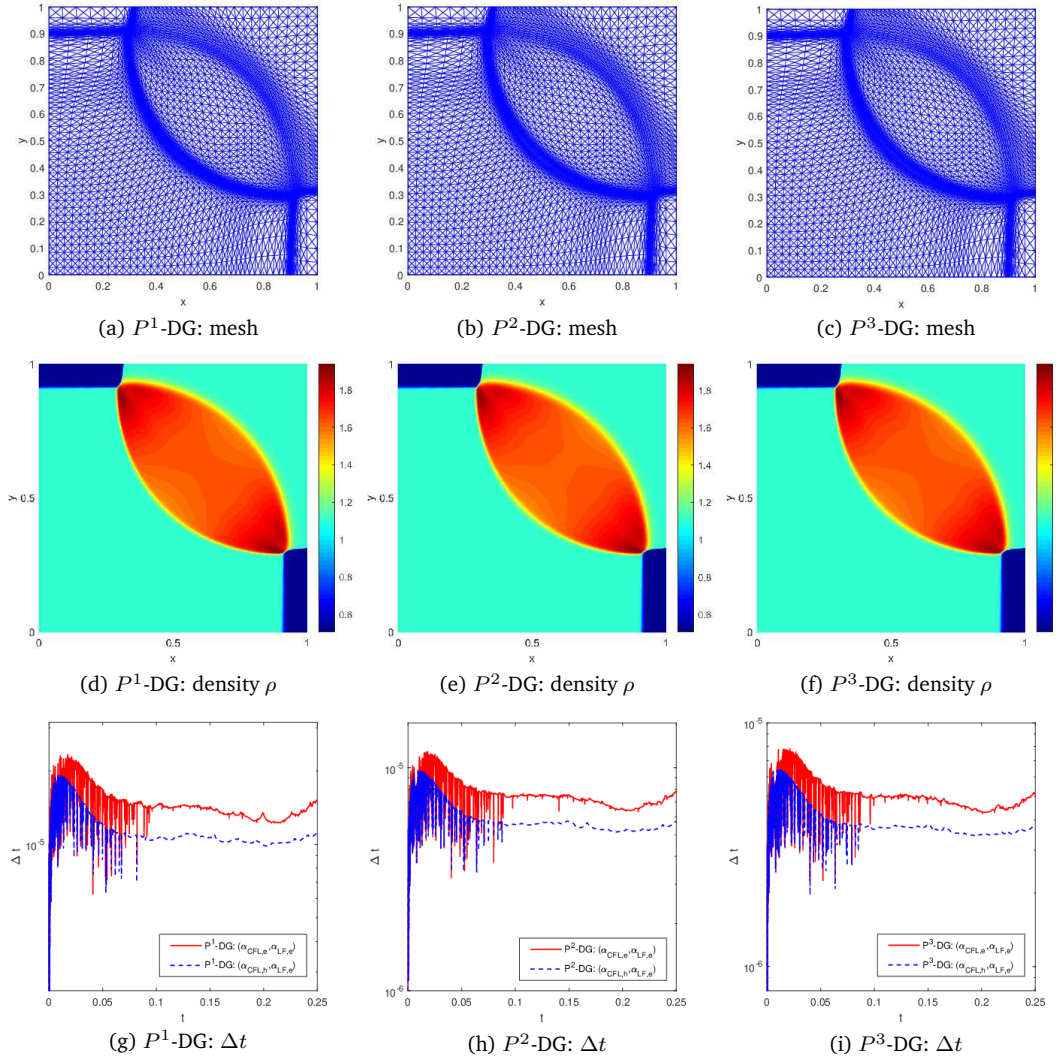
Figure 12: Example 4.5. The mesh, solution, and time step-size are obtained with the P^k-DG method ($k = 1, 2, 3$) and a moving mesh of $N = 50 \times 50 \times 4$.

5. Conclusions

In the previous sections we have studied the stability of a DG solution of conservation laws on adaptive moving meshes. Particularly, we have obtained a CFL condition (3.2) for moving mesh P^0-DG for linear scalar conservation laws. The condition shows that the allowed maximum time step depends on the coupling between α on any edge and the element height associated with the edge, where α (cf. 2.8) is the maximum absolute value of the eigenvalue of the Jacobian matrix of the flux $(\boldsymbol{F} - U\dot{\boldsymbol{X}}) \cdot \boldsymbol{n}$ with respect to U and the element height is the distance between the edge and the vertex opposite to the edge. The condition justifies a known intuition that time step can be
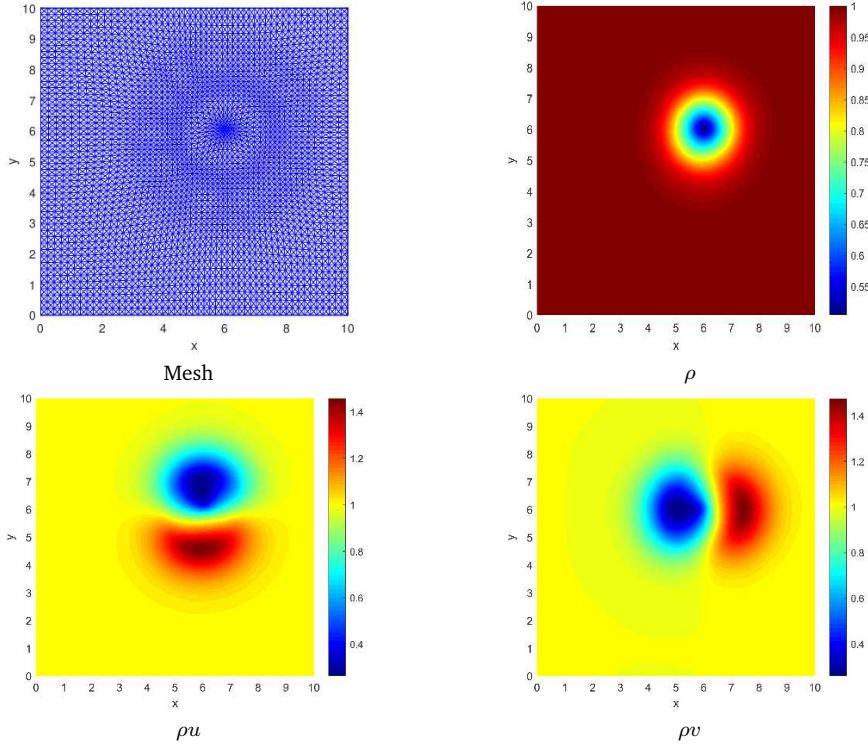
<center>Mesh</center>

<center>ρ</center>

<center>ρu</center>

<center>ρv</center>

Figure 13: Example 4.6. The mesh ($N = 50 \times 50 \times 4$) and contours of the solutions at $t = 1$ are obtained with the moving mesh P^3-DG method and CFL ($\alpha_{CFL,e}$, $\alpha_{LF,e}$) time step-size selection strategy.
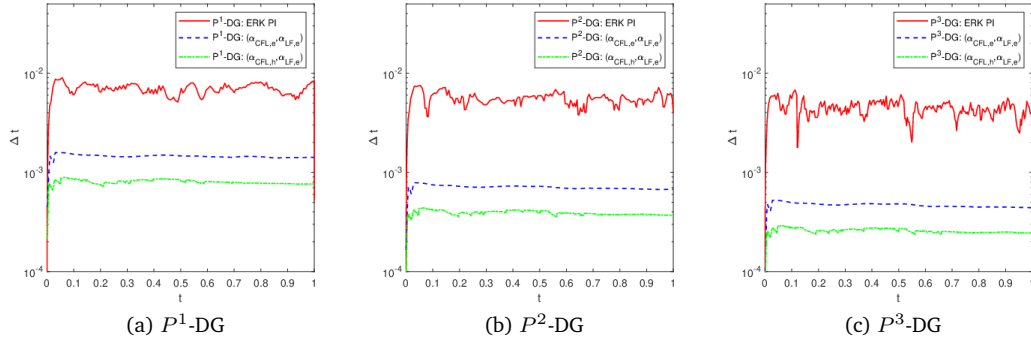

<center>(a) P^1-DG</center>

<center>(b) P^2-DG</center>

<center>(c) P^3-DG</center>

Figure 14: Example 4.6. The time step-size Δt results from with three selection strategies, ERK PI, CFL $\left(\alpha_{CFL,e},\ \alpha_{LF,e}\right)$, and CFL $\left(\alpha_{CFL,h},\ \alpha_{LF,e}\right)$. The P^1, P^2, and P^3-DG method with a moving mesh of $N = 50 \times 50 \times 4$ is used.

increased if the mesh velocity is chosen to minimize $(\boldsymbol{F} - U\dot{\boldsymbol{X}})$. On the other hand, if α does not change significantly over the domain, it is expected that the allowed maximum time step is determined mostly by the minimum element height of the mesh.

The stability analysis also shows that different choices of α can be used in the LF numerical flux and the CFL condition but a relation (3.6) should be satisfied for L^1 stability. Two common choices for α are α_e defined in (2.9) and α_h defined in (2.10).

Based on (3.2) and CFL conditions for DG method on fixed meshes, we have proposed to choose the time step according to (4.4) (with the same or a different choice of α) for moving mesh P^k-DG ($k = 1, 2, 3$) for general conservation laws. This condition reduces to the one on fixed meshes when α is taken as α_h.

Numerical examples have been presented with mesh adaptation by the MMPDE method for Burgers' and Euler equations in one and two dimensions. Numerical results show that (4.4) with $\alpha_{CFL} = \alpha_e$ or $\alpha_{CFL} = \alpha_h$ lead to stable computation. Moreover, $(\alpha_{CFL,e}, \alpha_{LF,e})$ typically gives larger Δt than $(\alpha_{CFL,h}, \alpha_{LF,e})$ and $(\alpha_{CFL,h}, \alpha_{LF,h})$. All but one example also show that Δt with $(\alpha_{CFL,e}, \alpha_{LF,e})$ has smaller oscillations.

In this work, we have also studied an error-based time step-size selection strategy with the explicit SSP embedded Runge-Kutta pairs for DG computation of hyperbolic conservation laws on adaptive moving meshes. Numerical examples show that the error-based strategy can lead to stable computation and result in larger time step-size especially for higher-order DG than the CFL based selection strategy. It should be pointed out that we have not considered moving mesh strategies such as Lagrangian-type methods to minimize α and increase Δt in the current work. Moreover, the error-based time step-size selection strategy seems to result in larger time step-size than CFL condition. These are interesting topics that may deserve more studies in near future.

Acknowledgments

References

[1] M. J. BAINES, *Moving Finite Elements*, Oxford University Press, 1994.

[2] M. J. BAINES, M. E. HUBBARD, AND P. K. JIMACK, *Velocity-based moving mesh methods for nonlinear partial differential equations*, Comm. Comput. Phys. 10 (2011), 509–576.

[3] K. BEY AND J. ODEN, hp-*version discontinuous Galerkin methods for hyperbolic conservation laws*, Comput. Meth. Appl. Mech. Engrg. 133 (1996), 259–286.

[4] P. BOGACKI AND L. F. SHAMPINE, *A 3(2) pair of Runge-Kutta formulas*, Appl. Math. Lett. 2 (1989), 321–325.

[5] C. J. BUDD, W. HUANG, AND R. D. RUSSELL, *Adaptivity with moving grids*, Acta Numer. 18 (2009), 111–241.

[6] G. CHAVENTG AND B. COCKBURN, *The local projection P^0-P^1-discontinuous-Galerkin finite element method for scalar conservation laws*, ESAIM: M2AN 23 (1989), 565–592.

[7] B. COCKBURN, S. HOU, AND C.-W. SHU, *The Runge-Kutta local projection discontinuous Galerkin method for conservation laws IV: Multidimensional systems*, Math. Comp. 54 (1990), 545–581.

[8] B. COCKBURN, S.-Y. LIN, AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One dimensional systems*, J. Comput. Phys. 84 (1989), 90–113.

[9] B. COCKBURN AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework*, Math. Comp. 52 (1989), 411–435.

[10] B. COCKBURN AND C.-W. SHU, *The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws*, ESAIM: M2AN 25 (1991), 337–361.

[11] B. COCKBURN AND C.-W. SHU, *The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems*, J. Comput. Phys. 141 (1998), 199–224.

[12] B. COCKBURN AND C.-W. SHU, *Runge-Kutta discontinuous Galerkin methods for convection dominated problems*, J. Sci. Comput. 16 (2001), 173–261.

[13] S. CONDE, I. FEKETE, AND J. N. SHADID, *Embedded error estimation and adaptive step-size control for optimal explicit strong stability preserving Runge-Kutta methods*, arxiv: 1806.08693.

[14] S. GOTTLIEB, C.-W. SHU, AND E. TADMOR, *Strong stability-preserving high-order time discretization methods*, SIAM Review 43 (2001), 89–112.

[15] M. DUMBSER AND W. BOSCHERI, *High-order unstructured Lagrangian one-step WENO finite volume schemes for non-conservative hyperbolic systems: Applications to compressible multi-phase flows*, Comput. & Fluids 86 (2013), 405–432.

[16] E. HAIRER AND G. WANNER, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, Springer-Verlag, 1991.

[17] C. W. HIRT, *An arbitrary Lagrangian-Eulerian computing technique*, in: *Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics*, Vol. 8, 1971.

[18] C. W. HIRT, A. A. AMSDEN, AND J. COOK, *An arbitrary Lagrangian-Eulerian computing method for all flow speeds*, J. Comput. Phys. 14 (1974), 227–253.

[19] W. HUANG AND L. KAMENSKI, *A geometric discretization and a simple implementation for variational mesh generation and adaptation*, J. Comput. Phys. 301 (2015), 322–337.

[20] W. HUANG AND L. KAMENSKI, *On the mesh nonsingularity of the moving mesh PDE method*, Math. Comp. 87 (2018), 1887–1911.

[21] W. HUANG, Y. REN, AND R. RUSSELL, *Moving mesh partial differential equations (MM-PDEs) based upon the equidistribution principle*, SIAM J. Numer. Anal. 31 (1994), 709–730.

[22] W. HUANG AND R. RUSSELL, *Adaptive Moving Mesh Methods*, Applied Mathematical Sciences Series, Vol. 174, Springer, 2011.

[23] W. HUANG AND W. SUN, *Variational mesh adaptation II: Error estimates and monitor functions*, J. Comput. Phys. 184 (2003), 619–648.

[24] M. KUCHARIK AND M. SHASHKOV, *Conservative multi-material remap for staggered multi-material arbitrary Lagrangian-Eulerian methods*, J. Comput. Phys. 258 (2014), 268–304.

[25] P. LESAINT AND P.-A. RAVIART, *On a finite element method for solving the neutron transport equation*, in: *Mathematical Aspects of Finite Elements in Partial Differential Equations, C. de Boor (Ed.)*, Academic Press, (1974), 89–123.

[26] R. LI AND T. TANG, *Moving mesh discontinuous Galerkin method for hyperbolic conservation laws*, J. Sci. Comput. 27 (2006), 347–363.

[27] D. LUO, W. HUANG, AND J. QIU, *A quasi-Lagrange moving mesh discontinuous Galerkin method for hyperbolic conservation laws*, J. Comput. Phys. 396 (2019), 544–578.

[28] D. LUO, S. LI, W. HUANG, J. QIU, AND Y. CHEN, *A quasi-conservative DG-ALE method for multi-component flows using the non-oscillatory kinetic flux*, J. Sci. Comput. 90 (2022), 46.

[29] J. MACKENZIE AND A. NICOLA, *A discontinuous Galerkin moving mesh method for Hamilton-Jacobi equations*, SIAM J. Sci. Comput. 29 (2007), 2258–2282.

[30] N. R. MORGAN, K. N. LIPNIKOV, D. E. BURTON, AND M. A. KENAMOND, *A Lagrangian staggered grid Godunov-like approach for hydrodynamics*, J. Comput. Phys. 259 (2014), 568–597.

[31] W. H. REED AND T. R. HILL, *Triangular Mesh Methods for Neutron Transport Equation*, Los Alamos Scientific Laboratory Report LA-UR-73-479, 1973.

[32] C.-W. SHU, *Total-variation-diminishing time discretizations*, SIAM J. Sci. Stat. Comput. 9 (1988), 1073–1084.

[33] C.-W. SHU, *Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws*, in: *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, Lecture Notes in Mathematics, Vol. 1697, Springer, (1998), 325–432.

[34] T. TANG, *Moving mesh methods for computational fluid dynamics flow and transport*, Recent Advances in Adaptive Computation (Hangzhou, 2004), AMS Contemporary Mathematics, Vol. 383, AMS, (2005), 141–173.

[35] M. UZUNCA, B. KARASÖZEN, AND T. KÜÇÜKSEYHAN, *Moving mesh discontinuous Galerkin methods for PDEs with traveling wave*, Appl. Math. Comput. 292 (2017), 9–18.

[36] F. VILAR, P.-H. MAIRE, AND R. ABGRALL, *A discontinuous Galerkin discretization for solving the two-dimensional gas dynamics equations written under total Lagrangian formulation on general unstructured grids*, J. Comput. Phys. 276 (2014), 188–234.

[37] C. WANG, H. LUO, AND M. SHASHKOV, *A reconstructed discontinuous Galerkin method for compressible flows in Lagrangian formulation*, Comput. & Fluids 202 (2020), 104522.

[38] M. ZHANG, J. CHENG, W. HUANG, AND J. QIU, *An adaptive moving mesh discontinuous Galerkin method for the radiative transfer equation*, Commun. Comput. Phys. 27 (2020), 1140–1173.

[39] M. ZHANG, W. HUANG, AND J. QIU, *High-order conservative positivity-preserving DG-interpolation for deforming meshes and application to moving mesh DG simulation of radiative transfer*, SIAM J. Sci. Comput. 42 (2020), A3109–A3135.

[40] M. ZHANG, W. HUANG, AND J. QIU, *A high-order well-balanced positivity-preserving moving mesh DG method for the shallow water equations with non-flat bottom topography*, J. Sci. Comput. 87 (2021), No. 88.

[41] M. ZHANG, W. HUANG, AND J. QIU, *A well-balanced positivity-preserving quasi-Lagrange moving mesh DG method for the shallow water equations*, Commun. Comput. Phys. 31 (2022), 94–130.

[42] X. ZHANG, Y. XIA, AND C.-W. SHU, *Maximum-principle-satisfying and positivity-preserving high order discontinuous Galerkin schemes for conservation laws on triangular meshes*, J. Sci. Comput. 50 (2012), 29–62.

[43] W. ZHANG, Y. XIA, AND Y. XU, *Positivity-preserving well-balanced arbitrary Lagrangian-Eulerian discontinuous Galerkin methods for the shallow water equations*, J. Sci. Comput. 88 (2021), No. 57.