A CLASS OF RECONSTRUCTED DISCONTINUOUS GALERKIN METHODS IN COMPUTATIONAL FLUID DYNAMICS

Hong Luo and Yidong Xia
Department of Mechanical and Aerospace Engineering
North Carolina State University
911 Oval Dr. - 3236EBIII, Campus Box 7910
Raleigh, NC 27695, USA
hong_luo@ncsu.edu; yxia2@ncsu.edu

Robert Nourgaliev
Idaho National Laboratory
Idaho Falls, ID 83415, USA
Robert.Nourgaliev@inl.gov

ABSTRACT

A class of reconstructed discontinuous Galerkin (DG) methods is presented to solve compressible flow problems on arbitrary grids. The idea is to combine the efficiency of the reconstruction methods in finite volume methods and the accuracy of the DG methods to obtain a better numerical algorithm in computational fluid dynamics. The beauty of the resulting reconstructed discontinuous Galerkin (RDG) methods is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of the RDG methods, and thus allow for a direct efficiency comparison. Both Green-Gauss and least-squares reconstruction methods and a least-squares recovery method are presented to obtain a quadratic polynomial representation of the underlying linear discontinuous Galerkin solution on each cell via a so-called in-cell reconstruction process. The devised in-cell reconstruction is aimed to augment the accuracy of the discontinuous Galerkin method by increasing the order of the underlying polynomial solution. These three reconstructed discontinuous Galerkin methods are used to compute a variety of compressible flow problems on arbitrary meshes to assess their accuracy. The numerical experiments demonstrate that all three reconstructed discontinuous Galerkin methods can significantly improve the accuracy of the underlying second-order DG method, although the least-squares reconstructed DG method provides the best performance in terms of both accuracy, efficiency, and robustness.

Key Words: Discontinuous Galerkin Methods, Reconstruction methods, Computational fluid dynamics, Compressible flows.

1. INTRODUCTION

The discontinuous Galerkin methods \(^1\textsuperscript{1-25}\) (DGM) 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 discontinuous Galerkin methods combine two advantageous features commonly associated to finite element and finite volume methods. As in classical finite element methods, accuracy is obtained by means of high-order polynomial approximation within an element rather than by wide stencils as in the case of
finite volume 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 finite volume methods. The discontinuous Galerkin methods have many attractive features: 1) They have several useful mathematical properties with respect to conservation, stability, and convergence; 2) The method can be easily extended to higher-order (>2nd) approximation; 3) The methods 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. The compactness also allows for structured and simplified coding for the methods; 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 finite volume methods. In contrast to the enormous advances in the theoretical and numerical analysis of the DGM, the development of a viable, attractive, competitive, and ultimately superior DG method over the more mature and well-established second order methods is relatively an untouched area. This is mainly due to the fact that the DGM have a number of weaknesses that have yet to be addressed, before they can be robustly used to flow problems of practical interest in a complex configuration environment. In particular, there are three most challenging and unresolved issues in the DGM: a) how to efficiently discretize diffusion terms required for the Navier-Stokes equations, b) how to effectively control spurious oscillations in the presence of strong discontinuities, and c) how to develop efficient time integration schemes for time accurate and steady-state solutions. Indeed, compared to the finite element methods and finite volume methods, the DG methods require solutions of systems of equations with more unknowns for the same grids. Consequently, these methods have been recognized as expensive in terms of both computational costs and storage requirements.

Dumbser et al\textsuperscript{18-20} have originally introduced a new family of reconstructed DG methods, termed PnPm schemes, where Pn indicates that a piecewise polynomial of degree n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m (m ≥ n) that is used to compute the fluxes. The beauty of PnPm schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of PnPm 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, P0Pm is nothing but classical high order finite volume schemes, where a polynomial solution of degree m (m ≥ 1) is reconstructed from a piecewise constant solution. When m=n, the reconstruction reduces to the identity operator, and PnPn scheme yields a standard DG method.

Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the PnPm 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., 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 finite volume methods provides an alternative, probably a better choice to obtain a higher-order polynomial representation. Luo et al. develop a reconstructed discontinuous Galerkin method using a Taylor basis\cite{5,35,36,37} 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.

More recently, Zhang et al.\cite{38,39} 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 finite volume methods. This provides a fast, simple, and robust way to obtain a higher-order polynomial solutions. The numerical experiments indicate that this efficient reconstruction scheme is able to achieve a third-order accuracy: one order accuracy higher than the underlying second order DG method.

The objective of the effort discussed in this paper is to develop a class of reconstructed discontinuous Galerkin methods for solving compressible flow problems on arbitrary grids. Both Green-Gauss and least-squares reconstruction methods and a least-squares recovery method are presented to obtain a quadratic polynomial representation from a underlying linear discontinuous Galerkin solution. A comparative study on these three reconstruction methods are conducted for a variety of test cases to access their accuracy for computing compressible flows on arbitrary grids. The numerical experiments demonstrate that all three reconstruction methods can significantly improve the accuracy of the underlying second-order DG method, although the least-squares reconstruction method provides the best performance in terms of both accuracy and efficiency. The remainder of this paper is organized as follows. The governing equations are described in Section 2. The underlying reconstructed discontinuous Galerkin methods are presented in Section 3. Extensive numerical experiments are reported in Section 4. Concluding remarks are given in Section 5.
2. GOVERNING EQUATIONS

The Euler equations governing unsteady compressible inviscid flows can be expressed as

\[
\frac{\partial \mathbf{U}(x,t)}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{U}(x,t))}{\partial x_k} = 0
\]  

(2.1)

where the summation convention has been used. The conservative variable vector \( \mathbf{U} \), and inviscid flux vector \( \mathbf{F} \) are defined by

\[
\mathbf{U} = \begin{pmatrix} \rho \\ \rho u_i \\ \rho e \end{pmatrix} \quad \mathbf{F}_j = \begin{pmatrix} \rho u_j \\ \rho u_i u_j + p \delta_{ij} \\ u_j (\rho e + p) \end{pmatrix}
\]  

(2.2)

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

\[
p = (\gamma - 1)\rho(e - \frac{1}{2}u_j u_j)
\]  

(2.3)

which is valid for perfect gas, where \( \gamma \) is the ratio of the specific heats.

3. RECONSTRUCTED DISCONTINUOUS GALERKIN METHODS

The governing equation (2.1) is discretized using a discontinuous Galerkin finite element formulation. To formulate the discontinuous Galerkin method, we first introduce the following weak formulation, which is obtained by multiplying the above conservation law by a test function \( W \), integrating over the domain \( \Omega \), and then performing an integration by parts,

\[
\int_{\Omega} \frac{\partial \mathbf{U}}{\partial t} \mathbf{W} d\Omega + \int_{\Omega} \mathbf{F} \mathbf{n} d\Gamma - \int_{\Omega} \mathbf{F}_k \frac{\partial \mathbf{W}}{\partial x_k} d\Omega = 0, \quad \forall \mathbf{W} \in \mathbf{V}
\]  

(3.1)

where \( \Gamma(=\partial \Omega) \) denotes the boundary of \( \Omega \), and \( \mathbf{n} \) the unit outward normal vector to the boundary. We assume that the domain \( \Omega \) is subdivided into a collection of non-overlapping elements \( \Omega_e \), which can be triangles, quadrilaterals, polygons, or their combinations in 2D and tetrahedra, prisms, pyramids, and hexahedra or their combinations in 3D. We introduce the following broken Sobolev space \( \mathbf{V}_h^p \)

\[
\mathbf{V}_h^p = \left\{ \mathbf{v}_h \in [L^2(\Omega)]^m : \mathbf{v}_h|_{\Omega_e} \in \left[ \mathbf{V}_p^m \right]_\Omega \right\}
\]  

(3.2)

which consists of discontinuous vector-values polynomial functions of degree \( p \), and where \( m \) is the dimension of the unknown vector and

\[
\mathbf{V}_p^m = \text{span} \left\{ \prod_{i=1}^d x_i^{\alpha_i} : 0 \leq \alpha_i \leq p, 0 \leq i \leq d \right\}
\]  

(3.3)

where \( \alpha \) denotes a multi-index and \( d \) is the dimension of space. Then, we can obtain the following semi-discrete form by applying weak formulation on each element \( \Omega_e \).
Find \( U_h \in V_h^p \) such as

\[
\frac{d}{dt} \int_{\Omega_e} U_h W_h d\Omega + \frac{1}{t_e} \int_{\Omega_e} F_k (U_h) n_k W_h d\Gamma - \frac{1}{t_e} \int_{\Omega_e} F_k (U_h) \frac{\partial W_h}{\partial x_k} d\Omega = 0, \quad \forall W_h \in V_h^p
\]

(3.4)

where \( U_h \) and \( W_h \) represent the finite element approximations to the analytical solution \( U \) and the test function \( W \) respectively, and they are approximated by a piecewise polynomial function of degrees \( p \), which are discontinuous between the cell interfaces. Assume that \( B \) is the basis of polynomial function of degrees \( p \), this is then equivalent to the following system of \( N \) equations,

\[
\frac{d}{dt} \int_{\Omega_e} U_h B_i d\Omega + \frac{1}{t_e} \int_{\Omega_e} F_k (U_h) n_k B_i d\Gamma - \frac{1}{t_e} \int_{\Omega_e} F_k (U_h) \frac{\partial B_i}{\partial x_k} d\Omega = 0, \quad 1 \leq i \leq N
\]

(3.5)

where \( N \) is the dimension of the polynomial space. Since the numerical solution \( U_h \) is discontinuous between element interfaces, the interface fluxes are not uniquely defined. The flux function \( F_k(U_h)n_k \) appearing in the second terms of Eq. (3.5) is replaced by a numerical Riemann flux function \( H_k(U_{hL}, U_{hR}, n) \) where \( U_{hL} \) and \( U_{hR} \) are the conservative state vector at the left and right side of the element boundary. This scheme is called discontinuous Galerkin method of degree \( p \), or in short notation DG(P) method. Note that discontinuous Galerkin formulations are very similar to finite volume schemes, especially in their use of numerical fluxes. Indeed, the classical first-order cell-centered finite volume scheme exactly corresponds to the DG(P\( \infty \)) method, i.e., to the discontinuous Galerkin method using a piecewise constant polynomial. Consequently, the DG(P\( \infty \)) methods with \( k > 0 \) can be regarded as a natural generalization of finite volume methods to higher order methods. By simply increasing the degree \( P \) of the polynomials, the DG methods of corresponding higher order are obtained.

In the traditional DGM, numerical polynomial solutions \( U_h \) in each element are expressed using either standard Lagrange finite element or hierarchical node-based basis as following

\[
U_h = \sum_{i=1}^{N} U_i(t)B_i(x), \tag{3.6}
\]

where \( B_i \) are the finite element basis functions. As a result, the unknowns to be solved are the variables at the nodes \( U_i \), as illustrated in Figure 1 for linear and quadratic polynomial approximations.

**Figure 1. Representation of polynomial solutions using finite element shape functions**

On each cell, a system of \( N \times N \) has to be solved, where polynomial solutions are dependent on the shape of elements. For example, for a linear polynomial approximation in 2D as shown in Fig. 1, a linear polynomial is used for triangular elements and the unknowns to be solved are the
variables at the three vertices and a bi-linear polynomial is used for quadrilateral elements and the unknowns to be solved are the variables at the four vertices. However, numerical polynomial solutions $U$ can be expressed in other forms as well. In the present work, the numerical polynomial solutions are represented using a Taylor series expansion at the center of the cell. For example, if we do a Taylor series expansion at the cell centroid, the quadratic polynomial solutions can be expressed as follows

$$U_h = U_c + \frac{\partial U}{\partial x} \bigg|_c (x - x_c) + \frac{\partial U}{\partial y} \bigg|_c (y - y_c) + \frac{\partial^2 U}{\partial x^2} \bigg|_c \frac{(x - x_c)^2}{2}$$

$$+ \frac{\partial^2 U}{\partial y^2} \bigg|_c \frac{(y - y_c)^2}{2} + \frac{\partial^2 U}{\partial x \partial y} \bigg|_c (x - x_c)(y - y_c)$$

which can be further expressed as cell-averaged values and their derivatives at the center of the cell:

$$U_h = \bar{U} + \frac{\partial U}{\partial x} \bigg|_c (x - x_c) + \frac{\partial U}{\partial y} \bigg|_c (y - y_c) + \frac{\partial^2 U}{\partial x^2} \bigg|_c \frac{(x - x_c)^2}{2} - \frac{1}{\Omega_c} \int_{\Omega_c} (x - x_c)^2 d\Omega$$

$$+ \frac{\partial^2 U}{\partial y^2} \bigg|_c \frac{(y - y_c)^2}{2} - \frac{1}{\Omega_c} \int_{\Omega_c} (y - y_c)^2 d\Omega + \frac{\partial^2 U}{\partial x \partial y} \bigg|_c ((x - x_c)(y - y_c) - \frac{1}{\Omega_c} \int_{\Omega_c} (x - x_c)(y - y_c) d\Omega)$$

where $\bar{U}$ is the mean value of $U$ in this cell. 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, as shown in Fig.2.

**Figure 2. Representation of polynomial solutions using a Taylor series expansion for a cell-centered scheme (left) and vertex-centered scheme (right)**

In this case, the dimension of the polynomial space is six and the six basis functions are

$$B_1 = 1 \quad B_2 = x - x_c \quad B_3 = y - y_c \quad B_4 = \frac{(x - x_c)^2}{2} - \frac{1}{\Omega_c} \int_{\Omega_c} (x - x_c)^2 d\Omega$$

$$B_5 = \frac{(y - y_c)^2}{2} - \frac{1}{\Omega_c} \int_{\Omega_c} (y - y_c)^2 d\Omega \quad B_6 = (x - x_c)(y - y_c) - \frac{1}{\Omega_c} \int_{\Omega_c} (x - x_c)(y - y_c) d\Omega$$

The discontinuous Galerkin formulation then leads to the following six equations

$$\frac{d}{dt} \int_{\Omega_c} \bar{U} d\Omega + \int_{\Gamma_e} F_k(U_h) n_k d\Gamma = 0, \quad i = 1$$

$$M_{5x5} \frac{d}{dt} \left( \frac{\partial U}{\partial x} \bigg|_c \frac{\partial U}{\partial y} \bigg|_c \frac{\partial^2 U}{\partial x^2} \bigg|_c \frac{\partial^2 U}{\partial y^2} \bigg|_c \frac{\partial^2 U}{\partial x \partial y} \bigg|_c \right)^T + R_{5x1} = 0$$

Note that in this formulation, equations for the cell-averaged variables are decoupled from equations for their derivatives due to the judicial choice of the basis functions and the fact that

\[ \int_{\Omega_e} B_i B_j d\Omega = 0, \quad 2 \leq i \leq 6 \]  

(3.11)

In the implementation of this DG method, the basis functions are actually normalized in order to improve the conditioning of the system matrix (3.5) as follows:

\[
\begin{align*}
B_1 &= 1, & B_2 &= \frac{x-x_c}{\Delta x}, & B_3 &= \frac{y-y_c}{\Delta y}, & B_4 &= \frac{(x-x_c)^2 - 1}{2\Delta x^2} d\Omega \\
B_5 &= \frac{(y-y_c)^2 - 1}{2\Delta y^2} d\Omega, & B_6 &= \frac{(x-x_c)(y-y_c)}{\Delta x\Delta y} \int d\Omega.
\end{align*}
\]

(3.12)

where \(\Delta x = 0.5(x_{\text{max}} - x_{\text{min}})\), and \(\Delta y = 0.5(y_{\text{max}} - y_{\text{min}})\), and \(x_{\text{max}}, x_{\text{min}}, y_{\text{max}},\) and \(y_{\text{min}}\) are the maximum and minimum coordinates in the cell \(\Omega_e\) in \(x\)-, and \(y\)-directions, respectively. A quadratic polynomial solution can then be rewritten as

\[ U_h = \tilde{U} + \frac{\partial U}{\partial x} \Delta x B_2 + \frac{\partial U}{\partial y} \Delta y B_3 + \frac{\partial^2 U}{\partial x^2} \Delta x^2 B_4 + \frac{\partial^2 U}{\partial y^2} \Delta y^2 B_5 + \frac{\partial^2 U}{\partial x\partial y} \Delta x\Delta y B_6 \]  

(3.13)

The above normalization is especially important to alleviate the stiffness of the system matrix for higher-order DG approximations.

This formulation allows us to clearly see the similarities and differences between the DG and FV methods. In fact, the discretized governing equations for the cell-averaged variables and the assumption of polynomial solutions on each cell are exactly the same for both methods. The only difference between them is the way how they obtain high-order (>1) polynomial solutions. In the finite volume methods, the polynomial solution of degrees \(p\) are reconstructed using information from the cell-averaged values of the flow variables, which can be obtained using either TVD/MUSCL or ENO/WENO reconstruction schemes. Unfortunately, the multi-dimensional MUSCL approach suffers from two shortcomings in the context of unstructured grids: 1) Uncertainty and arbitrariness in choosing the stencils and methods to compute the gradients in the case of linear reconstruction; This explains why a nominally second-order finite volume scheme is hardly able to deliver a formal solution of the second order accuracy in practice for unstructured grids. The situation becomes even more evident, severe, and profound, when a highly stretched tetrahedral grid is used in the boundary layers. Many studies, as reported by many researchers\(^{26-28}\) have demonstrated that it is difficult to obtain a second-order accurate flux reconstruction on highly stretched tetrahedral grids and that for the discretization of inviscid fluxes, the classic 1D-based upwind schemes using median-dual finite volume approximation suffer from excessive numerical diffusion due to such skewing. 2) Extended stencils required for the reconstruction of higher-order (>1\(^3\)) polynomial solutions. This is exactly the reason why the current finite-volume methods using the TVD/MUSCL reconstruction are not practical at higher order and have remained second-order on unstructured grids. When the ENO/WENO reconstruction schemes are used for the construction of a polynomial of degree \(p\) on unstructured grids, the dimension of the polynomial space \(N = N(p,d)\) depends on the degree of the polynomials of the expansion \(p\), and the number of spatial dimensions \(d\). One must have three, six, and ten cells in 2D and four, ten, and twenty cells in 3D for the construction of a linear,
quadratic, cubic Lagrange polynomial, respectively. Undoubtedly, it is an overwhelmingly challenging, if not practically impossible, task to judiciously choose a set of admissible and proper stencils that have such a large number of cells on unstructured grids especially for higher order polynomials and higher dimensions. This explains why the application of higher-order ENO/WENO methods hardly exists on unstructured grids, in spite of their tremendous success on structured grids and their superior performance over the MUSCL/TVD methods. Unlike the FV methods, where the derivatives are reconstructed using cell average values of the neighboring cells, the DG method computes the derivatives in a manner similar to the mean variables. This is compact, rigorous, and elegant mathematically in contrast with arbitrariness characterizing the reconstruction schemes with respect how to compute the derivatives and how to choose the stencils used in the FV methods. It is our believe that this is one of the main reasons why the second order DG methods are more accurate than the FV methods using either TVD/MUSCL or ENO/WENO reconstruction schemes and are less dependent on the mesh regularity, which has been demonstrated numerically. Furthermore, the higher order DG methods can be easily constructed by simply increasing the degree p of the polynomials locally, in contrast to the finite volume methods which use the extended stencils to achieve higher order of accuracy.

However, in comparison with reconstructed FV methods, the DG methods have a significant drawback in that they require more degrees of freedom, an additional domain integration, and more Gauss quadrature points for the boundary integration, and therefore more computational costs and storage requirements. On one hand, the reconstruction methods that FV methods use to achieve higher-order accuracy are relatively inexpensive but less accurate and robust. One the other hand, DG methods that can be viewed as a 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. This idea was originally introduced by Dumbser et al in the frame of PnPm scheme, where Pn indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m (m ≥ n) that is used to compute the fluxes and source terms. The beauty of PnPm schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of PnPm 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, P0Pm is nothing but classical high order finite volume schemes, where a polynomial solution of degree m (m ≥ 1) is reconstructed from a piecewise constant solution. When m=n, the reconstruction reduces to the identity operator, and PnPm scheme yields a standard DG method. Clearly, an accurate and efficient reconstruction is the key ingredient in extending the underlying DG method to higher order accuracy. Although our discussion in this work is mainly focused on the linear DG method, its extension to higher order DG methods is straightforward. In the case of DG(P1) method, a linear polynomial solution $U_i$ in any cell i is

$$U_i = \tilde{U}_i + \frac{\partial U}{\partial x} |\Delta x B_2 + \frac{\partial U}{\partial y} |\Delta y B_3$$

Using this underlying linear polynomial DG solution in the neighboring cells, one can reconstruct a quadratic polynomial solution $U^R_i$ as follows:

$$U^R_i = \tilde{U}^R_i + \frac{\partial U^R}{\partial x} |\Delta x B_2 + \frac{\partial^2 U^R}{\partial y} |\Delta y B_3 + \frac{\partial^2 U^R}{\partial x^2} |\Delta x^2 B_2 + \frac{\partial^2 U^R}{\partial y^2} |\Delta y^2 B_3 + \frac{\partial^2 U^R}{\partial x \partial y} |\Delta x \Delta y B_6$$

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 face with the cell $i$ under consideration. There are six degrees of freedom, and therefore 6 unknowns to be determined. However, the first three unknowns can be trivially obtained, by requiring that the reconstruction scheme has to be conservative, a fundamental requirement, and the values of the reconstructed first derivatives are equal to the ones of the first derivatives of the underlying DG solution at the centroid $i$. Due to the judicious choice of Taylor basis in our DG formulation, these three degrees of freedom (cell average and slopes) simply coincide with the ones from the underlying DG solution, i.e.,

$$
\tilde{U}_i^R = \tilde{U}_i, \quad \frac{\partial U^R}{\partial x} \bigg|_i = \frac{\partial U}{\partial x} \bigg|_{i}, \quad \frac{\partial U^R}{\partial y} \bigg|_i = \frac{\partial U}{\partial y} \bigg|_{i}.
$$

Three methods of determining the remaining three degrees of freedom: the second derivatives (curvatures) will be addressed and discussed in this section.

**A. Least-Squares Recovery**

The least-square recovery scheme by Dumbser et al.\textsuperscript{18-20} relies on $L_2$-projection to determine the second derivatives. Consider a face-neighboring cell $j$, the recovery principle requires

$$
\int_{L_j} U_i^R B_k^j d\Omega = \int_{L_j} U_j B_k^j d\Omega, \quad 1 \leq k \leq 3
$$

(3.17)

where $B_k^j$ ($k=1,2,3$) are the three basis function on cell $j$. 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. (3.17) can be written for all cells connected to the cell $i$ 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 $9 \times 3$ and $12 \times 3$, 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 exist six second derivatives in 3D. In this case, one cannot recover a quadratic polynomial solution from the underlying linear discontinuous Galerkin solution. This situation can be readily remedied by use of extended one-sided stencils, although the compactness of the underlying DG methods is then sacrificed. The resulting reconstructed DG method will be termed as P1P2(rc) method in this paper.

**B. 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. Consider a neighboring cell $j$, one requires

$$
\mathbf{U}_j = \mathbf{U}_i + \frac{\partial \mathbf{U}^R}{\partial x} \Delta x_i \mathbf{B}_2 + \frac{\partial \mathbf{U}^R}{\partial y} \Delta y_i \mathbf{B}_3 + \frac{\partial^2 \mathbf{U}^R}{\partial x^2} \Delta x_i^2 \mathbf{B}_4 + \frac{\partial^2 \mathbf{U}^R}{\partial y^2} \Delta y_i^2 \mathbf{B}_5 + \frac{\partial^2 \mathbf{U}^R}{\partial x \partial y} \Delta x_i \Delta y_i \mathbf{B}_6
$$

(3.18)

$$
\frac{\partial \mathbf{U}}{\partial x} = \frac{\partial \mathbf{U}^R}{\partial x} \Delta x_i \mathbf{B}_2 + \frac{\partial^2 \mathbf{U}^R}{\partial x^2} \Delta x_i^2 \mathbf{B}_4 + \frac{\partial^2 \mathbf{U}^R}{\partial x \partial y} \Delta x_i \Delta y_i \mathbf{B}_6
$$

$$
\frac{\partial \mathbf{U}}{\partial y} = \frac{\partial \mathbf{U}^R}{\partial y} \Delta y_i \mathbf{B}_3 + \frac{\partial^2 \mathbf{U}^R}{\partial y^2} \Delta y_i^2 \mathbf{B}_5 + \frac{\partial^2 \mathbf{U}^R}{\partial x \partial y} \Delta x_i \Delta y_i \mathbf{B}_6
$$

where the basis function $B$ are evaluated at the center of cell $j$, i.e., $B=B(x,y)$. This can be written in a matrix form as follows:

$$
\begin{bmatrix}
B_1^j & B_2^j & B_3^j \\
B_2^j & 0 & B_3^j \\
0 & B_3^j & B_2^j
\end{bmatrix}
\begin{bmatrix}
\frac{\partial^2 \mathbf{U}^R}{\partial x^2} \Delta x_i^2 \\
\frac{\partial^2 \mathbf{U}^R}{\partial y^2} \Delta y_i^2 \\
\frac{\partial^2 \mathbf{U}^R}{\partial x \partial y} \Delta x_i \Delta y_i
\end{bmatrix}
= \mathbf{U}_j - \left( \mathbf{U}_i B_1^j + \frac{\partial \mathbf{U}^R}{\partial x} \Delta x_i B_2^j + \frac{\partial \mathbf{U}^R}{\partial y} \Delta y_i B_3^j \right)
= \begin{bmatrix}
\mathbf{R}_1^j \\
\mathbf{R}_2^j \\
\mathbf{R}_3^j
\end{bmatrix}
$$

(3.19)

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 $i$ with a common face, which leads to a non-square matrix. The number of face-neighbor cells for a triangular and quadrilateral cell is three and four, respectively. As a result, the size of the resulting non-square matrix is 9x3 and 12x3, 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 3x3 as follows

$$
\begin{bmatrix}
\sum_j (B_1^j B_1^j + B_2^j B_4^j) & \sum_j B_4^j B_5^j & \sum_j (B_1^j B_6^j + B_2^j B_4^j) \\
\sum_j B_2^j B_5^j & \sum_j (B_2^j B_7^j + B_3^j B_5^j) & \sum_j (B_3^j B_6^j + B_2^j B_7^j) \\
\sum_j (B_4^j B_6^j + B_2^j B_3^j) & \sum_j (B_6^j B_7^j + B_3^j B_5^j) & \sum_j (B_6^j B_8^j + B_2^j B_7^j + B_3^j B_9^j)
\end{bmatrix}
\begin{bmatrix}
\frac{\partial^2 \mathbf{U}^R}{\partial x^2} \Delta x_i^2 \\
\frac{\partial^2 \mathbf{U}^R}{\partial y^2} \Delta y_i^2 \\
\frac{\partial^2 \mathbf{U}^R}{\partial x \partial y} \Delta x_i \Delta y_i
\end{bmatrix}
= \begin{bmatrix}
\sum_j (B_1^j \mathbf{R}_1^j + B_2^j \mathbf{R}_2^j) \\
\sum_j (B_3^j \mathbf{R}_1^j + B_4^j \mathbf{R}_3^j) \\
\sum_j (B_6^j \mathbf{R}_1^j + B_7^j \mathbf{R}_2^j + B_8^j \mathbf{R}_3^j)
\end{bmatrix}
$$

(3.20)
This linear system of 3x3 can be then trivially solved to obtain the second derivatives of the reconstructed quadratic polynomial solution. This reconstructed DG method will be referred as P1P2(RC) method from now on.

**C. Green-Gauss Reconstruction**

Green-Gauss reconstruction is probably the simplest reconstruction scheme, which is mostly used to reconstruct a gradient from the cell-averaged values in the finite volume methods. Similarly, the second derivatives in a cell $i$ can be reconstructed from the known first derivatives using Green's theorem as follows,

$$
\int_{\Omega_i} \frac{\partial^2 U}{\partial x^2} \, d\Omega = \Omega_i \left. \frac{\partial^2 U}{\partial x^2} \right|_{\Gamma_i} \int_{\Gamma_i} \frac{\partial U}{\partial x} n_x \, d\Gamma \tag{3.21}
$$

$$
\int_{\Omega_i} \frac{\partial^2 U}{\partial y^2} \, d\Omega = \Omega_i \left. \frac{\partial^2 U}{\partial y^2} \right|_{\Gamma_i} \int_{\Gamma_i} \frac{\partial U}{\partial y} n_y \, d\Gamma \tag{3.22}
$$

$$
\int_{\Omega_i} \frac{\partial^2 U}{\partial x \partial y} \, d\Omega = \Omega_i \left. \frac{\partial^2 U}{\partial x \partial y} \right|_{\Gamma_i} \int_{\Gamma_i} \frac{\partial U}{\partial x} n_x \, d\Gamma \tag{3.23}
$$

$$
\int_{\Omega_i} \frac{\partial^2 U}{\partial y \partial x} \, d\Omega = \Omega_i \left. \frac{\partial^2 U}{\partial y \partial x} \right|_{\Gamma_i} \int_{\Gamma_i} \frac{\partial U}{\partial y} n_y \, d\Gamma \tag{3.24}
$$

The cross-derivatives can be computed using either equation 3.23 or equation 3.24. In the present work, an arithmetic mean is used to obtain the cross-derivatives. In Eqs. 3.22, 3.23, 3.24, and 3.25, the first derivatives at the interface are computed using a simple volume-weighted average. The beauty of this reconstruction scheme is its simplicity, efficiency, and robustness, since it does not need to solve an over-determined system using a least-squares approach. However, the Green-Gauss reconstruction is not as accurate as its least-squares recovery/reconstruction counterparts, as it only involves the first derivatives of the underlying DG solutions. This reconstructed DG method is named P1P2(GG) hereafter.

In the reconstructed DG methods, the reconstructed quadratic polynomial solution is then used to compute the domain and boundary integrals of the underlying DG(P1) method in Eq. (3.5). The resulting DG method, termed a reconstructed DG method (RDG(P1P2) or P1P2 in short notation), is expected to have a third order of accuracy at a moderate increase of computing costs in comparison with the underlying DG(P1) method. The extra costs are mainly due to the reconstruction step, which is relatively cheap in comparison to the evaluation of fluxes, and an extra Gauss quadrature point, which is required to calculate the domain integrals for the triangular element (four quadrature points). Like DG(P1) methods, two quadrature points are used to calculate the boundary integrals, and four points are used to calculate the domain integrals for quadrilateral elements. In comparison to DG(P2) methods, this represents a significant savings in terms of flux evaluations. Furthermore, the number of degrees of freedom
is significantly reduced, which leads to a significant reduction in memory requirements, and from which implicit methods will benefit tremendously.

4. NUMERICAL EXAMPLES

A. Convection of an isentropic vortex

The convection of a 2D inviscid isentropic given for example by Dumbser et al\textsuperscript{18} is considered in this test case to conduct a convergence study of the RDG methods. The analytical solution to this problem at any time \( t \) is simply the passive advection of the initial solution at \( t=0 \), which provides a valuable reference for measuring the accuracy of a numerical solution. The initial condition is a linear superposition of a mean uniform flow with some perturbations \( \delta \). The free stream flow conditions are \((\rho_{\infty}, u_{\infty}, v_{\infty}, p_{\infty}) = (1,1,1,1)\). In this test case, the vortex strength \( \varepsilon=5 \), and the coordinate of the vortex center \((x_0,y_0)\) is \((5,5)\). The computational domain \( \Omega \) is \([0,10]\times[0,10]\) and the periodic boundary conditions are imposed. The numerical solutions are obtained after one period of time \( t=10 \), and compared with the exact solution simply given by the initial condition. The follow \( L_2 \)-norm

\[
\|\rho^R - \rho^0\|_{L^2(\Omega)} = \left( \int_{\Omega} \|\rho^R - \rho^0\|^2 \, d\Omega \right)^{\frac{1}{2}},
\]

is used to measure the error between the numerical and analytical solutions, where \( \rho^R \) is the reconstructed quadratic solution for the density. Figure 3 shows three successively refined triangular grids having 554, 2,216, and 8,864 triangles, respectively. The number of faces in each space dimension is 16, 32, and 64, respectively. Figure 4 provides the details of the spatial convergence of the three RDG methods for this numerical experiment. As expected, all three reconstructed (P1P2) methods offer a full \( O(h^{p+2}) \) order of the convergence, adding one order of accuracy to the underlying DG(P1) method.

Figure 3. Sequence of triangular grids used for computing the convection of an isentropic vortex.
This is a well-known test case: subsonic flow past a circular cylinder at a Mach number of $M_\infty=0.38$. A set of three hybrid grids is used in this test case to verify if a formal order of the convergence rate of the three RDG methods can be achieved for the compressible Euler equations on hybrid grids. Figure 5 shows three successively refined hybrid grids having 32x9, 64x17, and 128x33 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 radius of the cylinder is $r_1=0.5$, the domain is bounded by $r_{33}=20$, and the radii of concentric circles for 128x33 mesh are set up as

$$n_i = n_1 \left( 1 + \frac{2\alpha}{128} \sum_{j=0}^{i-1} \alpha^j \right), \quad i = 2, ..., 33$$

where $\alpha=1.1580372$. The coarser grids are generated by successively coarsing the finest mesh. Numerical solutions to this problem are computed using the three reconstruction RDG(P1P2) methods on these three grids to obtain quantitative measurement of the order of accuracy and discretization errors. In this case, the $L_2$-norm of the following entropy production $\varepsilon$ defined as

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

is served as the error measurement, where $S$ is the entropy. Note that the entropy production is a very good criterion to measure accuracy of the numerical solutions, since the flow under consideration is isentropic. Figures 6 and 7 show the computed density contours in the flow field using these three grids obtained by DG(P1) and P1P2(RC) method, respectively, where one can clearly observe the significant improvement of the computed solution using the RDG method. Figure 8 provides the details of the spatial convergence of each reconstruction method for this numerical experiment. As expected, all reconstructed RDG(P1P2) methods offer a full $O(h^{p+2})$. 

Figure 4. Comparison of the convergence histories for different RDG method for the convection of an isentropic vortex.
order of the convergence, adding one order of accuracy to the underlying DG(P1) method. However, the least-squares reconstruction and recovery methods are better than the Green-Gauss reconstruction method in terms of both absolute error and order of convergence, and the P1P2(RC) method outperforms P1P2(rc) method in this case.

Figure 5: Sequences of three successively globally refined hybrid meshes 32x9, 64x17, 128x33 for computing subsonic flow past a circular cylinder.

Figure 6. Computed density contours in the flow field obtained by the DG(P1) method on 32x9 mesh (left), 64x17 mesh (middle), and 128x33 mesh (right) for subsonic flow past a circular cylinder at $M_{\infty}=0.38$.

Figure 7. Computed density contours in the flow field obtained by the P1P2(RC) method on 32x9 mesh (left), 64x17 mesh (middle), and 128x33 mesh (right) for subsonic flow past a circular cylinder at $M_{\infty}=0.38$. 
Figure 8. Convergence history for the convection of an isentropic vortex for different RDG methods.

C. Inviscid Flow through a Channel with a Smooth Bump

Figure 9: Sequences of three successively globally refined unstructured meshes used for computing subsonic flow in a channel with a smooth bump.
Figure 10. Computed velocity contours in the flow field obtained by the DG(P1) method (left) and least-squares reconstructed RDG(P1P2) method (right) on the coarse mesh (top), medium mesh (middle), and fine mesh (right) for a subsonic flow through a channel with a bump on the lower surface at $M_{\infty}=0.5$.

Figure 11. Convergence history for the convection of an isentropic vortex for different reconstructed RDG methods.

This test case is chosen to access the accuracy and ability of the RDG methods for computing internal flows. The problem under consideration is a subsonic flow through a channel with a smooth bump. The computational domain is bounded between -1.5 and 1.5 in the x-coordinate and the bump defined by $0.0625\exp(-25x^2)$ and 0.8 in the y-coordinate. The inflow condition is at a Mach number of 0.5, and an angle of attack of $0^\circ$. Figure 9 shows the three successively refined triangular grids used for conducting the grid convergence study, having 127, 508, and 2,032 triangles, respectively. Figure 10 illustrates the computed velocity contours in the flow field obtained by DG(P1), P1P2(GG), and P1P2(RC) methods, respectively, where one can clearly observe the significant improvement of the computed solution using the RDG method. Figure 11 provides the details of the spatial convergence of the two reconstruction-based DG methods for this numerical experiment. Again, for this internal flow problem, all reconstructed 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. Note that the least-squares recovery DG method P1P2(rc) is unable to provide a stable solution for this test case, which is attributed to the lack of sufficient recovered equations for some of boundary cells.
D. Subsonic Flows past a NACA0012 airfoil

Figure 12. Unstructured hybrid mesh (top left) (nquad=1,533, ntria=3,469, npoin=3,346, nbfac=157) and computed Mach number contours by the DG(P1) (top right), the P1P1(GG) (bottom left), and the P1P2(RC)(bottom right) methods, respectively for subsonic flow past a NACA0012 airfoil at $M_{\infty} = 0.63, \alpha = 2^\circ$.

Our last test case involves an inviscid flow past a NACA0012 airfoil at a Mach number of 0.63, and an angle of attack $2^\circ$. This numerical experiment is designed to test the ability of the RDG method for obtaining a highly accurate solution to the Euler equations on a viscous hybrid grid. Being able to produce a highly accurate inviscid solution on a highly stretched Navier-Stokes grid is extremely difficult and challenging, and yet of utmost importance for the accurate solution to the Navier-Stokes equations, thus serving as a good criterion to measure accuracy and robustness of a numerical method. Many finite volume methods are unable to obtain the same quality of an inviscid solution on an anisotropic Navier-Stokes grid as on an isotropic Euler grid, suffering from either excessive numerical dissipation or spurious oscillations due to a combination of a mesh irregularity and reconstruction schemes. Figure 12 shows the mesh used in this numerical experiment and the computed Mach number contours in the flow field obtained by DG(P1), P1P2(GG), and P1P2(RC) methods, respectively. The mesh consists of 1,533 quadrilateral cells, 3,469 triangular cells, 3,346 grid points, and 157 boundary faces. The computed Mach number contours in the flow field obtained using DG(P1), P1P1(GG), and
P1P2(RC) methods are shown in Fig. 12, where accurate and smooth solutions are observed in spite of the highly stretched grid used in the boundary layer. The computed pressure coefficient and entropy production distribution on the surface of the airfoil obtained by these three methods are compared in Fig. 12. All three solutions are virtually identical by judging the Mach number contours in the flow field and the pressure coefficient distributions on the surface of the airfoil, indicating that the numerical solution is order-independent, (i.e., a convergence on these flow quantities is reached). The RDG(P1) solution however is significantly improved compared with the P1P2 solution by judging the entropy production distribution on the surface of the airfoil. Note that the entropy production corresponds directly to the error of the numerical methods, as it should be zero everywhere for subsonic flows. The P1P2(RC) solution provides a further improvement over the P1P2(GG) solution, although the difference is very small. This numerical experiment demonstrates that the DG methods, unlike some of its finite volume counterparts, have the ability to accurately solve the compressible Euler equation on an anisotropic grid designed for solving the Navier-Stokes equations.

![Figure 11. Comparison of the computed pressure coefficient (left) and entropy production (right) distributions for subsonic flow past a NACA0012 airfoil at M∞ = 0.63, α=2°.](image)

5. CONCLUSIONS

A class of reconstructed discontinuous Galerkin methods has been described for the solution of the compressible Euler equations on arbitrary grids. Both Green-Gauss and least-squares reconstruction methods and a least-squares recovery method have been developed to obtain a quadratic polynomial representation from a underlying linear discontinuous Galerkin solution. The developed RDG methods are compact and combine efficiency of the reconstruction methods and accuracy and robustness of the DG methods. A number of test cases have been conducted to demonstrate the superior performance of the reconstructed discontinuous Galerkin methods over the underlying DG method. The numerical experiments indicate that all three reconstruction methods can deliver the desired third order of accuracy and significantly improve the accuracy of the underlying second-order DG method. The future development will be focused on the extension of the reconstructed DG methods for 3D problems.
ACKNOWLEDGMENTS

This manuscript has been authored by Battelle Energy Alliance, LLC under contract No. DE-AC07-05ID14517 (INL/CON-09-16255) with the U.S. Department of Energy. The United States Government retains and the published, by accepting the article for publication, acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The authors would like to acknowledge the partial support for this work provided by DOE under Nuclear Engineering University Program.

REFERENCES


