Abstract

LUO, LUQING. A Reconstructed Discontinuous Galerkin Method for the Compressible Euler Equations on Arbitrary Grids. (Under the direction of Dr. Hong Luo.)

A reconstruction-based discontinuous Galerkin (RDG) method is presented for the solution of the compressible Euler equations on arbitrary grids. By taking advantage of handily available and yet invaluable information, namely the derivatives, in the context of the discontinuous Galerkin methods, a polynomial solution of one degree higher is reconstructed using a least-squares method. The stencils used in the reconstruction involve only the von Neumann neighborhood (face-neighboring cells) and are compact and consistent with the underlying DG method. The resulting RDG method can be regarded as an improvement of a recovery-based DG method, in the sense that it shares the same nice features, such as high accuracy and efficiency, and yet overcomes some of its shortcomings such as a lack of flexibility, compactness, and robustness. The developed RDG method is used to compute a variety of flow problems on arbitrary meshes to demonstrate its accuracy, efficiency, robustness, and versatility. The numerical results indicate that this RDG method is third-order accurate at a cost slightly higher than its underlying second-order DG method, at the same time providing a better performance than the third order DG method, in terms of both computing costs and storage requirements.

© Copyright 2010 by Luqing Luo

All Rights Reserved

A Reconstructed Discontinuous Galerkin Method for the Compressible
Euler Equations on Arbitrary Grids

by
Luqing Luo

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Aerospace Engineering

Raleigh, North Carolina

2010

APPROVED BY:

Dr. Hassan A. Hassan

Dr. Jack R. Edwards

Dr. Zhilin Li

Dr. Hong Luo
Chair of Advisory Committee

Biography

Luqing Luo was born in Jiangxi Province, China on June 21, 1986. After graduating from Nanfeng No. 1 Middle School in 2003, Luqing earned a Bachelor of Science in Electrical Engineering from Civil Aviation University of China, Tianjin, China, in 2007. She continued to be enrolled in department of electrical engineering for her graduate study during 2008. From there she went to North Carolina State University, where she is going to complete her Master of Science degree in Mechanical and Aerospace Engineering.

Acknowledgements

I would like to thank my advisor, Dr. Hong Luo, for his guidance and support throughout my masters, and his considerable patience and assistance in the completion of the current thesis work. Through him I was able to gain a great deal of knowledge concerning CFD and advice for the future. The comments, suggestions, and corrections from my committee members, Dr. Hassan, Dr. Edwards and Dr. Li are greatly appreciated. I would also like to thank them for serving on my committee. I also want to thank my parents and my family, who have been there for me every step of the way through my growth, especially for the most difficult time since I started my study in US.

Table of Contents

List of Tables	v
List of Figures	vi
Chapter 1 Background Introduction	1
Chapter 2 Governing Equations	6
Chapter 3 Reconstructed Discontinuous Galerkin Method	7
Chapter 4 Numerical Examples	15
4.1 Sod shock tube problem	15
4.2 Subsonic Flows past a Circular Cylinder	16
4.3 Low Mach Number Flow past a Circular Cylinder	17
4.4 Subsonic Flow past a NACA0012 airfoil	17
4.5 Subsonic Flow past a three-element airfoil	18
Chapter 5 Future Work	19
References	20
Appendix	22
Appendix A Figures and Tables	23

List of Tables

Table A.1	Cost Analysis for different numerical methods in 2D	38
Table A.2	Subsonic circular cylinder test case: FV(P1) is order of $O(h^2)$	39
Table A.3	Subsonic circular cylinder test case: DG(P1) is order of $O(h^2)$	40

List of Figures

Figure A.1	Representation of polynomial solutions using finite element shape functions.	23
Figure A.2	Representation of polynomial solutions using a Taylor series expansion for a cell-centered scheme (left) and vertex-centered scheme (right).	23
Figure A.3	Comparison of computed density profile for Sod shock tube problem obtained by unlimited DG(P0), DG(P1), DG(P2), and RDG(P1) solutions with the analytical solution.	25
Figure A.4	Sequences of four successively globally refined meshes 16x5, 32x9, 64x17, 128x33 for computing subsonic flow past a circular cylinder.	27
Figure A.5	Computed Mach number contours in the flow field obtained by the FV(P1) method on 128x33 mesh (a), DG(P1) method on 64x17 mesh (b), DG(P2) method on 32x9 mesh (c), and RDG(P1) on 32x9 mesh (d) for subsonic flow past a circular cylinder at $M_\infty=0.38$.	29
Figure A.6	Accuracy summary for subsonic flow past a circular cylinder for FV(P1), DG(P1), DG(P2), and RDG(P1) methods.	30
Figure A.7	L_2 -error of numerical solutions against the number of degrees of freedom for subsonic flow past a circular cylinder by FV(P1), DG(P1), DG(P2), and RDG(P1) methods.	30
Figure A.8	(a): Unstructured hybrid grid (ntria=1,024, nquad=512, npoin=1,536, nbfac=128) used for computing low Mach number flow past a circular cylinder. (b): Comparison of computed velocity distributions on the surface of the circular cylinder obtained by the RDG(P1) and DG(P2) methods with the incompressible solution for low Mach number flow past a circular cylinder at $M_\infty=0.01$.	31
Figure A.9	Computed Mach number contours obtained by RDG(P1) (a) and DG(P2) (b) methods for low Mach number flow past a circular cylinder at $M_\infty=0.01$.	32
Figure A.10	Unstructured hybrid mesh (a) (nquad=1,533, ntria=3,469, npoin=3,346, nbfac=157) and computed Mach number contours by the DG(P1) (b), the RDG(P1) (c), and the DG(P2)(d) methods, respectively for subsonic flow past a NACA0012 airfoil at $M_\infty = 0.63, \alpha = 2^\circ$.	34
Figure A.11	Comparison of the computed pressure coefficient (a) and entropy production (b) distributions for subsonic flow past a NACA0012 airfoil at $M_\infty = 0.63, \alpha = 2^\circ$.	35
Figure A.12	Unstructured hybrid unstructured Cartesian and triangle mesh (ncart=13,892, and ntria=3,981) used for computing subsonic flow past a three-element airfoil at $M_\infty = 0.2, \alpha = 10^\circ$.	36
Figure A.13	The computed Mach number (a) and pressure (b) contours obtained by the RDG(P1) method for subsonic flow past a three-element airfoil at $M_\infty = 0.2, \alpha = 10^\circ$.	37

Chapter 1

Background Introduction

Historically, fluid dynamics in engineering applications has been modeled mostly within the finite volume (FV) approach. FV methods are well-developed and robust, and provide a backbone for engineering analyses in nuclear, chemical, aerospace, and automotive industries. On the other hand, high-order (>2 nd) FV discretizations in multi-dimensions on unstructured grids are difficult. In this respect, finite element (FE) discretizations have certainly an edge, providing not only superior spatial accuracy, but also potential for well-established mathematical background and error analyses.

FE methods are particularly well-developed for elliptic/parabolic problems (structural mechanics, heat conduction). Their applications to hyperbolic problems were not as wide-spread [1] [2]. Recent development of the Discontinuous Galerkin (DG) family [3] [4] [5] [6] [7], which utilizes piecewise-discontinuous basis functions and encapsulates FV as its lowest-order subclass, offered a promising direction. DG enables the carry over of all advances made in the past 20 years for FV Godunov-based methods [8] [9] directly into the FE framework.

Nowadays, the Discontinuous Galerkin methods (DGM) have recently become popular for the solution of systems of conservation laws due to combining two advantageous features commonly associated to finite element and 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 methods can be easily extended to higher-order (>2 nd) 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 FE methods and FV 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.

DGM are indeed a natural choice for the solution of hyperbolic problems, such as the compressible Euler equations. However, the DG formulation is far less certain and advantageous for the compressible Navier-Stokes equations, where both convective and dissipative fluxes exist. A severe difficulty raised by the application of the DG methods to the Navier-Stokes equations is the approximation of the numerical fluxes for diffusion terms, which has to properly resolve the discontinuities at the interfaces.

Taking a simple arithmetic mean of the solution derivatives from the left and right is inconsistent, because the arithmetic mean of the solution derivatives does not take into account a possible jump of the solutions. A number of numerical methods have been proposed in the literature, such as those by Bassi and Rebay [10], Cockburn and Shu [11], Baumann and Oden [12] and many others. More recently, Van Leer et al. [13] [14] [15] pioneered a recovery-based DG method, called inter-cell recovery, where diffusive fluxes are computed from a smooth, locally recovered solution at the cell interfaces that in the weak sense is indistinguishable from the discontinuous Galerkin solution. Motivated by this recovery principal, Nourgaliev et al [16] [17] introduced the in-cell recovery concept aimed to enhance the accuracy of the DG methods. Rather than removing an interface discontinuity, the goal of the cell-recovery DG, referred to as rDG, is to increase the order of the polynomial solution using the recovery principal by weak interpolation.

In the inter-cell recovery approach the diffusive fluxes are computed from a smooth, locally recovered solution that in the weak sense is indistinguishable from the discontinuous discrete

solution [13] [19]. This eliminates the introduction of ad hoc penalty or coupling terms found in traditional DG methods; in the simple case of 1-D diffusion the recovery principle can be shown to generate a string of penalty-like terms not encountered in standard DG methods. The most attractive feature of inter-cell recovery approach is the simplicity and generality of the recovery principle, which immediately allows one to create and further develop DG diffusion schemes. In addition, these schemes are of higher accuracy, while boasting the smallest eigenvalues, hence the largest stability range for explicit time-marching [20]. However, the promising results are limited to recover a solution on the union of two neighboring, interface-sharing cells, which can be referred as a binary recovery. Such a recovery is anisotropic in its accuracy [19]: its representation of the solution in the direction normal to the interface is accurate to the order $2p+1$, while the representation in any direction parallel to the interface is only of the order p . Furthermore, the good approximations are restricted to Cartesian grids. In particular, if the centroids of three triangles line up, there is insufficient coverage of 2-D space when the approach is applied on unstructured grids. Similarly, if only the solution-average are matched in the neighboring cells (six data in total), the same degenerate geometry even prohibits going beyond a $p=1$ basis.

A traditional approach for high-order-accuracy extensions of FV method involves local in-cell recoveries. Since FV methods solve for cell-average quantities only (one degree of freedom per variable, per cell), to recover higher-order representation of the solution inside a cell, one needs to involve neighbors. While this is manageable in second-order, it becomes impractical when the third- and higher-order 3D discretizations are attempted on unstructured grids (through the ENO/WENO family [21]), due to the rapid increase of stencil and numerous options/configurations available/required. Moreover, cross-derivatives are typically ignored, which might lead to an unacceptable loss of accuracy in the directions not aligned with mesh. On the other hand, the DG method pursues another venue to achieve high-order accuracy, by solving for additional degrees of freedom per cell, rendering a compact stencil, but larger vectors of unknowns on the same grid. For example, quadratic DG (3rd-order) in 3D would involve 10 times larger solution vector (10 degrees of freedom per variable, per cell) than its FV-PPM counterpart (formally, also 3rd-order-accurate in each grid direction). In this situation, in-cell recovery was introduced and applied to DG in order to increase the accuracy of discretization, provide some sort of a balance/compromise between accuracy, stencil (compactness) and the size of the solution vector.

It can be shown that in 1D, the resulting recovery-based DG method using piecewise constant approximation rDG(P0) is nothing but FV-PPM method [22], linear rDG(P1) is 6th order accurate, quadratic rDG(P2) is 9th-order accurate, and cubic rDG(P3) 12th-order accurate, versus the 2nd, 3rd, and 4th-order accuracy of the underlying DG method, while keeping the same size of the degrees of freedom and being compact. This recovery-based DG method

has been successfully extended to 2D problems on quadrilateral grids. However, the resulting rDG methods are not completely satisfactory, since the stencils in the recovery have to involve the vertex-neighboring cells and thus destroy the compactness of the underlying DG method. For instance, in the case of rDG(P0) recovery, a quadratic polynomial solution for any cell can be fully recovered using piecewise constant solutions at that cell and its two neighbors in 1D. However, a fully quadratic polynomial has six degrees of freedom, and thus requires six cells in order to recover a quadratic solution in 2D. Unfortunately, there are only five cells available on quadrilateral grids and four cells on triangular grids, when only face-neighboring cells are used in the recovery. Clearly, most of appealing features possessed by the rDG method are lost for the multidimensional problems, and especially on unstructured arbitrary grids. The fundamental issue is that it is extremely difficult, if not impossible, to judiciously choose a proper form of a recovered polynomial and a set of cells in such a way that the resulting recovered linear system is well conditioned, and thus can be inverted. Indeed, recovery is probably not the best way to obtain a higher-order polynomial representation of an underlying DG solution. Rather, reconstruction widely used in the finite volume methods is a better choice to obtain a higher-order polynomial representation.

Part of the present effort is concerned with developing a reconstructed DG method (RDG) for the solution of the compressible Euler equations on arbitrary grids. The idea behind this RDG method is to combine the efficiency of the reconstruction methods widely used to increase the accuracy of the finite volume (FV) methods and the accuracy and robustness of the DG methods viewed as a generalization of the FV method to higher orders in an attempt to develop an accurate, efficient, and robust method for solving compressible flow problems. Starting from the underlying linear polynomial DG solution, a quadratic polynomial solution is reconstructed using a least-squares method. The reconstruction only involves the Von Neumann neighborhood in order to maintain the compactness of the underlying DG method, which is made possible by taking advantage of handily available and yet invaluable information, namely the derivative in the underlying DG method. Similar to the recovery-based DG methods (rDG), higher order accuracy can be expected due to a higher order representation of the reconstructed polynomial solution. Unlike the rDG methods which use both face-neighbors and vertex-neighbors in 2D to recover a higher order polynomial, this RDG method involves only the face-neighboring cells, thus is compact. Since the least-squares method is used to reconstruct the polynomial solutions, the reconstructed DG method is robust and flexible. As the underlying DG method is second-order, and the basis functions are at most linear functions, fewer quadrature points are then required for both domain and face integrals, and the number of unknowns (the number of degrees of freedom) remains the same as the DG(P1). Consequently, this reconstructed DG method is expected to be more efficient than the third order DG method. The developed RDG method is used to compute a variety of flow problems on arbitrary meshes to demonstrate

its accuracy, efficiency, and robustness. The numerical results indicate that this reconstructed DG method is able to obtain a third-order accurate solution at a slightly higher cost than its second-order DG method and provide a significant increase in performance over the third order DG method in terms of computing costs and storage requirements.

The remainder of this paper is outlined as follows: The governing equations are listed in Section 2. The underlying reconstructed discontinuous Galerkin method is presented in Section 3. Extensive numerical experiments are reported in Section 4, while Section 5 and 6 will present conclusions and future work.

Chapter 2

Governing Equations

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

$$\frac{\partial \mathbf{U}(x, t)}{\partial t} + \frac{\partial \{\mathbf{F}_k(\mathbf{U}(x, t))\}}{\partial x_k} = 0 \quad (2.1)$$

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

$$\mathbf{U}^T = (\rho, \rho u_i, \rho e), \mathbf{F}_j^T = (\rho u_j, \rho u_i u_j + p \delta_{ij}, u_j(\rho e + p)) \quad (2.2)$$

Here ρ , 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 i . The pressure can be computed from the equation of state

$$p = (\gamma - 1)\rho(e - \frac{1}{2}u_j u_j) \quad (2.3)$$

Chapter 3

Reconstructed Discontinuous Galerkin Method

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 $\mathbf{W}$, integrating over the domain Ω , and then performing an integration by parts,

$$\int_{\Omega} \frac{\partial \mathbf{U}}{\partial t} \mathbf{W} d\Omega + \int_{\Gamma} \mathbf{F}_k \mathbf{n}_k d\Gamma - \int_{\Omega} \mathbf{F}_k \frac{\partial \mathbf{W}}{\partial x_k} d\Omega = 0, \forall \mathbf{W} \in V \quad (3.1)$$

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

$$V_h^p = \{v_h \in [L_2(\Omega)]^m : v_h|_{\Omega_e} \in [V_p^m], \forall \Omega_e \in \Omega\} \quad (3.2)$$

which consists of discontinuous vector-valued polynomial functions of degree p , and where m is the dimension of the unknown vector and

$$V_p^m = \text{span}\{\prod x_i^{\alpha_i} : 0 < \alpha_i < p, 0 < i < d\} \quad (3.3)$$

where α 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 Ω_e .

Find $\mathbf{U} \in V_h^p$ such as

$$\frac{d}{dt} \int_{\Omega_e} \mathbf{U}_h \mathbf{W}_h d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h) \mathbf{n}_k \mathbf{W}_h d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h) \frac{\partial \mathbf{W}_h}{\partial x_k} d\Omega = 0, \forall \mathbf{W}_h \in V_h^p \quad (3.4)$$

where $\mathbf{U}_h$ and $\mathbf{W}_h$ represent the finite element approximations to the analytical solution $\mathbf{U}$ and the test function $\mathbf{W}$ 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,

$$\frac{d}{dt} \int_{\Omega_e} \mathbf{U}_h B_i d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h) \mathbf{n}_k B_i d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h) \frac{\partial B_i}{\partial x_k} d\Omega = 0, 1 < i < N \quad (3.5)$$

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.

The flux function $\mathbf{F}_k(\mathbf{U}_h) \mathbf{n}_k$ appearing in the second terms of Eq. (3.5) is replaced by a numerical Riemann flux function $\mathbf{H}_k(\mathbf{U}_h^L, \mathbf{U}_h^R, \mathbf{n}_k)$ 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. 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 $DG(P_0)$ method, i.e., to the discontinuous Galerkin method using a piecewise constant polynomial. Consequently, the $DG(P_k)$ 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.

The domain and boundary integrals in Eq. (3.5) are calculated using Gauss quadrature formulas. The number of quadrature points used is chosen to integrate exactly polynomials of order of $2p$ on the reference element. In 2D, two, three, and four points are used for linear, quadratic, and cubic basis function in the boundary integrals. The domain integrals are evaluated using three, six, and thirteen points for triangular elements and four, nine, and sixteen points for quadrilateral elements, respectively.

In the traditional DGM, numerical polynomial solutions $\mathbf{U}_h$ in each element are expressed using either standard Lagrange finite element or hierarchical node-based basis as following

$$\mathbf{U}_h = \sum_{i=1}^N \mathbf{U}_i(t) B_i(x) \quad (3.6)$$

where B_i are the finite element basis functions. As a result, the unknowns to be solved are the variables at the nodes $\mathbf{U}_i$, as illustrated in Figure B.1 for linear and quadratic polynomial

approximations.

On each cell, a system of NxN 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 $\mathbf{U}$ 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

$$\begin{aligned} \mathbf{U}_h = \mathbf{U}_c &+ \frac{\partial \mathbf{U}}{\partial x}|_c(x - x_c) + \frac{\partial \mathbf{U}}{\partial y}|_c(y - y_c) + \frac{\partial^2 \mathbf{U}}{\partial x^2}|_c \frac{(x - x_c)^2}{2} \\ &+ \frac{\partial^2 \mathbf{U}}{\partial y^2}|_c \frac{(y - y_c)^2}{2} + \frac{\partial^2 \mathbf{U}}{\partial x \partial y}|_c (x - x_c)(y - y_c) \end{aligned} \quad (3.7)$$

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

$$\begin{aligned} \mathbf{U}_h = \tilde{U} &+ \frac{\partial \mathbf{U}}{\partial x}|_c(x - x_c) + \frac{\partial \mathbf{U}}{\partial y}|_c(y - y_c) + \frac{\partial^2 \mathbf{U}}{\partial x^2}|_c \left(\frac{(x - x_c)^2}{2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(x - x_c)^2}{2} d\Omega \right) \\ &+ \frac{\partial^2 \mathbf{U}}{\partial y^2}|_c \left(\frac{(y - y_c)^2}{2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(y - y_c)^2}{2} d\Omega \right) \\ &+ \frac{\partial^2 \mathbf{U}}{\partial x \partial y}|_c \left((x - x_c)(y - y_c) - \frac{1}{\Omega_e} \int_{\Omega_e} (x - x_c)(y - y_c) d\Omega \right) \end{aligned} \quad (3.8)$$

where $\tilde{U}$ is the mean value of $\mathbf{U}$ 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. In this case, the dimension of the polynomial space is six and the six basis functions are

$$\begin{aligned} B_1 &= 1, B_2 = x - x_c, B_3 = y - y_c, \\ B_4 &= \frac{(x - x_c)^2}{2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(x - x_c)^2}{2} d\Omega, \\ B_5 &= \frac{(y - y_c)^2}{2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(y - y_c)^2}{2} d\Omega, \\ B_6 &= (x - x_c)(y - y_c) - \frac{1}{\Omega_e} \int_{\Omega_e} (x - x_c)(y - y_c) d\Omega \end{aligned} \quad (3.9)$$

The discontinuous Galerkin formulation then leads to the following six equations

$$\begin{aligned} \frac{d}{dt} \int_{\Omega_e} \tilde{U} d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h) \mathbf{n}_k d\Gamma &= 0, i = 1 \\ M_{5X5} \frac{d}{dt} \left(\frac{\partial \mathbf{U}}{\partial x} |_c \frac{\partial \mathbf{U}}{\partial y} |_c \frac{\partial^2 \mathbf{U}}{\partial x^2} |_c \frac{\partial^2 \mathbf{U}}{\partial y^2} |_c \frac{\partial^2 \mathbf{U}}{\partial x \partial y} |_c \right)^T + R_{5X1} &= 0 \end{aligned} \quad (3.10)$$

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

$$\int_{\Omega_e} B_1 B_i d\Omega = 0, 2 < i < 6 \quad (3.11)$$

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:

$$\begin{aligned} B_1 &= 1, B_2 = \frac{x - x_c}{\Delta x}, B_3 = \frac{y - y_c}{\Delta y}, \\ B_4 &= \frac{(x - x_c)^2}{2\Delta x^2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(x - x_c)^2}{2\Delta x^2} d\Omega, \\ B_5 &= \frac{(y - y_c)^2}{2\Delta y^2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(y - y_c)^2}{2\Delta y^2} d\Omega, \\ B_6 &= \frac{(x - x_c)(y - y_c)}{\Delta x \Delta y} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(x - x_c)(y - y_c)}{\Delta x \Delta y} d\Omega \end{aligned} \quad (3.12)$$

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

$$\mathbf{U}_h = \tilde{U} B_1 + \mathbf{U}_x B_2 + \mathbf{U}_y B_3 + \mathbf{U}_{xx} B_4 + \mathbf{U}_{yy} B_5 + \mathbf{U}_{xy} B_6 \quad (3.13)$$

where

$$\begin{aligned} \mathbf{U}_x &= \frac{\partial \mathbf{U}}{\partial x} |_c \Delta x, \mathbf{U}_y = \frac{\partial \mathbf{U}}{\partial y} |_c \Delta y, \mathbf{U}_{xx} = \frac{\partial^2 \mathbf{U}}{\partial x^2} |_c \Delta x^2, \\ \mathbf{U}_{yy} &= \frac{\partial^2 \mathbf{U}}{\partial y^2} |_c \Delta y^2, \mathbf{U}_{xy} = \frac{\partial^2 \mathbf{U}}{\partial x \partial y} |_c \Delta x \Delta y. \end{aligned} \quad (3.14)$$

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 [23] [24] [25] 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 ($> 1^{st}$) 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 [26]. 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. On 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. The reconstructed DG method to be described below, which can be regarded as a cross between DG and reconstruction finite volume methods, is our attempt to combine the efficiency of the reconstruction methods and the accuracy of the DG methods. 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

$$\mathbf{U}_i = \tilde{U}_i + \mathbf{U}_{xi}B_2 + \mathbf{U}_{yi}B_3 \quad (3.15)$$

Using this underlying linear polynomial DG solution in the neighboring cells, one can reconstruct a quadratic polynomial solution $\mathbf{U}_i^R$ as follows:

$$\mathbf{U}_i^R = \tilde{U}_i^R + \mathbf{U}_{xi}^R B_2 + \mathbf{U}_{yi}^R B_3 + \mathbf{U}_{xxi}^R B_4 + \mathbf{U}_{yyi}^R B_5 + \mathbf{U}_{xyi}^R B_6 \quad (3.16)$$

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 simply coincide with the ones from the underlying DG solution, i.e.,

$$\tilde{U}_i^R = \tilde{U}_i, \mathbf{U}_{xi}^R = \mathbf{U}_{xi}, \mathbf{U}_{yi}^R = \mathbf{U}_{yi} \quad (3.17)$$

As a result, only three second derivatives need to be determined, which can be accomplished 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

$$\begin{aligned} \mathbf{U}_j &= \tilde{U}_i + \mathbf{U}_{xi}B_2 + \mathbf{U}_{yi}B_3 + \mathbf{U}_{xxi}^R B_4 + \mathbf{U}_{yyi}^R B_5 + \mathbf{U}_{xyi}^R B_6 \\ \frac{\partial \mathbf{U}}{\partial x}|_j &= \mathbf{U}_{xi} \frac{1}{\Delta x_i} + \mathbf{U}_{xxi}^R \frac{B_2}{\Delta x_i} + \mathbf{U}_{xyi}^R \frac{B_3}{\Delta x_i}, \\ \frac{\partial \mathbf{U}}{\partial y}|_j &= \mathbf{U}_{yi} \frac{1}{\Delta y_i} + \mathbf{U}_{yyi}^R \frac{B_3}{\Delta y_i} + \mathbf{U}_{xyi}^R \frac{B_2}{\Delta y_i} \end{aligned} \quad (3.18)$$

where the basis function B are evaluated at the center of cell j , i.e., $B = \mathbf{B}(x_j, y_j)$. This can be written in a matrix form as follows:

$$\begin{pmatrix} B_4^j & B_5^j & B_6^j \\ B_2^j & 0 & B_3^j \\ 0 & B_3^j & B_2^j \end{pmatrix} \begin{pmatrix} \mathbf{U}_{xxi}^R \\ \mathbf{U}_{yyi}^R \\ \mathbf{U}_{xyi}^R \end{pmatrix} = \begin{pmatrix} \mathbf{U}_j - (\mathbf{U}_i B_1^j + \mathbf{U}_{xi} B_2^j + \mathbf{U}_{yi} B_3^j) \\ \frac{\Delta x_i}{\Delta x_j} \mathbf{U}_{xj} - \mathbf{U}_{xi} \\ \frac{\Delta y_i}{\Delta y_j} \mathbf{U}_{yj} - \mathbf{U}_{yi} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1^j \\ \mathbf{R}_2^j \\ \mathbf{R}_3^j \end{pmatrix} \quad (3.19)$$

where $\mathbf{R}$ is used to represent the right-hand-side for simplicity. Similar equations 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 9×3 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

$$\begin{pmatrix} \sum_j (B_4^j B_4^j + B_4^j B_4^j) & \sum_j B_4^j B_5^j & \sum_j (B_4^j B_6^j + B_2^j B_3^j) \\ \sum_j B_4^j B_5^j & \sum_j (B_5^j B_5^j + B_3^j B_3^j) & \sum_j (B_5^j B_6^j + B_2^j B_3^j) \\ \sum_j (B_4^j B_6^j + B_2^j B_3^j) & \sum_j (B_5^j B_6^j + B_2^j B_3^j) & \sum_j (B_6^j B_6^j + B_2^j B_2^j + B_3^j B_3^j) \end{pmatrix} \begin{pmatrix} \mathbf{U}_{xxi}^R \\ \mathbf{U}_{yyi}^R \\ \mathbf{U}_{xyi}^R \end{pmatrix} = \begin{pmatrix} \sum_j (B_4^j R_1^j + B_2^j R_2^j) \\ \sum_j (B_5^j R_1^j + B_3^j R_3^j) \\ \sum_j (B_6^j R_1^j + B_3^j R_2^j + B_2^j R_3^j) \end{pmatrix} \quad (3.20)$$

This linear system of 3×3 can be then trivially solved to obtain the second derivatives of the reconstructed quadratic polynomial solution.

This 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(P1) 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 least-squares reconstruction, 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. The cost analysis for the FV(P1), DG(P1), RDG(P1) and DG(P2) is summarized in Table 1, where the memory requirement for storing only the implicit diagonal matrix is given as well, which grows quadratically with the order of the DG methods. We would like to emphasize that the storage requirements for the implicit DG methods are extremely demanding, especially for higher-order DG methods.

Chapter 4

Numerical Examples

All of the computations are performed on a Dell XPS M1210 laptop computer (2.33 GHz Intel(R) Core(TM) 2 CPU T7600 with 4GBytes memory) using a Suse 11.0 Linux operating system. A well-tested finite volume code [25] [27], is used as a reference to compare the accuracy and performance of the DG method for some test cases.

4.1 Sod shock tube problem

The shock tube problem constitutes a particularly interesting and difficult test case, since it represents an exact solution to the full system of one-dimensional Euler equations containing simultaneously a shock wave, a contact discontinuity, and an expansion fan. This test case is chosen to demonstrate that the reconstructed DG method is able to maintain the robustness of the underlying DG methods, as the reconstruction scheme used in the finite volume methods will unavoidably generate negative values for density and energy in this case, leading to a breakdown of the finite volume solution process. The initial conditions in the present computation are the following: $\rho = 1, u = 0, p = 1$ for $0 < x < 0.5$, and $\rho = 0.125, u = 0, p = 0.1$ for $0.5 < x < 1$. Figure B.3 shows a comparison of the density profiles obtained by the unlimited DG(P0), DG(P1), DG(P2), and RDG(P1) and the exact solutions, respectively. This is a 2D simulation of a 1D problem. The mesh consists of 50 cells in the X-direction and 1 cell in the Y-direction. As expected, unlimited higher order DG(P1), DG(P2), and RDG(P1) solutions exhibit small oscillations in the vicinity of discontinuities and higher order DG solution yields a sharper resolution for both contact discontinuity and shock wave than a lower order DG solution. The RDG(P1) solution is more accurate than the DG(P1) solution, although not as accurate as DG(P2) solution. The reconstruction scheme used in the DG method does not create negative density, and thus does not lead to the breakdown of the solution process. This can be attributed to the robustness of the underlying DG(P1) method upon which a reconstruction is based.

4.2 Subsonic Flows past a Circular Cylinder

This is a well-known test case: subsonic flow past a circular cylinder at a Mach number of $M_\infty = 0.38$. This test case is chosen to verify if a formal order of convergence rate of the RDG method can be achieved for the compressible Euler equations on unstructured grids. Figure B.4 shows four successively refined o-type grids having 16x5, 32x9, 64x17, and 128x33 points, respectively. The first number refers to the number of points in the circular direction, and the second designates the number of concentric circles in the mesh. 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

$$r_i = r_1(1 + \frac{2\pi}{128} \sum_{j=0}^{i-1} \alpha^j), i = 2, \dots, 33 \quad (4.1)$$

where $\alpha = 1.1580372$. The coarser grids are generated by successively coarsening the finest mesh. Numerical solutions to this problem are computed using FV(P1), DG(P1), DG(P2), and RDG(P1) methods on these four grids to obtain quantitative measurement of the order of accuracy and discretization errors. The detailed results for this test case are presented in Tables 2-5. They show the mesh size, the number of degrees of freedom, the $L_2 - error$ of the solutions, and the order of convergence. In this case, the following entropy production defined as

$$\varepsilon = \frac{S - S_\infty}{S_\infty} = \frac{p}{p_\infty} \left(\frac{\rho_\infty}{\rho} \right)^\gamma - 1 \quad (4.2)$$

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. Figure B.5 shows the computed Mach number contours in the flow field obtained by FV(P1) on the 128x33 mesh, DG(P1) on the 64x17 mesh, and DG(P2) and RDG(P1) on the 32x8 mesh, respectively. One can see that the results obtained by DG(P2) on the 32x8 mesh are more accurate than the ones obtained by DG(P1) on the 64x17 mesh, which in turn are more accurate than the ones obtained by FV(P1) on the 128x33 mesh. Both RDG(P1) and DG(P2) solutions are virtually identical for this case. However, the DG(P2) does yield a slightly more accurate solution than the RDG(P1) as one can see in Figure B.6, which provides the details of the spatial accuracy of each method for this numerical experiment. As expected, the DG method exhibits a full $O(h^{p+1})$ order of convergence. The reconstructed DG(P1) method does offer a full $O(h^{p+2})$ order of convergence, and the reconstruction scheme does add one additional order of accuracy to the underlying DG(P1) method. Figure B.7 illustrates that a higher order DG method requires significantly less degrees of freedom than a lower order one for the same accuracy. However, the RDG(P1) outperforms DG(P2), by measuring the number of the degrees of freedom required to achieve the same accuracy.

4.3 Low Mach Number Flow past a Circular Cylinder

This test case is chosen to demonstrate that the reconstructed DG method is able to maintain the accuracy of the underlying DG method for solving low Mach number flow problems. The computation performed for low speed flows past a circular cylinder at a Mach number of $M_\infty = 0.01$ using both DG(P2) and RDG(P1) methods on a hybrid mesh is shown in Fig. 8, where the comparison of the velocity distributions on the surface of the circular cylinder obtained by these two methods, and those of the FV(P1) solution and the analytic solution for incompressible flows are presented as well. One can see clearly dramatic deterioration of the FV(P1) solution. The mesh consists of 1,024 triangular elements, 512 quadrilateral elements, 1,536 grid points, and 128 boundary faces. The computed Mach number contours in the flow field obtained by these two methods are shown in Fig. 9, respectively. The results illustrate that the RDG(P1) and the DG(P2) solutions are virtually identical, demonstrating that the RDG(P1) method is able to yield an accurate solution, and thus maintain the ability of the underlying DG method for the low Mach number flows.

4.4 Subsonic Flow past a NACA0012 airfoil

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° . 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 B.10 shows the mesh used in this numerical experiment and the computed Mach number contours in the flow field obtained by DG(P1), RDG(P1), and DG(P2) 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), RDG(P1), and DG(P2) methods are shown in Fig. 10, 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. 11. 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 DG(P1) 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 DG(P2) solution provides a further improvement over the RDG(P1) solution, although the difference is very small. This numerical experiment demonstrates that the RDG(P1) method, unlike some of its finite volume counterparts, has the ability to accurately solve the compressible Euler equation on an anisotropic grid designed for solving the Navier-Stokes equations.

4.5 Subsonic Flow past a three-element airfoil

Finally, an illustrative example is presented to demonstrate that the developed RDG method can be applied to problems of scientific and industrial interests. The computation is performed on a three-element airfoil at a free stream of Mach number of 0.2 and an angle of attack of 10° . The hybrid unstructured Cartesian and triangle grid is shown in Figure B.12 and consists of 13,892 Cartesian cells and 3,981 triangular elements. The computed Mach number and pressure contours in the flow field are shown in Figure B.13, demonstrating the accuracy and robustness of the developed RDG method for computing complicated flows of practical importance.

Chapter 5

Future Work

A reconstructed discontinuous Galerkin method using a Taylor basis has been introduced for the solution of the compressible Euler equations on arbitrary grids. The developed RDG method combines efficiency of the reconstruction methods and accuracy and robustness of the DG methods and is able to achieve one order higher accuracy than the underlying DG method. A number of numerical experiments have been conducted to demonstrate the superior performance of this reconstructed discontinuous Galerkin method over the underlying DG method. The developed RDG method has also successfully been extended to perform an inter-cell reconstruction for the discretization of the viscous fluxes in the Navier-Stokes equations. The current development is focused on the extension of this reconstructed DG method for 3D problems.

References

- [1] Zienkiewica, O.C., Taylor, R.L. and Nithiarasu, P. (2005) *The Finite Element for Fluid Dynamics*. Elsevier, Butterworth-Heinemann.
- [2] Gresho, P.M. R.L., S. (1998) *Incompressible Flow and the Finite Element Method, Volume 1, Advection-Diffusion and Isothermal Laminar Flow*. John Wiley & Sons Ltd.
- [3] Reed, W.H. and Hill, T.R. (1973) *Triangular Mesh Methods for the Neutron Transport Equation*. Los Alamos Scientific Laboratory Report.
- [4] Cockburn, B., Lin, S.-Y. and Shu, C.-W. (1989) *TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws III: One-Dimensional Systems*. Journal of Computational Physics 84, 90-113.
- [5] Cockburn, B. and Shu, C.-W. (1989) *TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws II: General Framework*. Mathematics of Computation 52, 411-435.
- [6] Cockburn, B. (1998) *Introduction to the Discontinuous Galerkin Method for Convection-Dominated Problems, Vol. 1697*. Springer, Berlin Heidelberg.
- [7] Cockburn, B., Hou, S. and Shu, C.W. (1990) *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case*. Mathematics of Computation 54, 545-581.
- [8] van Leer, B. (1977) *Towards the Ultimate Conservation Difference Scheme IV. A New Approach to Numerical Convection*. Journal of Computational Physics 23, 276-299.
- [9] Toro, E.F. (1999) *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer-Verlag, Berlin Heidelberg.
- [10] Bassi, F. and Rebay, S. (1997) *A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier-Stokes Equations*. Journal of Computational Physics, Vol. 131, pp. 267-179.
- [11] Cockburn, B. and Shu, C.W. (2001) *The Local Discontinuous Galerkin Method for Time-depend Convection-Diffusion System*. SIAM, Journal of Numerical Analysis, Vol 16.
- [12] Baumann, C. E. and Oden, J. T. (1999) *A discontinuous hp Finite element Method for the Euler and Navier-Stokes Equations*. International Journal for Numerical Methods in Fluids, Vol. 31.
- [13] van Leer, B. and Nomura, S. (2005) *Discontinuous Galerkin Method for Diffusion*. AIAA paper 2005-5108.
- [14] van Leer, B. and Lo, M. (2007) *A Discontinuous Galerkin Method for Diffusion Based on Recovery*. AIAA paper 2007-4083.

- [15] Raalte, M. and van Leer, B. (2009) *Bilinear Forms for the Recovery-Based Discontinuous Galerkin Method for Diffusion*. Communication of Computational Physics, Vol. 5, No. 2-4, pp. 683-693.
- [16] Nourgaliev, R., Theofanous, T. G., Park, H., Mousseau, V. and Knoll, D. (2008) *Direct Numerical Simulation of Interfacial Flows: Implicit Sharp-Interface Method*. AIAA paper 2008-1453.
- [17] Nourgaliev, R., Park, H. and Mousseau, V. (2009) *Recovery Discontinuous Galerkin Jacobian-Free Newton-Krylov Method for Multiphysics Problems*. Computational Fluid Dynamics Review 2009.
- [18] van Leer, B., Lo, M. and van Raalte, M. (2007) *A Discontinuous Galerkin method for diffusion based on recovery*. AIAA Paper 2007-4083.
- [19] Huynh, H. T. (2009) *A Reconstruction Approach to High-order Schemes including Discontinuous Galerkin for Diffusion*. AIAA paper 2009-0403.
- [20] Shu, C.-W. and Osher, S. (1989) *Efficient implementation of essentially non-oscillatory shock-capturing schemes*. Journal of Computational Physics 77, 439-471.
- [21] Colella and Woodward, P. R. (1984) *The Piecewise Parabolic Method (PPM) for Gas-Dynamical Simulations*. Journal of Computational Physics, Vol. 54, No. 1, pp. 115-173.
- [22] Aftosmis, M., Gaitonde, D. and Tavares, T. S. (1994) *On the Accuracy, Stability, and Monotonicity of Various Reconstruction Algorithms for Unstructured Meshes*. AIAA paper 94-0415.
- [23] Haselbacher, A., McGuirk, J. J. and Page, G. J. (1999) *Finite-Volume Discretization Aspects for Viscous Flows for Mixed Unstructured Meshes*. AIAA Journal, 37(2), pp. 177-184.
- [24] Luo, H., Baum, J. D. and Lohner, R. (1998) *A Fast, Matrix-free Implicit Method for Compressible Flows on Unstructured Grids*. Journal of Computational Physics, Vol. 146, No. 2, pp. 664-690.
- [25] Luo, H., Baum, J. D. and Lohner, R. (2008) *A Discontinuous Galerkin Method Using Taylor Basis for Compressible Flows on Arbitrary Grids*. Journal of Computational Physics, DOI: 210.1016/J.JCP.2008.06.035, Vol. 227, No. 20, pp. 8875-8893.
- [26] Luo, H., Sharov, D., Baum, J. D. and Lohner, R. (2001) *On the Computation of Compressible Turbulent Flows on Unstructured Grids*. International Journal of Computational Fluid Dynamics, Vo. 14, pp. 253-270.
- [27] Lo, M. and van Leer, B. (2009) *Analysis and Implementation of Recovery-Based Discontinuous Galerkin for Diffusion*. AIAA paper 2009-3786.

Appendix

Appendix A

Figures and Tables

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

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

Figure A.3: Comparison of computed density profile for Sod shock tube problem obtained by unlimited DG(P0), DG(P1), DG(P2), and RDG(P1) solutions with the analytical solution.

Figure A.4: Sequences of four successively globally refined meshes 16×5 , 32×9 , 64×17 , 128×33 for computing subsonic flow past a circular cylinder.

(a)

(b)

Figure A.5: Computed Mach number contours in the flow field obtained by the FV(P1) method on 128x33 mesh (a), DG(P1) method on 64x17 mesh (b), DG(P2) method on 32x9 mesh (c), and RDG(P1) on 32x9 mesh (d) for subsonic flow past a circular cylinder at $M_\infty=0.38$.

Figure A.6: Accuracy summary for subsonic flow past a circular cylinder for FV(P1), DG(P1), DG(P2), and RDG(P1) methods.

Figure A.7: L_2 -error of numerical solutions against the number of degrees of freedom for subsonic flow past a circular cylinder by FV(P1), DG(P1), DG(P2), and RDG(P1) methods.

Figure A.8: (a): Unstructured hybrid grid ($n_{\text{tria}}=1,024$, $n_{\text{quad}}=512$, $n_{\text{poin}}=1,536$, $n_{\text{bfac}}=128$) used for computing low Mach number flow past a circular cylinder. (b): Comparison of computed velocity distributions on the surface of the circular cylinder obtained by the RDG(P1) and DG(P2) methods with the incompressible solution for low Mach number flow past a circular cylinder at $M_{\infty}=0.01$.

(a)

(b)

Figure A.9: Computed Mach number contours obtained by RDG(P1) (a) and DG(P2) (b) methods for low Mach number flow past a circular cylinder at $M_\infty=0.01$.

(a)

(b)

(c)

(d)

Figure A.10: Unstructured hybrid mesh (a) (nquad=1,533, ntria=3,469, npoin=3,346, nbfac=157) and computed Mach number contours by the DG(P1) (b), the RDG(P1) (c), and the DG(P2)(d) methods, respectively for subsonic flow past a NACA0012 airfoil at $M_\infty = 0.63, \alpha = 2^\circ$.

Figure A.11: Comparison of the computed pressure coefficient (a) and entropy production (b) distributions for subsonic flow past a NACA0012 airfoil at $M_\infty = 0.63, \alpha = 2^\circ$.

Figure A.12: Unstructured hybrid unstructured Cartesian and triangle mesh ($n_{\text{cart}}=13,892$, and $n_{\text{tria}}=3,981$) used for computing subsonic flow past a three-element airfoil at $M_{\infty} = 0.2$, $\alpha = 10^{\circ}$.

Figure A.13: The computed Mach number (a) and pressure (b) contours obtained by the RDG(P1) method for subsonic flow past a three-element airfoil at $M_\infty = 0.2$, $\alpha = 10^\circ$.

Table A.1: Cost Analysis for different numerical methods in 2D

	FV(P1)	DG(P1)	RDG(P1)	DG(P2)
Quadrature Points for Boundary Integrals	1	2	2	3
Quadrature Points for Domain Integrals	0	3 (triangle) 4 (quadrilateral)	4 4	6 9
Order of Accuracy	$O(h^2)$	$O(h^2)$	$O(h^3)$	$O(h^3)$
Reconstruction	Yes	No	Yes	No
Storage for Implicit Diagonal Matrix	25 words per element	225	225	900

Table A.2: Subsonic circular cylinder test case: FV(P1) is order of $O(h^2)$

Mesh	No. DOFs	L_2 -error	Order
16X5	80	2.37148E-01	-
32X9	288	7.76551E-01	1.595
64X17	1,088	1.36962E-02	2.551
128X33	4,224	3.54568E-03	1.951

Table A.3: Subsonic circular cylinder test case: DG(P1) is order of $O(h^2)$

Mesh	No. DOFs	L_2 -error	Order
16X5	360	5.68722E-02	-
32X9	1,536	1.07103E-02	2.443
64X17	6,144	1.67302E-03	2.688
128X33	24,576	2.34369E-04	2.838