Computers and Fluids 139 (2016) 68–79

Contents lists available at ScienceDirect

Computers and Fluids

journal homepage: www.elsevier.com/locate/compfluid

A hybrid reconstructed discontinuous Galerkin method for compressible flows on arbitrary grids

Jian Cheng[a], Tiegang Liu[a], Hong Luo[b,*]

[a] School of Mathematics and Systems Science, Beihang University, Beijing 100191, P.R.China
[b] Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, NC 27695, USA

ARTICLE INFO

Article history:
Received 17 December 2015
Revised 30 March 2016
Accepted 1 April 2016
Available online 7 April 2016

Keywords:
Reconstructed methods
Discontinuous Galerkin methods
Arbitrary grids

ABSTRACT

A new reconstructed Discontinuous Galerkin (rDG) method based on a hybrid least-squares recovery and reconstruction, named P1P2(HLSr), is developed for solving the compressible Euler and Navier-Stokes equations on arbitrary grids. The development of the new hybrid rDG method is motivated by the observation that the original least-squares reconstruction does not have the property of the 2-exactness. As a remedy, the new hybrid reconstruction obtains a quadratic polynomial solution from the underlying linear DG solution by use of a hybrid recovery and reconstruction strategy. The resultant hybrid rDG method combines the simplicity of the reconstruction-based DG method and the accuracy of the recovery-based DG method, and has the desired property of 2-exactness. A number of test cases for a variety of flow problems are presented to assess the performance of the new P1P2(HLSr) method. Numerical experiments demonstrate that this hybrid rDG method is able to achieve the designed optimal 3rd order of accuracy for both inviscid and viscous flows and outperform the rDG methods based on either Green-Gauss or least-squares reconstruction.

1. Introduction

The discontinuous Galerkin (DG) methods [1–13],[14–24] have recently become popular for the solution of systems of conservation laws. Nowadays, they are widely used in computational fluid dynamics, computational acoustics, and computational electromagnetics. The DG methods combine two advantageous features commonly associated to finite element (FE) and finite volume (FV) methods. As in classical FE methods, accuracy is obtained by means of high-order polynomial approximation within an element rather than by wide stencils as in the case of FV methods. The physics of wave propagation is, however, accounted for by solving the Riemann problems that arise from the discontinuous representation of the solution at element interfaces. In this respect, the methods are therefore similar to FV methods. The DG methods have many attractive features: 1) They have several useful mathematical properties with respect to conservation, stability, and convergence; 2) The methods can be easily extended to higher-order approximation; 3) They are well suited for complex geometries since they can be applied on unstructured grids. In addition, the methods can also handle non-conforming elements, where the grids are allowed to have hanging nodes; 4) The methods are highly parallelizable, as they are compact and each element is independent. Since the elements are discontinuous, and the inter-element communications are minimal, domain decomposition can be efficiently employed; 5) They can easily handle adaptive strategies, since refining or coarsening a grid can be achieved without considering the continuity restriction commonly associated with the conforming elements. The methods allow easy implementation of *hp*-refinement, for example, the order of accuracy, or shape, can vary from element to element; 6) They have the ability to compute low Mach number flow problems without recourse to the time-preconditioning techniques normally required for the FV methods.

However, the DG methods have their own weaknesses. Indeed, as compared with the FE and FV methods, the DG methods require solutions of systems of equations with more unknowns for the same grids. Consequently, the DG method are recognized as expensive in terms of both computational costs and storage requirements. In order to reduce high costs associated with the DG methods, Dumbser et al. [18,19] have introduced a new family of reconstructed DG (rDG) methods, termed P_nP_m schemes, where P_n indicates that a piecewise polynomial of degree of n is used to approximate a DG solution, and P_m represents a reconstructed polynomial solution of degree of m ($m \geq n$) that is used to compute the fluxes. The P_nP_m schemes are designed to enhance the accuracy of

* Corresponding author. Tel.: +1 919 513 3898.
E-mail address: hong_luo@ncsu.edu (H. Luo).

http://dx.doi.org/10.1016/j.compfluid.2016.04.001
0045-7930/© 2016 Elsevier Ltd. All rights reserved.

the DG methods by increasing the order of the underlying polynomial solution. The beauty of P_nP_m schemes is that they provide a unified formulation for both FV and DG methods, and contain both classical FV and standard DG methods as two special cases of P_nP_m schemes, and thus allow for a direct efficiency comparison. When $n = 0$, i.e. a piecewise constant polynomial is used to represent a numerical solution, P_0P_m is nothing but classical high-order FV schemes, where a polynomial solution of degree m ($m \geq 1$) is reconstructed from a piecewise constant solution. When $m = n$, the reconstruction reduces to the identity operator, and P_nP_n scheme yields a standard DG method.

Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the P_nP_m schemes. In Dumbser's work, this is achieved using a so-called in-cell recovery similar to the inter-cell recovery originally proposed by van Leer et al. [25–27] where recovered equations are obtained using a L_2 projection, i.e., the recovered polynomial solution is uniquely determined by making it indistinguishable from the underlying DG solutions in the contributing cells in the weak sense. The resultant over-determined system is then solved using a least-squares method that guarantees exact conservation, not only of the cell averages but also of all higher-order moments in the reconstructed cell itself, such as slopes and curvatures. However, this conservative least-squares recovery approach is computationally expensive, as it involves both recovery of a polynomial solution of higher-order and least-squares solution of the resulting over-determined system. Furthermore, the recovery might be problematic for a boundary cell, where the number of the face-neighboring cells might be not enough to provide the necessary information to recover a polynomial solution of a desired order.

Fortunately, recovery is not the only way to obtain a polynomial solution of higher-order from the underlying discontinuous Galerkin solutions. Rather, reconstruction widely used in the FV methods provides an alternative, probably a better choice to obtain a higher-order polynomial representation. Luo et al. [28–30] developed a reconstructed DG method using a Taylor basis for the solution of the compressible Euler and Navier-Stokes equations on arbitrary grids, where a higher-order polynomial solution is reconstructed by use of a strong interpolation, requiring point values and derivatives to be interpolated on the face-neighboring cells. The resulting over-determined linear system of equations is then solved in the least-squares sense. This reconstruction scheme only involves the von Neumann neighborhood, and thus is compact, simple, robust, and flexible. Furthermore, the reconstruction scheme guarantees exact conservation, not only of the cell averages but also of their slopes due to a judicious choice of our Taylor basis. However, the least-squares reconstruction does not have the property of k-exactness.

More recently, Zhang et al. [31–33] presented a class of hybrid DG/FV methods for the conservation laws where the second derivatives in a cell are obtained from the first derivatives in the cell itself and its neighboring cells using a Green-Gauss reconstruction widely used in the FV methods. This provides a fast, simple, and robust way to obtain higher-order polynomial solutions. However, this reconstruction is less accurate, since it only uses the information on first derivatives. Furthermore, the least-squares reconstruction does not have the property of k-exactness. Luo et al. [35] have conducted a comparative study for these three rDG methods for solving the compressible Euler equations on arbitrary grids. The numerical experiments indicates that all three rDG P1P2 methods are able to achieve a third-order of accuracy: one order accuracy higher than the underlying second-order DG method and significantly improve the accuracy of the underlying second-order DG methods, although the P1P2 method based on the reconstruction provides the best performance in terms of both accuracy and robustness.

The objective of the effort presented in this paper is to develop a new third-order rDG method based on a hybrid recovery and reconstruction, termed P1P2(HLSr), for compressible flows on unstructured grids. The motivation to develop the new hybrid rDG method lies in the reason that the original least-squares reconstruction described above does not have the property of the 2-exactness, which is violated by the requirement that the value of the reconstructed solution is equal to the one of the underlying DG solution at the centroid of the face-neighboring cells. As a remedy, the developed hybrid rDG method combines the simplicity of the reconstruction-based DG method and the accuracy of the recovery-based DG method, and has the desired property of 2-exactness. A number of test cases for a variety of flow problems are presented to assess the performance of the new P1P2(HLSr) method. Numerical experiments demonstrate that this hybrid rDG method is able to achieve the designed optimal 3rd order of accuracy for both inviscid and viscous flows and outperform the rDG method based on either Green-Gauss or least-squares reconstruction.

The remainder of this paper is organized as follows. The governing equations and the traditional discontinuous Galerkin method are described in Section 2 and Section 3. The underlying reconstructed discontinuous Galerkin methods are presented in Section 4. Extensive numerical experiments are reported in Section 5. Concluding remarks are given in Section 6.

2. Governing equations

The unsteady compressible Navier-Stokes equations can be expressed as

$$\frac{\partial \mathbf{U}(x,t)}{\partial t} + \frac{\partial \mathbf{F}_k(\mathbf{U}(x,t))}{\partial x_k} = \frac{\partial \mathbf{G}_k(\mathbf{U}(x,t))}{\partial x_k}, \tag{1}$$

where the summation convention is used. The conservative variable vector **U**, advective (inviscid) flux vector **F**, and viscous flux vector **G** are defined by

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u_i \\ \rho e \end{pmatrix}, \quad \mathbf{F}_j = \begin{pmatrix} \rho u_j \\ \rho u_i u_j + p\delta_{ij} \\ u_j(\rho e + p) \end{pmatrix}, \quad \mathbf{G}_j = \begin{pmatrix} 0 \\ \sigma_{ij} \\ u_i(\sigma_{ij} + q_j) \end{pmatrix}. \tag{2}$$

Here ρ, p, and e denote the density, pressure, and specific total energy of the fluid, respectively. u_i is the velocity of the flow in the coordinate direction x_i. The pressure can be computed from the equation of state

$$p = (\gamma - 1)\rho\left(e - \frac{1}{2}u_j u_j\right), \tag{3}$$

where $\gamma = 1.4$ is the ratio of the specific heats. The components of the viscous stress tensor σ_{ij} and the heat flux vector q_j are given by

$$\sigma_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{ij}, \qquad q_j = \frac{1}{\gamma - 1}\frac{\mu}{Pr}\frac{\partial T}{\partial x_j}, \tag{4}$$

where, T is the temperature of the fluid, and Pr is the laminar Prandtl number which is taken as 0.7 for air. μ represents the molecular viscosity determined through Sutherland's law

$$\frac{\mu}{\mu_0} = \left(\frac{T}{T_0}\right)^{\frac{3}{2}} \frac{T_0 + S}{T + S},$$

where μ_0 denotes the viscosity at the reference temperature T_0 and $S = 110K$ is a constant. The temperature of the fluid T is determined by

$$T = \gamma\frac{p}{\rho}.$$

Neglecting viscous effects, the left-hand side of Eq. (1) represents the Euler equations governing unsteady compressible inviscid flows.

3. Discontinuous Galerkin method

The governing Eq. (1) are discretized using a discontinuous Galerkin finite element formulation. To formulate the discontinuous Galerkin method, we first introduce some notations. We assume that the flow domain Ω is subdivided into a collection of non-overlapping elements Ω_e. We introduce the following broken Sobolev space $\mathbf{V}_h^p$

$$\mathbf{V}_h^p = \{\upsilon_h \in [L_2(\Omega)]^m : \upsilon_h|_{\Omega_e} \in [V_p^m] \ \forall \Omega_e \in \Omega\}, \tag{5}$$

which consists of discontinuous vector-valued polynomial functions of degree p, and where m is the dimension of the unknown vector and V_p is the space of all polynomials of degree $\leq p$. To formulate the discontinuous Galerkin method, we introduce the following weak formulation, which is obtained by multiplying the above governing Eq. (1) by a test function W_h, integrating over an element Ω_e, and then performing an integration by parts: find $\mathbf{U}_h \in \mathbf{V}_h^p$, such that

$$\frac{d}{dt}\int_{\Omega_e} \mathbf{U}_h W_h d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k W_h d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h)\frac{\partial W_h}{\partial x_k} d\Omega$$
$$= \int_{\Gamma_e} \mathbf{G}_k(\mathbf{U}_h)\mathbf{n}_k W_h d\Gamma - \int_{\Omega_e} \mathbf{G}_k(\mathbf{U}_h)\frac{\partial W_h}{\partial x_k} d\Omega, \qquad \forall W_h \in \mathbf{V}_h^p, \tag{6}$$

where $\mathbf{U}_h$ and W_h are represented by a piecewise polynomial function of degrees p, which are discontinuous between the cell interfaces, and $\mathbf{n}_k$ denotes the unit outward normal vector to $\Gamma_e(=\partial\Omega_e)$: the boundary of Ω_e. Assume that B_i is the basis of polynomial function of degrees p. The DG formulation is then equivalent to the following system of N equations,

$$\frac{d}{dt}\int_{\Omega_e} \mathbf{U}_h B_i d\Omega + \int_{\Gamma_e} \widehat{\mathbf{F}}_k(\mathbf{U}_h)\mathbf{n}_k B_i d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k} d\Omega$$
$$= \int_{\Gamma_e} \widehat{\mathbf{G}}_k(\mathbf{U}_h)\mathbf{n}_k B_i d\Gamma - \int_{\Omega_e} \mathbf{G}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k} d\Omega, \qquad 1 \leq i \leq N, \tag{7}$$

where N is the dimension of the polynomial space. Since the numerical solution $\mathbf{U}_h$ is discontinuous between element interfaces, the interface fluxes are not uniquely defined. Like in the FV methods, the inviscid flux function $\widehat{\mathbf{F}}_k(\mathbf{U}_h)\mathbf{n}_k$ appearing in the boundary integral can be replaced by a numerical Riemann flux function $\mathbf{F}_k(\mathbf{U}_h^L, \mathbf{U}_h^R, \mathbf{n}_k)$ where $\mathbf{U}_h^L$, and $\mathbf{U}_h^R$ are the conservative state vector at the left and right side of the element boundary. The computation of the viscous fluxes in the boundary integral has to properly resolve the discontinuities at the interfaces. This scheme is called a DG method of degree p, or in short notation DG(p) method. Note that DG formulations are very similar to FV schemes, especially in their use of numerical fluxes. Indeed, the classical first-order cell-centered FV scheme exactly corresponds to the DG(p_0) method, i.e., to the DG method using a piecewise constant polynomial. Consequently, the DG(p_k) methods with $k > 0$ can be regarded as a natural generalization of FV methods to higher-order methods. By simply increasing the degree p of the polynomials, the corresponding higher-order DG methods are obtained. In this work, the HLLC Riemann solver is used to compute the inviscid fluxes and the second Bassi-Rebay (BR2) scheme [21] is used to compute the viscous fluxes.

Numerical polynomial solutions $\mathbf{U}_h$ in each element can be represented using a either nodal or modal DG formulation as follows

$$\mathbf{U}_h = \sum_i^N \mathbf{U}_i(t)B_i(\mathbf{x}), \tag{8}$$

where B_i, $(1 \leq i \leq N)$ are the finite element basis functions.

In this work, the Taylor basis functions [13, 15] are used. For example, in the DG(p_2) formulation, a quadratic polynomial solution $\mathbf{U}_h$ can be expressed as

$$\mathbf{U}_h = \widetilde{\mathbf{U}} + \frac{\partial \mathbf{U}}{\partial x}|_c \Delta x B_2 + \frac{\partial \mathbf{U}}{\partial y}|_c \Delta y B_3 + \frac{\partial^2 \mathbf{U}}{\partial x^2}|_c \Delta x^2 B_4$$
$$+ \frac{\partial^2 \mathbf{U}}{\partial y^2}|_c \Delta y^2 B_5 + \frac{\partial^2 \mathbf{U}}{\partial x \partial y}|_c \Delta x \Delta y B_6, \tag{9}$$

with

$$B_1 = 1,$$
$$B_2 = \frac{x - x_c}{\Delta x},$$
$$B_3 = \frac{y - y_c}{\Delta y},$$
$$B_4 = \frac{(x - x_c)^2}{2\Delta x^2} - \frac{1}{|\Omega_e|}\int_{\Omega_e} \frac{(x - x_c)^2}{2\Delta x^2} d\Omega,$$
$$B_5 = \frac{(y - y_c)^2}{2\Delta y^2} - \frac{1}{|\Omega_e|}\int_{\Omega_e} \frac{(y - y_c)^2}{2\Delta y^2} d\Omega,$$
$$B_6 = \frac{(x - x_c)(y - y_c)}{\Delta x \Delta y} - \frac{1}{|\Omega_e|}\int_{\Omega_e} \frac{(x - x_c)(y - y_c)}{\Delta x \Delta y} d\Omega, \tag{10}$$

where $\Delta x = 0.5(x_{max} - x_{min})$, $\Delta y = 0.5(y_{max} - y_{min})$, and (x_c, y_c) is the centroid of the element Ω_e. The unknowns to be solved in this formulation are the cell-averaged variables and their derivatives at the center of the cells, regardless of element shapes. This formulation belongs to the so-called modal discontinuous Galerkin method and has a number of attractive, distinct, and useful features. First, cell-averaged variables and their derivatives are handily available in this formulation. This makes the implementation of reconstruction schemes and Hermite WENO limiters straightforward and simple [15,36]. Secondly, the Taylor basis is hierarchic. This greatly facilitates implementation of p-multigrid methods and p-refinement. Thirdly, the same basis functions are used for any shapes of elements: triangles and quadrilaterals in 2D and tetrahedron, pyramid, prism, and hexahedron in 3D. This makes the implementation of DG methods on arbitrary meshes straightforward. It should be noted that, unfortunately, the Taylor basis also suffers from a drawback [29] that under some particular mesh geometry with higher order polynomials, the inverse of mass matrix may not exist, although it rarely happens in general application.

The spatial discretization Eq. (7) leads to a system of ordinary differential equations:

$$\mathbf{M}\frac{d\mathbf{U}}{dt} + \mathbf{R}(\mathbf{U}) = 0, \tag{11}$$

where $\mathbf{M}$ denotes the mass matrix, $\mathbf{U}$ is the solution vector, and $\mathbf{R}$ is residual vector. Using Euler implicit time-integration, the spatially discretized Navier-Stokes Eq. (7) can be linearized in time and written as

$$\left(\frac{\mathbf{M}}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)\Delta \mathbf{U} = -\mathbf{R}(\mathbf{U}), \tag{12}$$

where $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ is the Jacobian matrix, Δt is the time increment, and $\Delta \mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$ is the solution difference between time level n and $n+1$. This system of linear equations at each time step is solved by a GMRES+LU-SGS method [34].

4. Reconstructed discontinuous Galerkin method

In comparison with reconstructed FV methods, the DG methods have a significant drawback in that they require more degrees of freedom, additional domain integration, and more Gauss quadrature points for the boundary integration, and therefore more computational costs and storage requirements. On the one hand, the reconstruction methods that FV methods use to achieve higher-order accuracy are relatively inexpensive but less accurate and robust. On the other hand, the DG methods that can be viewed as a

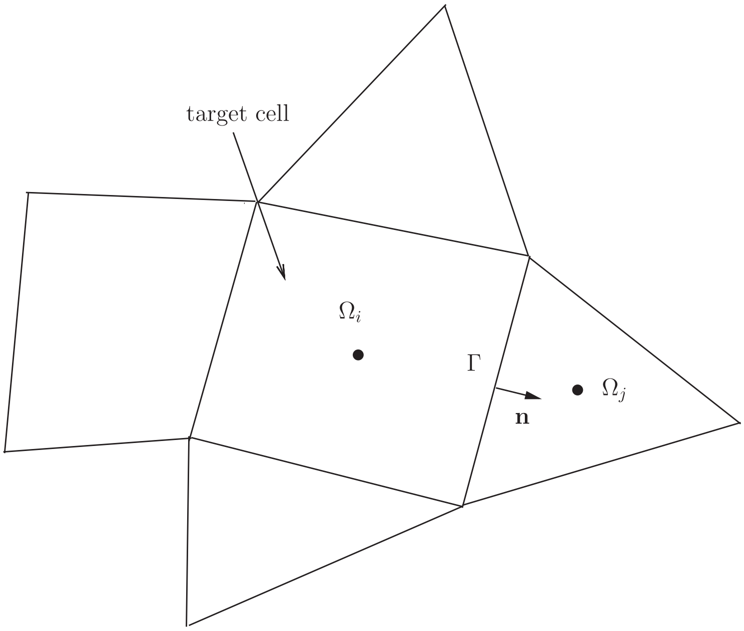


Fig. 1. Representation of the target cell and its neighbors for reconstruction.

different way to extend a FV method to higher orders, are accurate and robust but costly. It is only natural and tempting to combine the efficiency of the reconstruction methods and the accuracy of the DG methods. The rDG methods represent a good step in this direction, which can be expressed as

$$\frac{d}{dt}\int_{\Omega_e}\mathbf{U}_{p_n}B_i d\Omega+\int_{\Gamma_e}\widehat{\mathbf{F}}_k(\mathbf{U}_{p_m})\mathbf{n}_k B_i d\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U}_{p_m})\frac{\partial B_i}{\partial x_k}d\Omega$$
$$=\int_{\Gamma_e}\widehat{\mathbf{G}}_k(\mathbf{U}_{P_m})\mathbf{n}_k B_i d\Gamma-\int_{\Omega_e}\mathbf{G}_k(\mathbf{U}_{p_m})\frac{\partial B_i}{\partial x_k}d\Omega, \qquad 1\le i\le N, \quad (13)$$

where $\mathbf{U}_{p_n}$ indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and $\mathbf{U}_{p_m}$ represents a reconstructed polynomial solution of degree of $m (m \ge n)$ that is used to compute the fluxes and source terms. The resulting scheme is called rDG(p_np_m). The beauty of rDG(p_np_m) schemes is that they provide a unified formulation for both FV and DG methods, and contain both classical FV and standard DG methods as two special cases of rDG(p_np_m) schemes.

In the case of DG(P$_1$) method, a linear polynomial solution $\mathbf{U}_i$ in any cell Ω_i is expressed as

$$\mathbf{U}_i=\widetilde{\mathbf{U}}_i+\frac{\partial \mathbf{U}}{\partial x}|_i\Delta x_i B_2+\frac{\partial \mathbf{U}}{\partial y}|_i\Delta y_i B_3. \quad (14)$$

A quadratic polynomial solution $\mathbf{U}_i^R$ can be reconstructed using the underlying linear polynomial DG solution from its neighboring cells as follows:

$$\mathbf{U}_i^R=\widetilde{\mathbf{U}}_i^R+\frac{\partial \mathbf{U}^R}{\partial x}|_i\Delta x_i B_2+\frac{\partial \mathbf{U}^R}{\partial y}|_i\Delta y_i B_3$$
$$+\frac{\partial^2 \mathbf{U}^R}{\partial x^2}|_i\Delta x_i^2 B_4+\frac{\partial^2 \mathbf{U}^R}{\partial y^2}|_i\Delta y_i^2 B_5+\frac{\partial^2 \mathbf{U}^R}{\partial x\partial y}|_i\Delta x_i\Delta y_i B_6. \quad (15)$$

In order to maintain the compactness of the DG methods, the reconstruction is required to only involve Von Neumann neighborhood, i.e., the adjacent cells that share a common face with the cell Ω_i. There are six degrees of freedom, and therefore six unknowns to be determined. However, the first three unknowns can be trivially obtained, by requiring the reconstruction scheme to be conservative, and the values of the reconstructed first-order derivatives to be equal to the ones of the underlying DG solut ion as shown in Fig. 1. Due to the judicious choice of Taylor basis in our DG formulation, these three degrees of freedom (cell average and first derivatives) simply coincide with the ones from the underlying DG solution, i.e.,

$$\mathbf{U}_i^R=\widetilde{\mathbf{U}}_i, \quad \frac{\partial \mathbf{U}^R}{\partial x}|_i=\frac{\partial \mathbf{U}}{\partial x}|_i, \quad \frac{\partial \mathbf{U}^R}{\partial y}|_i=\frac{\partial \mathbf{U}}{\partial y}|_i. \quad (16)$$

As a result, only three second derivatives of the solution need to be determined. We will present four methods for determining these three second derivatives (curvatures), in the following sections.

4.1. Green-Gauss reconstruction

Green-Gauss reconstruction [31–33] is probably the simplest reconstruction scheme, which is widely used in the finite volume methods to reconstruct a gradient of a solution from its cell-averaged values. Similarly, second derivatives of a solution in a given cell Ω_i can be reconstructed from its first derivatives using Green's theorem as follows,

$$\mathbf{U}_{xxi}^R=\frac{1}{|\Omega_i|}\int_{\Omega_i}\frac{\partial^2 \mathbf{U}}{\partial x^2}d\Omega=\frac{1}{|\Omega_i|}\sum_{\Gamma_i\in\partial\Omega_i}\int_{\Gamma_i}\frac{\partial \mathbf{U}}{\partial x}n_x^{(i)}d\Gamma,$$
$$\mathbf{U}_{yyi}^R=\frac{1}{|\Omega_i|}\int_{\Omega_i}\frac{\partial^2 \mathbf{U}}{\partial y^2}d\Omega=\frac{1}{|\Omega_i|}\sum_{\Gamma_i\in\partial\Omega_i}\int_{\Gamma_i}\frac{\partial \mathbf{U}}{\partial y}n_y^{(i)}d\Gamma. \quad (17)$$

The cross derivatives can be reconstructed as follows,

$$\mathbf{U}_{xyi}^R=\frac{1}{|\Omega_i|}\int_{\Omega_i}\frac{\partial^2 \mathbf{U}}{\partial x\partial y}d\Omega=\frac{1}{|\Omega_i|}\sum_{\Gamma_i\in\partial\Omega_i}\int_{\Gamma_i}\frac{\partial \mathbf{U}}{\partial x}n_y^{(i)}d\Gamma, \quad (18)$$

or

$$\mathbf{U}_{yxi}^R=\frac{1}{|\Omega_i|}\int_{\Omega_i}\frac{\partial^2 \mathbf{U}}{\partial x\partial y}d\Omega=\frac{1}{|\Omega_i|}\sum_{\Gamma_i\in\partial\Omega_i}\int_{\Gamma_i}\frac{\partial \mathbf{U}}{\partial y}n_x^{(i)}d\Gamma. \quad (19)$$

The cross-derivatives can be computed using either Eq. (18) or Eq. (19). In the present work, an arithmetic mean is used to obtain the cross-derivatives.

There are several choices in selecting the integral domain and calculating the first-order derivatives along faces Γ_i in Eqs. (17)–(19). In this work, we adopt the cell-vertex type implementation, where the values of first derivatives at each nodes are computed through a weighted average approach with its surrounding elements, and then. a two-point quadrature rule is directly used to calculate the face integral. The beauty of this reconstruction scheme is its simplicity, efficiency, and robustness, since it does not need to solve an over-determined system which is required by a least-squares approach. However, Green-Gauss reconstruction is less accurate than the least-squares reconstruction, as it only uses the information on first derivatives. The rDG method based on this Green-Gauss reconstruction is named P1P2(GG) hereafter.

4.2. Least-squares recovery

The least-squares recovery [18,19,25–27] relies on L_2-projection to determine the second derivatives. Consider a face-neighboring cell Ω_j, the recovery principle requires

$$\int_{\Omega_j} \mathbf{U}_i^R B_k^{(j)}\, d\Omega = \int_{\Omega_j} \mathbf{U}_j B_k^{(j)}\, d\Omega, \qquad 1 \le k \le 3, \tag{20}$$

where $B_k^{(j)}$ are the basis functions associated with element j. Note that during the recovery process, the recovered polynomial solution is continuously extended over the face-neighboring cells, and the locally recovered solution is indistinguishable from the underlying DG solution in the neighboring cells in the weak sense. The same recovery equations constituted by Eq. (20) can be written for all cells connected to the cell Ω_i with a common face, which leads to a non-square matrix. The number of face-neighboring cells for a triangular and quadrilateral cell is three and four, respectively. As a result, the size of the resulting non-square matrix is 9 × 3 and 12 × 3, respectively. This over-determined linear system of 9 or 12 equations for 3 unknowns can be solved in the least-squares sense. One disadvantage of the recovery is the need to compute the integral on the left-hand-side of the recovered equations, which is done using the classical multidimensional Gaussian quadrature of an appropriate order. Furthermore, the recovery might be problematic for a boundary cell, where the number of the face-neighboring cells might be not enough to recover a polynomial solution of a desired order. For example, a corner tetrahedral cell with three boundary faces has only one face-neighboring cell, which can only provide four recovered equations. However, there are six second-order derivatives need to be determined. In this case, one cannot recover a quadratic polynomial solution from the underlying linear discontinuous Galerkin solution. This situation can be remedied by use of extended one-sided stencils, although the compactness of the underlying DG methods is then sacrificed. The resulting rDG method using this least-squares recovery will be referred to as P1P2(LSR) in this paper.

4.3. Least-squares reconstruction

Alternatively, the remaining three degrees of freedom can be determined by requiring that the reconstructed solution and its first derivatives are equal to the underlying DG solution and its first derivatives for all the adjacent face neighboring cells [28–30]. Consider a neighboring cell Ω_j, one requires

$$\begin{aligned}
\mathbf{U}_j &= \widetilde{\mathbf{U}}_i + \mathbf{U}_{xi}\Delta x_i B_2^{(i)}(\mathbf{x}_j) + \mathbf{U}_{yi}\Delta y_i B_3^{(i)}(\mathbf{x}_j) + \mathbf{U}^R_{xxi}\Delta x_i^2 B_4^{(i)}(\mathbf{x}_j)\\
&\quad + \mathbf{U}^R_{yyi}\Delta y_i^2 B_5^{(i)}(\mathbf{x}_j) + \mathbf{U}^R_{xyi}\Delta x_i \Delta y_i B_6^{(i)}(\mathbf{x}_j),\\
\mathbf{U}_{xj} &= \mathbf{U}_{xi} + \mathbf{U}^R_{xxi}\Delta x_i B_2^{(i)}(\mathbf{x}_j) + \mathbf{U}^R_{xyi}\Delta y_i B_3^{(i)}(\mathbf{x}_j),\\
\mathbf{U}_{yj} &= \mathbf{U}_{yi} + \mathbf{U}^R_{yyi}\Delta y_i B_3^{(i)}(\mathbf{x}_j) + \mathbf{U}^R_{xyi}\Delta x_i B_2^{(i)}(\mathbf{x}_j),
\end{aligned} \tag{21}$$

where the basis functions $B_k^{(i)}(\mathbf{x}_j)$ are evaluated at the center of cell Ω_j. This can be written in a matrix form as follows:

$$\begin{pmatrix} B_4^{(i)} & B_5^{(i)} & B_6^{(i)} \\ B_2^{(i)} & 0 & B_3^{(i)} \\ 0 & B_3^{(i)} & B_2^{(i)} \end{pmatrix}
\begin{pmatrix} \mathbf{U}^R_{xxi}\Delta x_i^2 \\ \mathbf{U}^R_{yyi}\Delta y_i^2 \\ \mathbf{U}^R_{xyi}\Delta x_i \Delta y_i \end{pmatrix}
= \begin{pmatrix} \mathbf{U}_j - (\widetilde{\mathbf{U}}_i + \mathbf{U}_{xi}\Delta x_i B_2^{(i)} + \mathbf{U}_{yi}\Delta y_i B_3^{(i)}) \\ \Delta x_i(\mathbf{U}_{xj} - \mathbf{U}_{xi}) \\ \Delta y_i(\mathbf{U}_{yj} - \mathbf{U}_{yi}) \end{pmatrix}
= \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \\ \mathbf{R}_3 \end{pmatrix} \tag{22}$$

where $\mathbf{R}$ is used to represent the right-hand-side for simplicity. Similar equations could be written for all cells connected to the cell Ω_i with a common face, which leads to a non-square matrix. The size of the resulting non-square matrix is 9 × 3 and 12 × 3, respectively. This over-determined linear system of 9 or 12 equations for 3 unknowns can be solved in the least-squares sense. In the present work, it is solved using a normal equation approach, which, by pre-multiplying through by matrix transpose, yields a symmetric linear system of equations 3 × 3 as follows

$$\begin{pmatrix}
\sum\limits_j (B_2^{(i)}B_2^{(i)} + B_4^{(i)}B_4^{(i)}) & \sum\limits_j B_4^{(i)}B_5^{(i)} & \sum\limits_j (B_2^{(i)}B_3^{(i)} + B_4^{(i)}B_6^{(i)}) \\
\sum\limits_j B_4^{(i)}B_5^{(i)} & \sum\limits_j (B_3^{(i)}B_3^{(i)} + B_5^{(i)}B_5^{(i)}) & \sum\limits_j (B_2^{(i)}B_3^{(i)} + B_5^{(i)}B_6^{(i)}) \\
\sum\limits_j (B_2^{(i)}B_3^{(i)} + B_4^{(i)}B_6^{(i)}) & \sum\limits_j (B_2^{(i)}B_3^{(i)} + B_5^{(i)}B_6^{(i)}) & \sum\limits_j (B_2^{(i)}B_2^{(i)} + B_3^{(i)}B_3^{(i)} + B_6^{(i)}B_6^{(i)})
\end{pmatrix}
\begin{pmatrix} \mathbf{U}^R_{xxi}\Delta x_i^2 \\ \mathbf{U}^R_{yyi}\Delta y_i^2 \\ \mathbf{U}^R_{xyi}\Delta x_i \Delta y_i \end{pmatrix}
= \begin{pmatrix} \sum\limits_j (B_4^{(i)}\mathbf{R}_1 + B_2^{(i)}\mathbf{R}_2) \\ \sum\limits_j (B_5^{(i)}\mathbf{R}_1 + B_3^{(i)}\mathbf{R}_3) \\ \sum\limits_j (B_6^{(i)}\mathbf{R}_1 + B_3^{(i)}\mathbf{R}_2 + B_2^{(i)}\mathbf{R}_3) \end{pmatrix} \tag{23}$$

This linear system of 3 × 3 can be then trivially solved to obtain the second-order derivatives of the reconstructed quadratic polynomial solution. The rDG method based on this least-square reconstruction will be referred as P1P2(LSr) method from now on.

4.4. Hybrid least-squares reconstruction

It is well-known that the least-squares reconstruction used in the FV methods to reconstruct a solution gradient has the so-called 1-exact property. Unfortunately, the least-squares reconstruction described above does not have the property of the 2-exactness, which is violated by the requirement that the value of the reconstructed solution is equal to the one of the underlying DG solution at the centroid of the face-neighboring cells. In order to have the property of the 2-exactness, a new hybrid reconstruction approach is proposed in this work. This new reconstruction is hybrid in the sense that it uses one equation from the recovery and two equations from the reconstruction as follows:

$$\begin{aligned}
&\int_{\Omega_j} \mathbf{U}_i^R B_1^{(j)}\, d\Omega = \int_{\Omega_j} \mathbf{U}_j B_1^{(j)}\, d\Omega,\\
&\mathbf{U}_{xj} = \mathbf{U}_{xi} + \mathbf{U}^R_{xxi}\Delta x_i B_2^{(i)}(\mathbf{x}_j) + \mathbf{U}^R_{xyi}\Delta y_i B_3^{(i)}(\mathbf{x}_j),\\
&\mathbf{U}_{yj} = \mathbf{U}_{yi} + \mathbf{U}^R_{yyi}\Delta y_i B_3^{(i)}(\mathbf{x}_j) + \mathbf{U}^R_{xyi}\Delta x_i B_2^{(i)}(\mathbf{x}_j).
\end{aligned} \tag{24}$$

Specifically, instead of using the centroid value of the reconstructed DG solution that is required to be the one of the underlying DG solution at its adjacent neighboring cell Ω_j in Eq. (21), we use the cell average $\widetilde{\mathbf{U}}_j$ as follows

$$\begin{aligned}
\widetilde{\mathbf{U}}_j &= \frac{1}{|\Omega_j|}\int_{\Omega_j} (\widetilde{\mathbf{U}}_i + \mathbf{U}_{xi}\Delta x_i B_2^{(i)} + \mathbf{U}_{yi}\Delta y_i B_3^{(i)} + \mathbf{U}^R_{xxi}\Delta x_i^2 B_4^{(i)}\\
&\quad + \mathbf{U}^R_{yyi}\Delta y_i^2 B_5^{(i)} + \mathbf{U}^R_{xyi}\Delta x_i \Delta y_i B_6^{(i)})\, d\Omega.
\end{aligned} \tag{25}$$

Similar to the least-squares recovery or least-squares reconstruction, the least-square hybrid reconstruction leads to a symmetric linear system of equations 3 × 3

$$\begin{pmatrix} \sum\limits_j (B_2^{(i)}B_2^{(i)} + \widetilde{B}_4^{(i)}\widetilde{B}_4^{(i)}) & \sum\limits_j \widetilde{B}_4^{(i)}\widetilde{B}_5^{(i)} & \sum\limits_j (B_2^{(i)}B_3^{(i)} + \widetilde{B}_4^{(i)}\widetilde{B}_6^{(i)}) \\ \sum\limits_j \widetilde{B}_4^{(i)}\widetilde{B}_5^{(i)} & \sum\limits_j (B_3^{(i)}B_3^{(i)} + \widetilde{B}_5^{(i)}\widetilde{B}_5^{(i)}) & \sum\limits_j (B_2^{(i)}B_3^{(i)} + \widetilde{B}_5^{(i)}\widetilde{B}_6^{(i)}) \\ \sum\limits_j (B_2^{(i)}B_3^{(i)} + \widetilde{B}_4^{(i)}\widetilde{B}_6^{(i)}) & \sum\limits_j (B_2^{(i)}B_3^{(i)} + \widetilde{B}_5^{(i)}\widetilde{B}_6^{(i)}) & \sum\limits_j (B_2^{(i)}B_2^{(i)} + B_3^{(i)}B_3^{(i)} + \widetilde{B}_6^{(i)}\widetilde{B}_6^{(i)}) \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{U}^R_{xxi}\Delta x_i^2 \\ \mathbf{U}^R_{yyi}\Delta y_i^2 \\ \mathbf{U}^R_{xyi}\Delta x_i \Delta y_i \end{pmatrix} = \begin{pmatrix} \sum\limits_j (\widetilde{B}_4^{(i)}\widetilde{\mathbf{R}}_1 + B_2^{(i)}\mathbf{R}_2) \\ \sum\limits_j (\widetilde{B}_5^{(i)}\widetilde{\mathbf{R}}_1 + B_3^{(i)}\mathbf{R}_3) \\ \sum\limits_j (\widetilde{B}_6^{(i)}\widetilde{\mathbf{R}}_1 + B_3^{(i)}\mathbf{R}_2 + B_2^{(i)}\mathbf{R}_3) \end{pmatrix} \tag{26}$$

where

$$\widetilde{B}_k^{(i)} = \frac{1}{|\Omega_j|}\int_{\Omega_j} B_k^{(i)}\ d\Omega, \qquad k = 4, 5, 6, \tag{27}$$

and

$$\widetilde{\mathbf{R}}_1 = \widetilde{\mathbf{U}}_j - \frac{1}{|\Omega_j|}\int_{\Omega_j} (\widetilde{\mathbf{U}}_i + \mathbf{U}_{xi}\Delta x_i B_2^{(i)} + \mathbf{U}_{yi}\Delta y_i B_3^{(i)}\ )d\Omega. \tag{28}$$

The above linear system of 3 × 3 can be solved to obtain the second-order derivatives.

Compared to the original least-squares reconstruction described previously, the new approach uses cell average of the underlying solution based on DG weak form instead of using strong interpolation with the centroid value. This modification is essential to attain the designed optimal order of accuracy for solving the compressible Navier-Stokes equations. The inherent reason why the reconstruction based on weak form becomes critical in the current work is that, when Taylor basis is adopted, the value at cell centroid is no longer equal to the cell average for higher-order polynomials. Thus, strong interpolation via cell centroid values does not have the property of 2-exactness. Fortunately, one simple remedy is the hybrid approach developed in this work, which has the property of 2-exactness. This new hybrid least-squares rDG method will be referred to as P1P2(HLSr) method from now on.

5. Numerical experiments

5.1. A convergence study on reconstruction methods

This test case is chosen to demonstrate the superior accuracy and the 2-exactness of the new hybrid reconstruction method in comparison with the original reconstruction method. The computational domain is a rectangle ($0 \leq x \leq 4$ and $0 \leq y \leq 2$). A grid convergence study is conducted on three successively refined unstructured grids with 128, 512 and 2048 cells, respectively. A fully quadratic polynomial function

$$f(x, y) = 1 + x + y + x^2 + xy + y^2$$

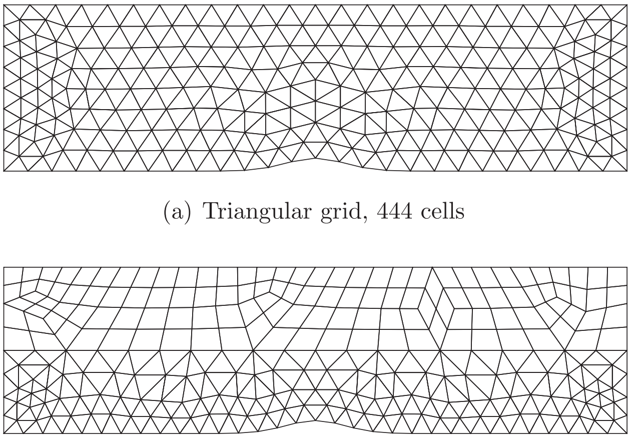
and a smooth function

$$f(x, y) = sin(0.5\pi x)cos(\pi y)$$

are used to assess the accuracy, the order of convergence, and the 2-exact property of the two reconstruction methods. Numerical results presented in Table. 1 indicate that the original least-squares reconstruction does not strictly have the 2-exact property for a fully quadratic polynomial, and the new hybrid least-squares reconstruction, however, can reconstruct the quadratic polynomial exactly, i.e., having the so-called property of the 2-exactness. As for the generally smooth function, both the original least-squares reconstruction and the new hybrid least-squares reconstruction can achieve the designed order of accuracy and the new hybrid least-squares reconstruction is more accurate than its original counterpart, especially on a fine mesh. As will be seen later, the satisfaction of the 2-exact property can improve the spatial accuracy for inviscid flow simulations and becomes critical for the rDG methods to obtain the designed optimal order of accuracy for viscous flow simulations.

Table 1
Convergence study on reconstruction methods.

	P1P2(LSr)		P1P2(HLSr)	
Num of cells	L_2-error	Order	L_2-error	Order
Case 1: $f(x, y) = 1.0 + x + y + x^2 + xy + y^2$				
128	4.19E-3		1.47e-012	
512	1.01E-3	2.05	2.79e-012	———
2048	2.46E-4	2.03	5.63e-012	———
Case 2: $f(x, y) = sin(0.5\pi x)cos(\pi y)$				
128	1.94E-2		1.95E-2	
512	2.38E-3	3.03	2.34E-3	3.06
2048	3.20E-4	2.89	2.84E-4	3.04


(a) Triangular grid, 444 cells

(b) Hybrid grid, 380 cells

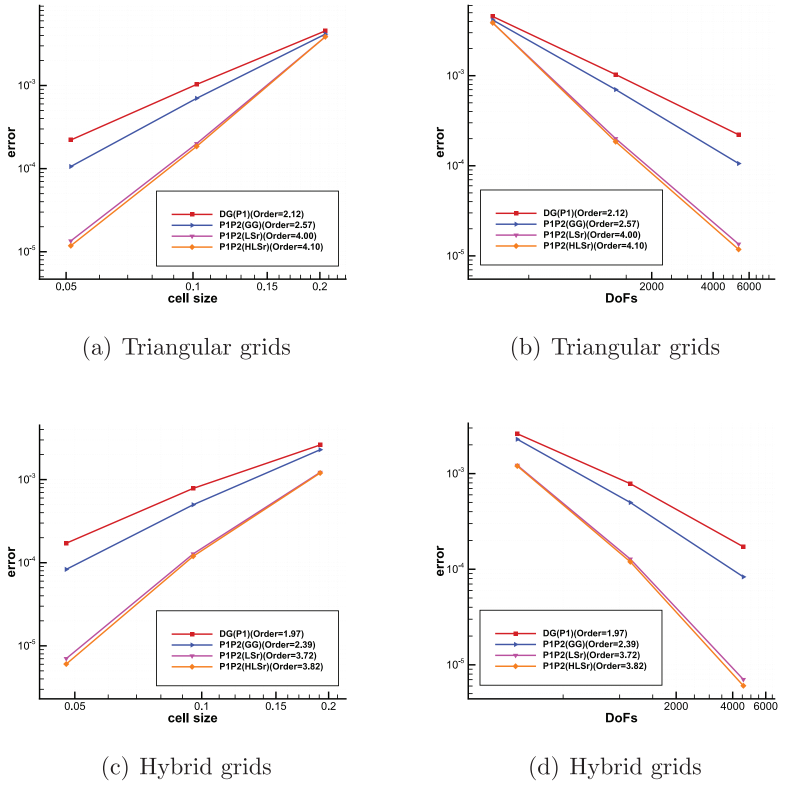
Fig. 2. Two types of grids used for computing a subsonic flow through channel with a smooth bump.

5.2. Inviscid subsonic flow through a channel with a smooth bump

This test case is selected to verify the accuracy and the convergence of the three rDG methods: P1P2(GG), P1P2(LSr), and P1P2(HLSr) for computing internal flows. The problem under consideration is a inviscid subsonic flow through a channel with a smooth bump. The computational domain is bounded between -1.5 to 1.5 in the x-coordinate and 0 to 0.8 in the y-coordinate. The bump defined by $y = 0.0625e^{-25x^2}$. The inflow Mach number is 0.5 with an angle of attack 0^o. A grid convergence study is conducted on two types of three successively refined second-order grids: one triangular grid and one hybrid triangular and quadrilateral grid as shown in Fig. 2, where the medium grids are only displayed. In this case, the L_2-norm of the following entropy production ε defined as

$$\varepsilon = \frac{S - S_\infty}{S_\infty} = \frac{p}{p_\infty}(\frac{\rho_\infty}{\rho})^\gamma - 1, \tag{29}$$

is used for error measurement. Note that the entropy production is a very good criteria to measure accuracy of the numerical solutions as the flow is isentropic. The L_2-errors and the order of convergence obtained for the DG(P1) and the three rDG(P1P2) methods are presented in Table. 2 and in Fig. 3. It can be seen clearly that



(a) Triangular grids (b) Triangular grids

(c) Hybrid grids (d) Hybrid grids

Fig. 3. Comparison of the convergence order for different rDG methods for computing a subsonic flow through channel with a smooth bump.

Table 2
Comparison of convergence order among the three reconstruction DG methods for computing a subsonic flow through channel with a smooth bump.

Triangular grids	P1P2(GG)		P1P2(LSr)		P1P2(HLSr)	
Num of cells	L_2-error	Order	L_2-error	Order	L_2-error	Order
111	3.79E-3		3.42E-3		3.45E-3	
444	6.64E-4	2.51	1.89E-4	4.17	1.77E-4	4.28
1776	1.07E-4	2.63	1.33E-5	3.83	1.17E-5	3.92
Hybrid grids	P1P2(GG)		P1P2(LSr)		P1P2(HLSr)	
Num of cells	L_2-error	Order	L_2-error	Order	L_2-error	Order
95	2.29E-3		1.22E-3		1.20E-3	
380	4.98E-4	2.20	1.28E-4	3.25	1.19E-4	3.33
1520	8.32E-5	2.58	7.02E-6	4.19	6.04E-6	4.30

Table 3
Comparison of convergence order among the three reconstruction DG methods for computing a subsonic flow past a circular cylinder.

Triangular grids	P1P2(GG)		P1P2(LSr)		P1P2(HLSr)	
Num of nodes	L_2-error	Order	L_2-error	Order	L_2-error	Order
16 × 5	5.08E-2		2.75E-2		2.81E-2	
32 × 9	4.01E-3	3.66	9.17E-4	4.91	9.99E-4	4.81
64 × 17	4.34E-4	3.21	5.94E-5	3.95	6.34E-5	3.98
128 × 33	6.20E-5	2.81	6.60E-6	3.17	6.02E-6	3.40
Hybrid grids	P1P2(GG)		P1P2(LSr)		P1P2(HLSr)	
Num of nodes	L_2-error	Order	L_2-error	Order	L_2-error	Order
16 × 5	6.85E-2		4.95E-2		4.93E-2	
32 × 9	3.63E-3	4.23	2.80E-3	4.14	2.86E-3	4.11
64 × 17	4.32E-4	3.07	2.01E-4	3.80	1.96E-4	3.87
128 × 33	6.80E-5	2.67	1.75E-5	3.52	1.58E-5	3.63

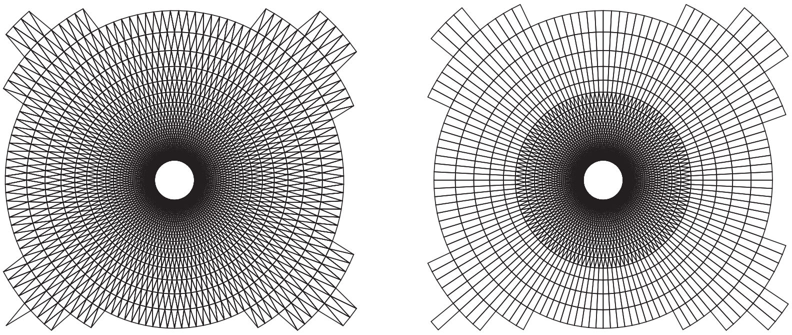
all three rDG methods can significantly improve the spatial accuracy as compared with the underlying DG(P1) method. The new hybrid rDG method: P1P2(HLSr) is the best among the three rDG methods, as it delivers the smallest magnitude of error and the highest order of accuracy.

5.3. Inviscid subsonic flow past a circular cylinder

An inviscid subsonic flow past a circular cylinder at $Ma = 0.38$ is considered in this test case to verify if a formal order of the convergence rate for the three rDG methods can be achieved for the compressible Euler equations. The radius of the cylinder is $r_1 = 0.5$, the domain is bounded by $r_{33} = 20$ and the radius of concentric circles for very fine mesh are set up as

$$r_i = r_1\left(1 + \frac{2\pi}{128}\sum_{j=0}^{i-1}\alpha^j\right), \qquad i = 2, \cdots, 33, \tag{30}$$

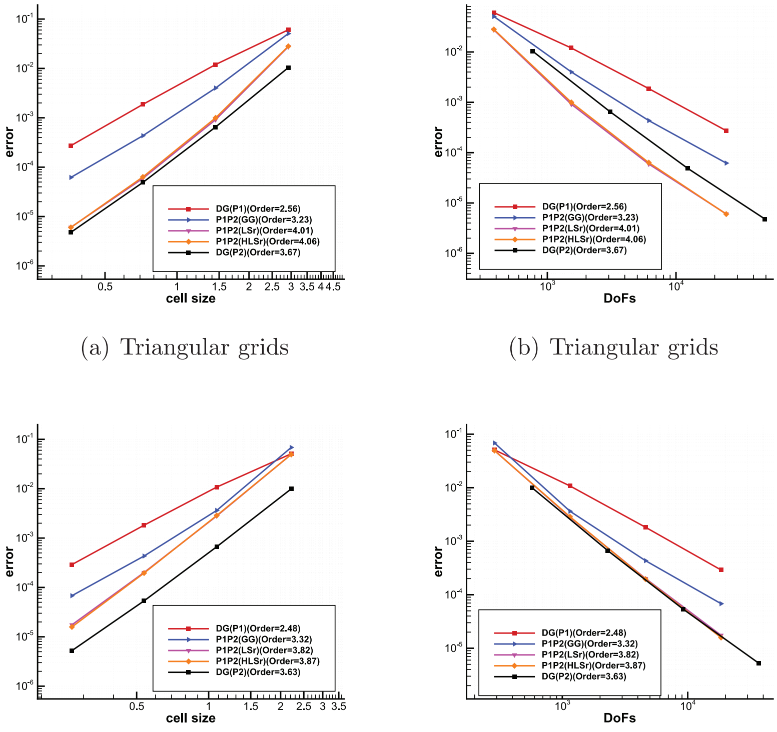
where $\alpha = 1.1580372$. Two types of four successively refined grids: triangular grids and hybrid grids, are used in this test case for conducting a convergence study of the three rDG methods, which have 16 × 5, 32 × 9, 64 × 17, and 128 × 33 grid points, respectively. The first number is the number of points in the angular direction, and the second number is the number of points in the radial direction. The coarser grids are generated by successively coarsing the very fine mesh. Fig. 4 shows the 128 × 33 grids for the triangular and hybrid grids, respectively. Like in the previous test case, the L_2-norm of the entropy production is served as the error measurement. The errors and the convergence orders for the three rDG(P1P2) together with DG(P1) and DG(P2) methods are presented in Table. 3 and Fig. 5. As expected, all three rDG(P1P2) methods offer a full $O(h^{p+2})$ order of the convergence, adding one order of accuracy to the underlying DG(P1) method. Again, our newly developed hybrid rDG method: P1P2(HLSr) provides the best performance in terms of both the magnitude of the error and the order of the convergence. Fig. 6 shows a comparison of convergence history among the DG(P1), DG(P2), and three rDG methods. As expected, the DG(P1) method provides the best con-



(a) Triangular grid, 128×33

(b) Hybrid grid, 128×33

Fig. 4. Two types of grids used for a subsonic flow past a circular cylinder.



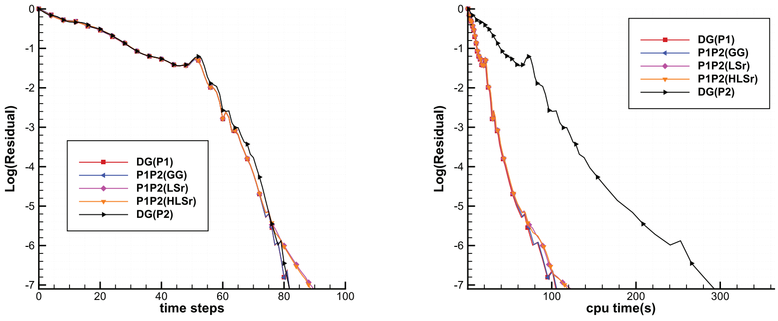
(a) Triangular grids

(b) Triangular grids

(c) Hybrid grids

(d) Hybrid grids

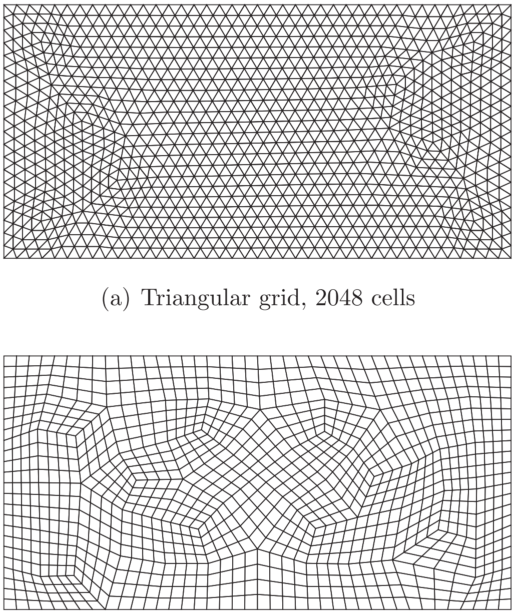
Fig. 5. Comparison of convergence order among DG(P1), DG(P2) and reconstructed DG methods for computing a subsonic flow past a circular cylinder.



(a) time steps

(b) cpu time(s)

Fig. 6. Comparison of convergence history for computing a subsonic flow past a circular cylinder, based on hybrid fine grid.



(a) Triangular grid, 2048 cells

(b) Quadrilateral grid, 1072 cells

Fig. 7. Two types of grids used for computing Couette flow.

vergence performance and all three rDG(P1P2) methods are much faster than the DG(P2) method, clearly demonstrating the advantage of the rDG(P1P2) method over the DG(P2) method.

5.4. Couette flow

A compressible Couette flow is considered in this test case to verify the accuracy and convergence of the rDG methods for solving the compressible Navier-Stokes equations on arbitrary grids. It is the laminar viscous flow between two parallel plates. The bottom plate with a fixed temperature (T_0) is stationary, while the top one with a fixed temperature (T_1) moves at a velocity of U. The two plates are separated by a distance H. Assume that the viscosity coefficient μ is a constant. The analytic solution is given by

Table 4
Comparison of convergence order among the three reconstructed DG methods for computing Couette flow.

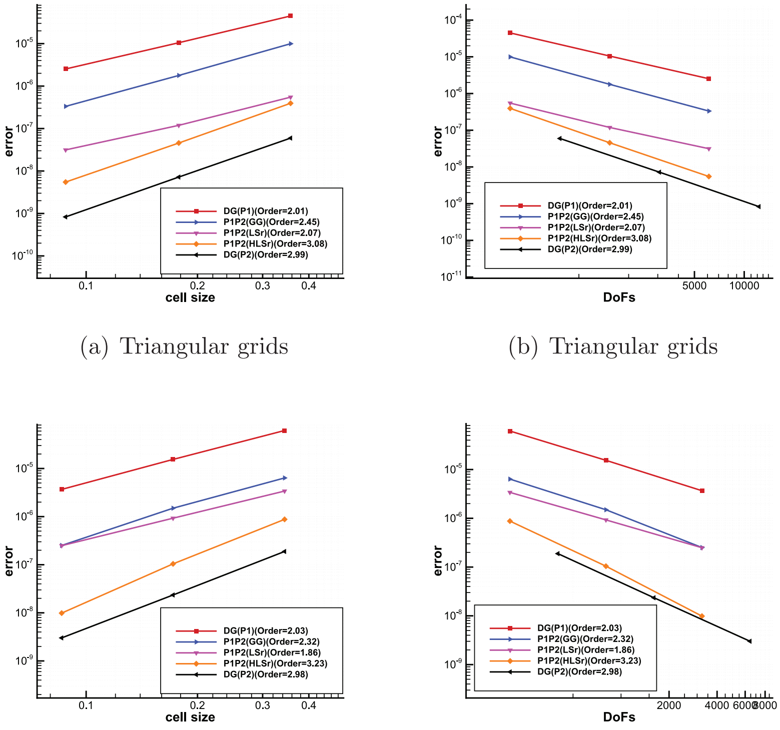
Triangular grids	P1P2(GG)		P1P2(LSr)		P1P2(HLSr)	
Num of cells	L_2-error	Order	L_2-error	Order	L_2-error	Order
128	9.95E-6		5.49E-7		3.94E-7	
512	1.78E-6	2.48	1.19E-7	2.21	4.56E-8	3.11
2048	3.33E-7	2.42	3.14E-8	1.92	5.49E-9	3.05
Quadrilateral grids	P1P2(GG)		P1P2(LSr)		P1P2(HLSr)	
Num of cells	L_2-error	Order	L_2-error	Order	L_2-error	Order
67	6.36E-6		3.40E-6		8.76E-7	
268	1.50E-6	2.08	9.27E-7	1.87	1.04E-7	3.07
1072	2.51E-7	2.58	2.49E-7	1.90	9.86E-9	3.40

$$u = \frac{y}{H}U, \qquad v = 0,$$

$$p = p_\infty, \qquad \rho = \frac{p}{RT},$$

$$T = T_0 + \frac{y}{H}(T_1 - T_0) + \frac{y}{H}\left(1.0 - \frac{y}{H}\right)\frac{PrU^2}{2c_p}, \tag{31}$$

where Pr is the Prandtl number and c_p is the specific heat capacity at constant pressure. For our numerical experiments, we take H=2, $T_0 = 0.8$, and $T_1 = 0.85$. The Mach number for the upper wall $M_1 = 0.1$, and the Reynolds number $Re = 100$ with a constant



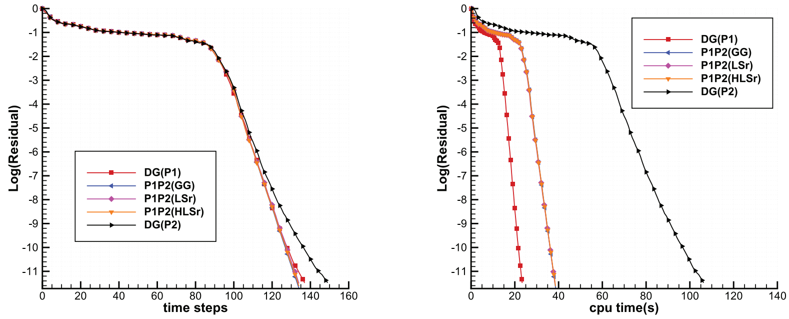
(a) Triangular grids

(b) Triangular grids

(c) Quadrilateral grids

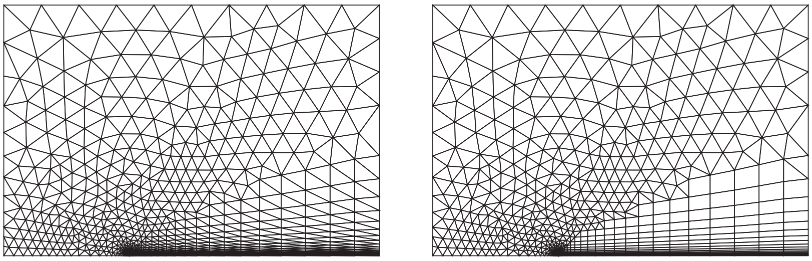
(d) Quadrilateral grids

Fig. 8. Comparison of convergence order among DG(P1), DG(P2) and reconstructed DG methods for computing Couette flow.



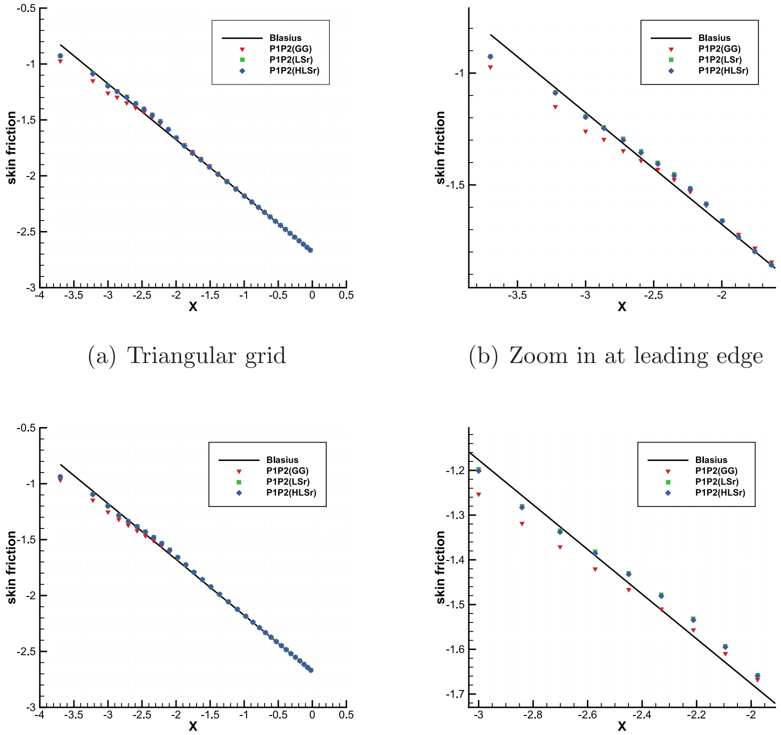
(a) Convergence history: time steps (b) Convergence history: CPU time

Fig. 9. Comparison of convergence history for computing Couette flow, based on quadrilateral fine grid.



(a) Triangular grid (b) Hybrid grid

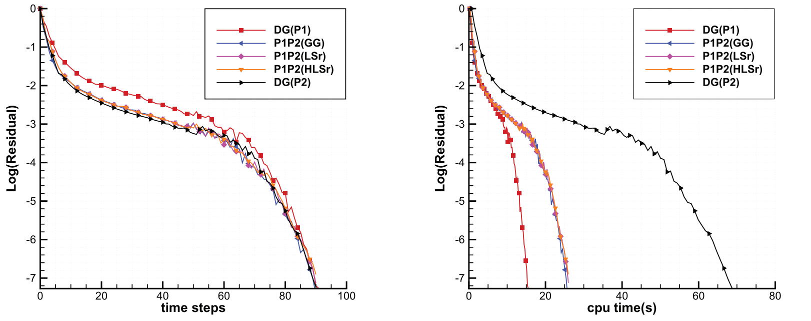
Fig. 10. Two types of grids used for computing a laminar flow over a flat plate.



(a) Triangular grid (b) Zoom in at leading edge

(c) Hybrid grid (d) Zoom in at leading edge

Fig. 11. Comparison of skin friction coefficient among the three reconstructed DG methods for computing a laminar flow over a flat plate.



(a) Convergence history: time steps (b) Convergence history: CPU time

Fig. 12. Comparison of convergence history for computing a laminar flow over a flat plate, based on hybrid grid.

viscosity coefficient $\mu = 0.001$. The computational domain is a rectangle ($0 \le x \le 2H$ and $0 \le y \le H$). Exact boundary condition has been set at the ghost states for all the boundary faces. Two types of three successively refined unstructured grids: triangular and quadrilateral grids, as shown in Fig. 7 are used to assess the accuracy and convergence of the rDG methods to solve the compressible Navier-Stokes equations on unstructured grids. The finest grids are shown in Fig. 7. The difference between the computed density and exact density is used to measure the order of accuracy. The results obtained by the three rDG methods are presented in Table. 4. Fig. 8 provides the details of the spatial accuracy of the DG(P1), DG(P2) and three rDG methods. Fig. 9 compares the convergence histories for the DG(P1), DG(P2), and three rDG methods. It can be seen that the new hybrid rDG method, P1P2(HLSr) is the only one that can deliver the designed 3rd order of accuracy for this experiment. Furthermore, the DG(P1) method provides the best convergence performance and is about 5 times faster than the DG(P2) method. All three rDG(P1P2) methods are much faster than the DG(P2) method, clearly demonstrating the advantage of the rDG(P1P2) method over the DG(P2) method.

5.5. Laminar flow over a flat plate

The laminar boundary layer over an adiabatic flat plate at a free-stream Mach number of 0.2 and a Reynolds number of 100,000 based on the freestream velocity and the length of the flat plate is considered in this test case, where the computational domain is bounded from -0.5 to 1.0 in the x-direction and 0.0 to 1.0 in the y-direction and the flat plate starts at point (0.0,0.0). This problem is chosen to access the accuracy of the different rDG methods for the solution of the compressible Navier-Stokes equations, as the Blasius solution can be used to measure accuracy of the numerical solutions. The unstructured triangular and hybrid grids used in this test case shown in Fig. 10. consist of 900 grid points, 105 boundary points, and 31 grid points on the flat plate. The height of the first element is 0.3464E-03 and 0.82649E-03 at the leading and trailing edge of the flat plate respectively. The skin friction coefficients obtained by the three rDG methods and Blasius solution are presented in Fig. 11. One can observe that the new P1P2(HLSr) method is slightly more accurate than the P1P2(LSr) method, which in turn is much more accurate than the P1P2(GG) method. Fig. 12 shows a comparison of convergence history among the DG(P1), DG(P2), and three rDG methods. As expected, the DG(P1) method provides the best convergence performance and is about 5 times faster than the DG(P2) method. All three rDG(P1P2) methods are much faster than the DG(P2) method, clearly demonstrating the advantage of the rDG(P1P2) method over the DG(P2) method.

6. Conclusions

A new rDG (P1P2) method based on a hybrid least-squares reconstruction and recovery has been developed to solve the compressible Euler and Navier-Stokes equations on arbitrary grids. The new hybrid least-squares reconstruction combines the simplicity of the reconstruction and accuracy of the recovery and has the desired 2-exact property. A number of test cases have been presented to assess the performance of the new hybrid rDG method for a variety of flow problems. Numerical experiments have demonstrated that the newly developed hybrid rDG method is able to attain the designed 3rd order of accuracy for solving both the compresible Euler and Navier-Stokes equations and outperforms the rDG methods based on either Green-Gauss reconstruction or least-squares reconstruction. Future development will be focused on implementation of the new hybrid reconstructed DG method for the 3D problems.

Acknowledgments

The authors would like to acknowledge the partial support provided by China Scholarship Council, the National Natural Science Foundation of China (Nos.91130019 and 10931004), the International Cooperation Project (No.2010DFR00700).

References

[1] Reed WH, Hill TR. Triangular mesh methods for the neutron transport equation. Los Alamos Scientific Laboratory Report, LA-UR-73-479; 1973.
[2] Cockburn B, Hou S, Shu CW. TVD runge-kutta local projection discontinuous galerkin finite element method for conservation laws IV: the multidimensional case. Math Comput 1990;55:545–81.
[3] Cockburn B, Shu CW. The runge-kutta discontinuous galerkin method for conservation laws v: multidimensional system. J Comput Phys 1998;141:199–224.
[4] Cockburn B, Karniadakis G, Shu CW. The development of discontinuous galerkin method, theory, computation, and applications. In: Lecture notes in computational science and engineering, 11. Springer-Verlag; 2000. p. 5–50.
[5] Bassi F, Rebay S. High-order accurate discontinuous finite element solution of the 2d euler equations. J Comput Phys 1997a;138:251–85.
[6] Atkins HL, Shu CW. Quadrature free implementation of discontinuous galerkin method for hyperbolic equations. AIAA J 1998;36:775–82.
[7] Bassi F, Rebay S. GMRES discontinuous galerkin solution of the compressible navier-stokes equations. In: Lecture notes in computational science and engineering, 11. Springer-Verlag; 2000. p. 197–208.
[8] Warburton TC, Karniadakis GE. A discontinuous galerkin method for the viscous MHD equations. J Comput Phys 1999;152:608–41.
[9] Hesthaven JS, Warburton T. Nodal discontinuous Galerkin methods: algorithms, analysis, and applications. Springer Science & Business Media; 2007.
[10] Rasetarinera P, Hussaini MY. An efficient implicit discontinuous spectral galerkin method. J Comput Phys 2001;172:718–38.
[11] Helenbrook BT, Mavriplis D, Atkins HL. Analysis of p-multigrid for continuous and discontinuous finite element discretizations. AIAA 2003:2003–3989.
[12] Fidkowski KJ, Oliver TA, Lu J, Darmofal DL. P-multigrid solution of high-order discontinuous galerkin discretizations of the compressible navier-stokes equations. J Comput Phys 2005;207:92–113.
[13] Luo H, Baum JD, Löhner R. A discontinuous galerkin method using taylor basis for compressible flows on arbitrary grids. J Comput Phys 2008a;227:8875–93.

[14] Luo H, Baum JD, Löhner R. On the computation of steady state compressible flows using discontinuous galerkin method. Int J Numer Meth Eng 2008b;73:597–623.

[15] Luo H, Baum JD, Löhner R. A hermite WENO-based limiter for discontinuous galerkin method on unstructured grids. J Comput Phys 2007;225(1):686–713.

[16] Luo H, Baum JD, Löhner R. A p-multigrid discontinuous galerkin method for the euler equations on unstructured grids. J Comput Phys 2006;211:767–83.

[17] Luo H, Baum JD, Löhner R. A fast, p-multigrid discontinuous galerkin method for compressible flows at all speeds. AIAA Journal 2008c;46:635–52.

[18] Dumbser M, Balsara DS, Toro EF, Munz CD. A unified framework for the construction of one-step finite volume and discontinuous galerkin schemes on unstructured meshes. J Comput Phys 2008;227:8209–53.

[19] Dumbser M. Arbitrary high-order pnpm schemes on unstructured meshes for the compressible navier-stokes equations. Comput Fluids 2010;39:60–76.

[20] Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier-stokes equations. J Comput Phys 1997b;131:267–79.

[21] Bassi F, Rebay S. Discontinuous galerkin solution of the reynolds-averaged navier-stokes and $k-\omega$ turbulence model equations. J Comput Phys 2005;34:507–40.

[22] Cockburn B, Shu CW. The local discontinuous galerkin method for time-dependent convection-diffusion system. SIAM, J Numer Anal 2001;16:2440–63.

[23] Baumann CE, Oden JT. A discontinuous hp finite element method for the euler and navier-stokes equations. Int J Numer Meth Fluids 1999;31:79–95.

[24] Peraire J, Persson PO. The compact discontinuous galerkin method for elliptic problems. SIAM J Sci Comput 2008;30:1806–24.

[25] van Leer B, Nomura S. Discontinuous galerkin method for diffusion. AIAA 2005:2005–5108.

[26] van Leer B, Lo M. A discontinuous galerkin method for diffusion based on recovery. AIAA 2007:2007–4083.

[27] Raalte M, van Leer B. Bilinear forms for the recovery-based discontinuous galerkin method for diffusion. Commun Computat Phys 2009;5:683–93.

[28] Luo H, Luo LQ, Nourgaliev R. A reconstructed discontinuous galerkin method for the compressible euler equations on arbitrary grids. Commun Computat Phys 2012;12:1495–519.

[29] Luo H, Luo LQ, Nourgaliev R, Mousseau VA, Dinh N. A reconstructed discontinuous galerkin method for the compressible navier-stokes equations on arbitrary grids. J Comput Phys 2010;229:6961–78.

[30] Luo H, Luo LQ, Ali A, Nourgaliev R, Cai CP. A parallel, reconstructed discontinuous galerkin method for the compressible flows on arbitrary grids. Commun Computat Phys 2011a;9:363–89.

[31] Zhang LP, Liu W, He LX, Deng XG, Zhang HX. A class of hybrid DG/FV schemes for 2d conservation scalar law. In: The sixth international conference on computational fluid dynamics, St. Petersburg, Russia; 2010. p. 12–16.

[32] Zhang LP, Liu W, He LX, Deng XG, Zhang HX. A class of hybrid DG/FV methods for conservation laws II: two-dimensional cases. J Comput Phys 2012;231:1104–20.

[33] Zhang LP, Liu W, Li M, Zhang HX. A class of hybrid DG/FV methods for conservation laws IV: 2d viscous flows and implicit algorithm for steady cases. Comput Fluids 2014;97:110–25.

[34] Luo H, Baum JD, Löhner R. A fast, matrix-free implicit method for compressible flows on unstructured grids. J Comput Phys 1998;146:664–90.

[35] Luo H, Xiao H, Nourgaliev R, Cai C. A comparative study of different reconstruction schemes for reconstructed discontinuous galerkin methods for the compressible flows on arbitrary grids. AIAA 2011b:2011–3839.

[36] Luo H, Xia YD, Spiegel S, Nourgaliev R, Jiang ZG. A reconstructed discontinuous galerkin method based on a hierarchical WENO reconstruction for compressible flows on tetrahedral grids. J Comput Phys 2013;236:477–92.