AIAA 2011-3406

20th AIAA Computational Fluid Dynamics Conference, June 27 – 30, 2011, Honolulu, Hawaii

# Recovery-Based Discontinuous Galerkin for Navier-Stokes Viscous Terms

Marcus Lo* and Bram van Leer†

*University of Michigan, Ann Arbor, MI 48109–2140 USA*

## Abstract

**Recovery-based discontinuous Galerkin (RDG) is presented as a new generation of discontinuous Galerkin (DG) methods for diffusion. The recovery concept is easily understood and integrated; in particular, RDG does not require additional data structures for surrogate variables and lifting operators, nor intricate manipulation of the partial differential equations, as is the case with Local Discontinuous Galerkin. In our previous papers we have illustrated RDG's remarkable accuracy and stability for scalar diffusion problems via numerical experiment and Fourier analysis. This paper demonstrates RDG's ability to solve the 2-D Navier-Stokes equations. We extend the original recovery concept for the diffusion flux; recovery is now used for solution enhancement. Solution enhancement via recovery is attractive because the subroutines can be used repetitively until a desired level of enhancement is achieved. We then present numerical results for a 2-D nonlinear equation and the Navier-Stokes equations. Lastly, we compare the various RDG schemes by Fourier analysis.**

## I. Brief history of Recovery-based discontinuous Galerkin

Reed and Hill[1] introduced discontinuous Galerkin (DG) for neutron mass-transport equations in 1973. The discontinuous basis functions in DG worked well for advection equations where discontinuities such as shocks, contact discontinuities and slip occur. Using a discontinuous solution representation to emulate discontinuous flow seems logical; however, using a discontinuous solution to represent the smooth diffusion process is completely unnatural. This dilemma stems from the diffusion flux being proportional to the solution gradient at the cell interface, where neither the solution nor its derivatives are well defined. Discontinuous basis functions became a double-edged sword that has plagued the DG community for a long time.

*Research Assistant, Department of Aerospace Engineering, AIAA Student Member.
†Professor, Department of Aerospace Engineering, AIAA Fellow.

Arnold[2] first attempted to alleviate the issue by penalizing the discontinuity in the interior penalty method (IPM) in 1982. Many years later, Baumann showed that the interior penalty was unnecessary if the coefficient of another penalty term in the scheme was given the opposite sign. Eventually, Van Leer[3] recognized all penalty-like schemes belong to a two-parameter family and explored the entire family for $K = 1$ (piecewise-linear) with regard to stability, consistency and accuracy. His search revealed there exists only one set of coefficients that yields a formally high-order and stable scheme; in contrast, RDG automatically determines such coefficients for high-order accuracy.

Bassi and Rebay[4] (1997) as well as Cockburn and Shu[5] (1998) introduced a different approach to diffusion by writing the 2nd-order diffusion equation as a system of two 1st-order equations for the solution and its first derivative. Their methods are better known as Bassi-Rebay (BR1) and local discontinuous Galerkin (LDG), respectively. The addition of a surrogate derivative variable and lifting operators increases the memory requirement. Furthermore, stability contants must be handpicked by the user for maximum accuracy, which hints at the lack of a physical motivation besides enabling mathematical proofs.

The original RDG[6] was designed to improve upon the diffusion flux for the scalar Laplace equation by a quantum leap. We applied integration by parts twice (hence the name RDG-2x) to the weak equation and used the recovered function, $f$, as the interface value for the surface integrals. Further numerical analyses and experiments for RDG are found in our previous papers.[7,8]

In this paper we extend the recovery concept to include solution enhancement: a method to increase the order of the solution polynomial. Park at el.[9] experimented in 2008 with solution enhancement using the recovery concept over a large stencil. We differentiate ourselves by focusing on recovery between two cells only (binary recovery). Binary recovery involves far fewer cells, significantly reducing the size of matrix operations and allowing for effective parallelization. Solution enhancement increases the accuracy of interior volume integrals and overcomes the face-normal bias of binary recovery (further explained in Section 3).

In order to tackle the nonlinear viscous terms found in the Navier-Stokes equations, the new generation of RDG schemes is based on different weak forms of the governing equations, and uses solution enhancement. We depart from RDG-2x because for nonlinear PDE's twice integrating by parts is not practical; instead, we focus on single integration by parts (1x) and no integration by parts at all (0x) when forming the weak equation. The schemes to be considered in this paper are RDG-1x+, RDG-0x+, RDG-1x++, and RDG-1x++CO, where the "+" and "CO" stand for solution enhancement and Cartesian optimization, respectively.

This paper first reviews the original recovery concept for the diffusion flux in Section 2. We then extend the recovery concept to include solution enhancement in Section 3 and differentiate between interior-solution enhancement and recovered-function enhancement. Next, we present numerical results for both 2-D nonlinear diffusion and 2-D Navier-Stokes viscous terms. Finally, we include Fourier analysis to compare various RDG schemes on the basis of the diffusion-shear equation.

## II. Review of the Original Recovery Concept: Diffusion Flux

We consider a time-dependent system of conservation laws with viscous terms only,

$$U_t = \nabla \cdot F\left(U, \nabla U\right), \tag{1}$$

in a domain $\Omega$, with conserved variables $U$, and viscous flux $F$. Let $v$ be a test function and $u$ be the discretized solution of interest. In a DG formulation both $v$ and $u$ share the same finite-element space, e.g. $v \in V$ and $u \in V$, where $V$ is the space of polynomial functions of degree at most $K \geq 0$. Next, we discretize the domain into cells $\Omega_i$. The solution $U$ is replaced by $u_i$, and the governing equations are tested with $v_i$ to obtain the DG formulation:

$$\begin{aligned}\int_{\Omega_i} (v_k)_i \; u_{i,t} \, d\Omega &= \int_{\Omega_i} (v_k)_i \, \nabla \cdot F\left(u_i, \nabla u_i\right) \, d\Omega, \quad k = 1, .., (K+1)^d, \\ &= \oint_{\partial\Omega_i} (v_k)_i \; F\left(u_i, \nabla u_i\right) \cdot \hat{n} \, d\partial\Omega \\ &\quad - \int_{\Omega_i} \nabla \left(v_k\right)_i \cdot F\left(u_i, \nabla u_i\right) \, d\Omega, \quad k = 1, .., (K+1)^d. \end{aligned} \tag{2}$$

Here, $\partial\Omega$ represents the cell boundary of $\Omega$, and $\hat{n}$ is the outward normal. $(K+1)^d$ stands for the number of tensor-product basis functions involved for physical dimension $d$. The solution along $\partial\Omega$ is undefined and we proceed with the recovery method to obtain an unique function along $\partial\Omega$.

The original recovery concept is simple in terms of physical interpretation and elegant in terms of mathematical formulation. Our objective is to "recover" a smooth function from the discontinuous solutions between two cells. Similar to fitting a function through points, recovery fits a function through the solution spaces of two abutting cells. We define $f_{(L,R)}$ to be the smooth continuous recovered function spanning the union of the two cells, $\Omega_L$ and $\Omega_R$, with shared interface, $S_{L,R}$. Here the subscripts $L$ and $R$ denote the left and right side of an interface respectively. We introduce the recovery concept in mathematical pretext. The recovered function is uniquely determined by making it indistinguishable from the discretized solution in the weak sense,

$$\begin{aligned}\int_{\Omega_L} (v_k)_L \, u_L d\Omega &= \int_{\Omega_L} (v_k)_L \, f_{(L,R)} d\Omega, \quad k = 1, .., (K+1)^d, \\ \int_{\Omega_R} (v_k)_R \, u_R d\Omega &= \int_{\Omega_R} (v_k)_R \, f_{(L,R)} d\Omega, \quad k = 1, .., (K+1)^d. \end{aligned} \tag{3}$$

¿From here on all equations with $v_k$ are applied to the complete $V$ space and we drop the notation "$k = 1...(K+1)^d$" for simplicity, unless stated otherwise. The set of recovery equations provides $2(K+1)^d$ equations to solve for the unknown coefficients, $b_k$:

$$f_{(L,R)} = \sum_{k=0}^{2(K+1)^d} b_k \omega_k, \quad \omega \in W. \tag{4}$$

where $\omega_k$ is the recovery basis in $W$, a polynomial space of degree at most $2(K+1)^d \geq 0$; for more information see.[10] We perform binary recovery at every interface and insert these recovery functions into the surface integrals of the viscous terms of the governing PDE's

$$\int_{\Omega_i} (v_k)_i \; u_{i,t} \, d\Omega = \oint_{\partial\Omega_i} (v_k)_i \; F\left(f_j, \nabla f_j\right) \cdot \hat{n} \, d\partial\Omega - \int_{\Omega_i} \nabla \left(v_k\right)_i \cdot F\left(u_i, \nabla u_i\right) \, d\Omega. \tag{5}$$

The recovery concept provides an intuitive way to uniquely determine the diffusion flux; in the next section we introduce another usage of the recovered functions.

## III. Extension of recovery concept: solution enhancement

Recovery was first conceptualized for DG diffusion; however, there is a much broader application of the recovered function. Solution enhancement is the operation to increase the polynomial order of a solution based on information from the surrounding cells. Very often we need to enhance the polynomial order of the solution to compensate for the reduction of polynomial order when a derivative of the solution is taken. The reduction of polynomial order leads to negative effects on accuracy and stability of a scheme as seen in the RDG-1x.

Park at el.[9] first experimented with solution enhancement over a large stencil in the cell-centered RDG (cRDG) scheme. When dealing with a large multi-scale system, the most resource-consuming part of the scheme is the implicit time-marching algorithm. The number of equations to solve increases rapidly with the polynomial order of the scheme; hence, cRDG seeks to increase the order of the scheme without increasing the number of solution coefficients. cRDG recovers a new interior solution that satisfies all weak equations of all neighboring cells. This is very similar to the reconstruction process in a finite-volume code, except cRDG works for $K \geq 0$.

The new RDG schemes make use of binary recovery for solution enhancement. Our approach results in a smaller systems of equations and efficiently recycles the recovered function: the recovered function is used for both diffusion flux and solution enhancement. Consider a rough comparison of the numerical work required for solution enhancement between RDG and cRDG on a Cartesian grid. cRDG solves a system of $N_{cRDG,d}\,(K+1)^d$ equations, where $N_{cRDG}$ is the number of cells in the cRDG stencil and $d$ is the physical dimension. RDG solves $2\,(K+1)^d$ equations per binary recovery for $N_{RDG,d}$ interfaces per cell, followed by a solution enhancement step with $(K+1)^d + N_{RDG,d}(K+1)^{d-1}$ equations. Assuming $\frac{n+n^2}{2}$ floating point operations (flops) to invert an $(n \times n)$ matrix via LU-decomposition, Table 1 shows the approximate amount of computational work required for solution enhancement by RDG and cRDG. Clearly, cRDG becomes increasingly expensive for higher polynomial order and physical dimension.

| 1-D | | |
|---|---|---|
| $K$ | RDG | cRDG |
| 1 | 30 | 21 |
| 2 | 57 | 45 |
| 3 | 93 | 78 |
| 4 | 138 | 120 |

| 2-D | | |
|---|---|---|
| $K$ | RDG | cRDG |
| 1 | 222 | 666 |
| 2 | 915 | 3321 |
| 3 | 2640 | 10440 |
| 4 | 6135 | 25425 |

| 3-D | | |
|---|---|---|
| $K$ | RDG | cRDG |
| 1 | 1344 | 23436 |
| 2 | 12231 | 266085 |
| 3 | 62416 | $1.49 \times 10^6$ |
| 4 | 226200 | $5.69 \times 10^6$ |

**Table 1. Flops comparison between RDG and cRDG for solution enhancement on a Cartesian grid, where $(N_{RDG}, N_{cRDG})$ is $(2,3)$, $(4,9)$, and $(6,27)$ for 1-D, 2-D, and 3-D, respectively. RDG in 2-D and 3-D is more than an order of magnitude cheaper than cRDG.**

Although the difference in computational work required between RDG and cRDG is significant, cRDG's enhanced solution contains the complete set of cross-derivatives which are useful if the PDE requires them. RDG's enhanced solution does not span a complete higher-order polynomial space (see Figure 2); however, the Navier-Stokes equations after

integration by parts once no longer contain cross-derivatives. Ideally, the level of solution enhancement should be optimized for the PDE being solved.

The recovery function proves to be a valuable tool; it is used for both diffusion flux and solution enhancement. One could say the recovery concept represents a new green movement in CFD, in which both subroutines and results are reused to obtain a high order of accuracy. Starting off with the basic information of $u$ in each cell, a binary recovery code is first applied to produce all $f$'s along the cell interfaces. A unifying code is then applied to acquire an enhanced solution, $\hat{u}$, in each cell (see Figure 1). We can apply the same binary recovery code again to get an enhanced recovery function, $\hat{f}$, and follow it up with the same solution enhancement code to get a doubly enhanced solution, $\hat{\hat{u}}$. Furthermore, $\hat{f}$ can be used in an enhanced diffusion flux. Solution enhancement comes at the expense of a growing stencil; therefore, the optimal level of enhancement must be chosen for each PDE. The process can be repeated until the desired level of enhancement dictated by the PDE's is achieved. The following sections detail two types of solution enhancement necessary for approximating the Navier-Stokes viscous terms: the enhancement of the discretized solution and then the enhancement of the recovered function.

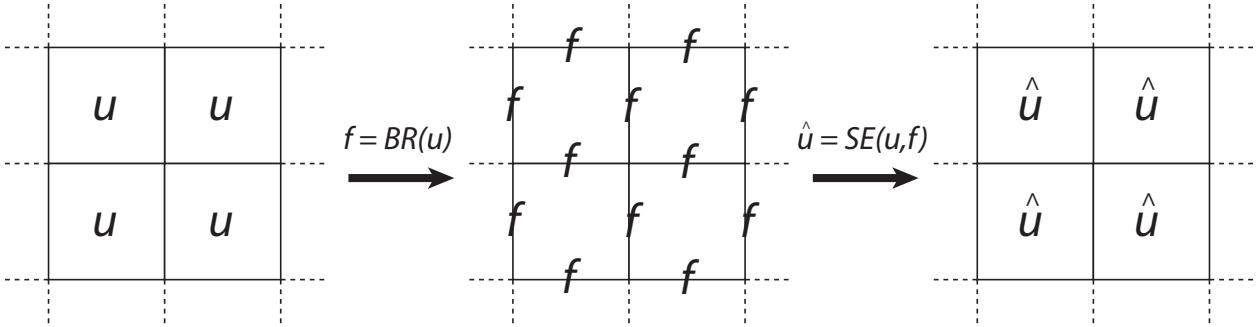


**Figure 1. Recovery concept is used for both diffusion flux and solution enhancement. BR stands for binary recovery; SE stands for solution enhancement. The cycle of BR followed by SE can be repeated over and over again until the desired level of enhancement is achieved.**

### III.A. Interior-solution enhancement

We skip the procedure to acquire $f$ via binary recovery (see previous papers) and focus on the solution-enhancement step. The enhanced solution, $\hat{u}$, replaces the original solution $u$ in the volume integral,

$$\int_{\Omega_i} (v_k)_i \; u_{i,t} \, d\Omega \;\; = \;\; \oint_{\partial\Omega_i} (v_k)_i \; F\left(f_j, \nabla f_j\right) \cdot \hat{n} \, d\partial\Omega - \int_{\Omega_i} \nabla \left(v_k\right)_i \cdot F\left(\hat{u}_i, \nabla \hat{u}_i\right) \, d\Omega. \tag{6}$$

We require $\hat{u}$ to share all original moments with $u$, and in addition, to share all moments with $f$ along the cell boundaries. The equations for solution enhancement in a general case are as follows:

$$\int_{\Omega_i} (v_k)_i \; u_i \, d\Omega = \int_{\Omega_i} (v_k)_i \; \hat{u}_i \, d\Omega, \tag{7}$$

$$\oint_{\partial\Omega_i} (v_m)_i \; \hat{u}_{i,t} \, d\partial\Omega = \oint_{\partial\Omega_i} (v_m)_i \; f_j \, d\partial\Omega, \quad m = 1...(K+1)^{d-1}, \tag{8}$$

where $m$ is the index to a $(d-1)$-dimensional test function. The second equation uses the test functions from a reduced space because certain coordinate variables are fixed at the cell boundary. The only task left is to define the basis functions of $\hat{u}$, which lies in a larger function space $\hat{V}$. The following sections show 1-D and 2-D examples on a Cartesian grid.

### *III.A.1. 1-D solution-enhancement basis functions*

In 1-D, Eqn. 8 becomes a strong interpretation; $\hat{u}$ is required to equal $f$ in the two points at the left and right end of the cell. Let $P(K)$ be the function space with polynomials of degree of at most $K \geq 0$. If $u \in P(K)$, then $\hat{u} \in P(K+2)$.

### *III.A.2. 2-D solution-enhancement basis functions*

In 2-D, $\hat{u}$ is required to satisfy $K+1$ moments of $f$ along a line. The choice of basis functions for $\hat{u}$ is best illustrated through Figure 2. The positive $x$-axis represents increasing polynomial order in $x$, and the negative $y$-axis represents increasing polynomial order in $y$. Consider the solid-line square to represent the original basis functions of $u$. The left and right boundaries of a cell add two $(K+1)$-element columns to the right of the square, and the top and bottom boundaries add two $(K+1)$-element rows to the bottom of the square.

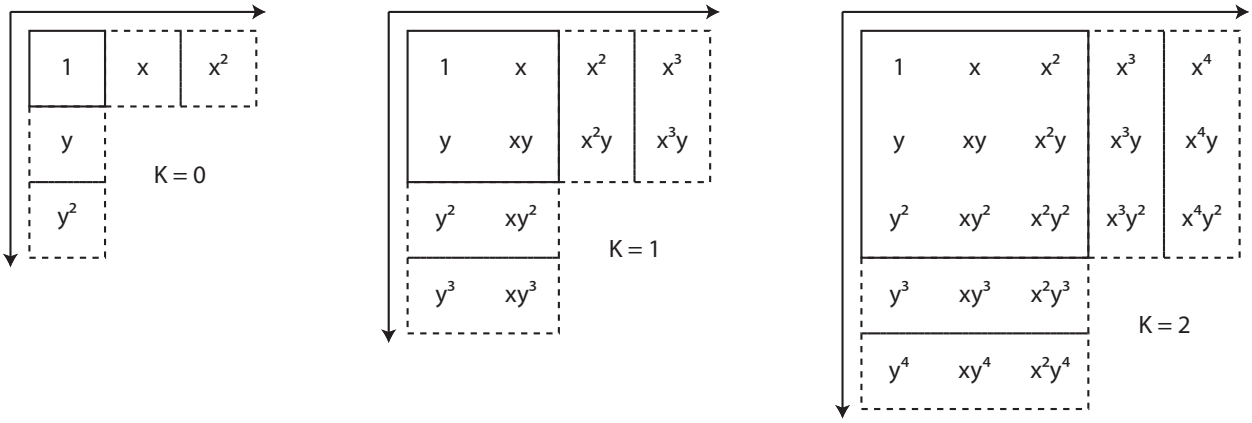


**Figure 2. Basis functions for $\hat{u}$ in 2-D Cartesian grid for $K = 0, 1,$ and $2$ from left to right.**

### *III.A.3. RDG-0x+*

We briefly introduce an experimental scheme called RDG-0x+. As the name implies, the scheme acts upon the original weak form of the PDE's without any integration by parts, and replaces $u$ with a special enhanced solution, $\tilde{u}$,

$$\int_{\Omega_i} (v_k)_i \; u_{i,t} \, d\Omega \;\; = \;\; \int_{\Omega_i} (v_k)_i \, \nabla \cdot F\left(\tilde{u}_i, \nabla \tilde{u}_i\right) \, d\Omega. \tag{9}$$

The construction of $\tilde{u}$ is similar to that of $\hat{u}$; however, $\tilde{u}$ also shares the normal derivative, $f_n$, of $f$,

$$\oint_{\partial\Omega_i} (v_m)_i \; \tilde{u}_{n,i} \, d\partial\Omega = \oint_{\partial\Omega_i} (v_m)_i \; f_{n,j} \, d\partial\Omega, \quad m = 1...(K+1)^{d-1}. \tag{10}$$

A comparison between RDG-1x+ and RDG-0x+ reveals many interesting differences. The number of terms RDG-0x+ deals with is significantly less in 2-D and 3-D, which translates to greater computational speed. However, this is offset by a more complicated solution-enhancement procedure that is more expensive. Users of RDG may choose between dealing with complicated terms in the 1x form, or spend more computational work to acquire $\tilde{u}$. Currently, this experimental scheme only works with a PDE without cross derivatives.

An interesting note: Huynh[11] analyzed various DG schemes for the scalar diffusion equation and found RDG-2x, albeit expensive, to be the most accurate and to have the smallest eigenvalue range. Smaller eigenvalues in the spatial discretization allow an explicit time-marching scheme to take larger time steps. For the scalar Laplace operator, both RDG-0x+ and RDG-1x+ expands into the original RDG-2x scheme; thus, both RDG-0x+ and RDG-1x+ come from the same fine pedigree as RDG-2x, with the additional ability to solve nonlinear equations.

### *III.A.4. Numerical results for 2-D Nonlinear Viscous Terms*

The following numerical experiment is designed to isolate the need for recovery-function enhancement, and focuses on solution enhancement only. We consider the scalar 2-D nonlinear diffusion equation with a source term,

$$u_t = e^{-u^2} \left( u_{xx} + u_{yy} \right) + S\left(t\right), \tag{11}$$

on the domain $x \in [0,1]$, where $S\left(t\right)$ is a time-dependent source term determined by Mathematica software for the manufactured solution

$$u\left(x,t\right) = \sin\left(2\pi x\right) \sin\left(2\pi y\right) e^{-t}. \tag{12}$$

Table 2 shows the $L_2$-error of the cell average at $t = 1$ for RDG-1x+ and RDG-0x+. The RDG spatial discretizations are coupled with the 3rd, 4th, and 5th-order explicit Runge-Kutta temporal schemes for $K = 1, 2$, and 3, respectively. (Note that $O(\Delta t) = O\left((\Delta x)^2\right)$ for an explicit diffusion scheme.) RDG-0x+ is slightly more accurate for $K = 2$ and 3, while both schemes obtain the same order of accuracy (O.O.A) as the original RDG-2x scheme for scalar diffusion.

## III.B. Recovered-function enhancement

Using the recovered function for the viscous fluxes in 1-D problems is sufficient; however, in multi-dimensional problem, the recovered function is not accurate in the face-tangential direction of the cell boundary. The need for accurate representation of the solution's face-tangential derivative, such as appear in the Navier-Stokes shear terms, is imperative for achieving overall high-order accuracy. Our newest scheme RDG-1x++ performs binary recovery over the new enhanced solution $\hat{u}$ from the previous section to obtain an enhanced recovered function $\hat{f}$ as shown in Figure 3.

| RDG-1x+ | | | |
|---|---|---|---|
| $K$ | Cells | $L_2$-error | O.O.A. |
| 1 | $18 \times 18$ | $1.24e-05$ | |
| | $24 \times 24$ | $3.96e-06$ | 4.0 |
| | $30 \times 30$ | $1.63e-06$ | 4.0 |
| | $36 \times 36$ | $7.89e-07$ | 4.0 |
| | | | |
| 2 | $18 \times 18$ | $1.50e-10$ | |
| | $24 \times 24$ | $1.65e-11$ | 7.7 |
| | $30 \times 30$ | $2.87e-12$ | 7.8 |
| | $36 \times 36$ | $6.70e-13$ | 8.0 |
| | | | |
| 3 | $6 \times 6$ | $8.66e-08$ | |
| | $8 \times 8$ | $6.35e-09$ | 9.1 |
| | $10 \times 10$ | $7.14e-10$ | 9.8 |
| | $12 \times 12$ | $1.08e-10$ | 10.3 |

| RDG-0x+ | | | |
|---|---|---|---|
| $K$ | Cells | $L_2$-error | O.O.A. |
| 1 | $18 \times 18$ | $1.23e-05$ | |
| | $24 \times 24$ | $3.96e-06$ | 4.0 |
| | $30 \times 30$ | $1.63e-06$ | 4.0 |
| | $36 \times 36$ | $7.91e-07$ | 4.0 |
| | | | |
| 2 | $6 \times 6$ | $5.01e-11$ | |
| | $12 \times 12$ | $5.16e-12$ | 7.9 |
| | $18 \times 18$ | $9.00e-13$ | 7.8 |
| | $24 \times 24$ | $2.60e-13$ | 6.8 |
| | | | |
| 3 | $6 \times 6$ | $2.46e-09$ | |
| | $8 \times 8$ | $7.07e-11$ | 12.3 |
| | $10 \times 10$ | $6.02e-12$ | 11.0 |
| | $12 \times 12$ | $9.27e-13$ | 10.3 |

**Table 2.** $L_2$**-error of the cell average for the RDG-1x+ and RDG-0x+ schemes.**

As mentioned before, the subroutine for recovery is the same for any level of enhancement, with the equations for the enhanced recovered equation similar to that of $f$:

$$\begin{aligned} \int_{\Omega_L} (\hat{v}_k)_L \, \hat{u}_L d\Omega &= \int_{\Omega_L} (\hat{v}_k)_L \, \hat{f}_{(L,R)} d\Omega, \quad \forall k \text{ such that } \hat{v}_k \in \hat{V}, \\ \int_{\Omega_R} (\hat{v}_k)_R \, \hat{u}_R d\Omega &= \int_{\Omega_R} (\hat{v}_k)_R \, \hat{f}_{(L,R)} d\Omega, \quad \forall k \text{ such that } \hat{v}_k \in \hat{V}. \end{aligned} \tag{13}$$

Here, $\hat{v}$ is a basis function of $\hat{V}$ and $\hat{f}$ belongs to a different function space $\hat{W}$,

$$\hat{f}_{(L,R)} = \sum_k \hat{b}_k \hat{\omega}_k, \quad , \hat{\omega} \in \hat{W}, \tag{14}$$

where $\hat{\omega}$ denotes the basis function for $\hat{f}$ (see[10] for more information about recovery bases).

This extra layer of binary recovery comes at a hefty cost due to the increased stencil size, which effectively decreases the maximum allowable time-step in an explicit scheme, or increases connectivity cost in an implicit scheme. Figure 4 compares the stencil size of various RDG schemes. It is worth noting that both Compact DG (CDG)[12] and BR2 share the compact stencil of RDG-1x, but these schemes cannot handle PDE's with cross-derivatives at the $K = 0$ level. Clearly, cRDG is the most expensive out of all the schemes present, while RDG-1x++CO is the optimal scheme to handle PDE's with cross-derivatives for all $K \geq 0$, if the grid is Cartesian.

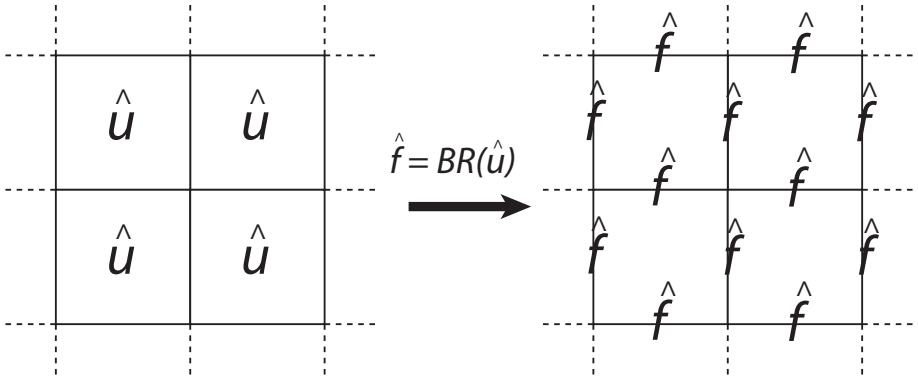


**Figure 3. The recovered function is inaccurate in the face-tangential direction. We apply binary recovery on top of $\hat{u}$ to get an enhanced recovered function $\hat{f}$ to improve on the accuracy of $f$ in the face-tangential directions.**

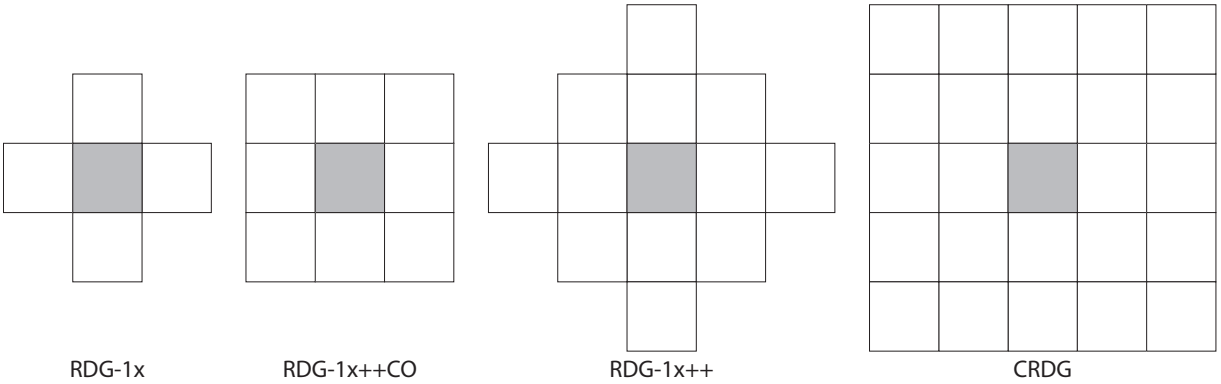


**Figure 4. The 2-D stencils for various RDG schemes. Stencil size has direct influence on the time-step of explicit time-marching schemes, and also on the matrix density of implicit time-marching schemes.**

### *III.B.1. RDG-1x++CO (Cartesian Optimization)*

We draw our inspiration to optimize the RDG-1x++ scheme from dimension-splitting techniques found in.[13,14] We recognize the solution-enhancement step for 2-D Cartesian grids can be factorized into 1-D steps. The convenience of such a factorization is twofold. Firstly, the same operator developed for 1-D is reusable for 2-D case; secondly, multi-dimensional codes reduce to a sequence of 1-D sweeps.

On a Cartesian grid the enhanced recovered function is too accurate in face-normal direction; hence we eliminate the extraneous information in the face-normal direction by using fewer moments in Eqn 7. Consider the stencils in Figure 5 used to obtain the enhanced recovered function for RDG-1x++ and RDG-1x++CO. The thick solid line indicates the interface of interest. Since we do not need more information in the face-normal direction, we can optimize RDG-1x++ by taking away the two cells furthest away from the interface in the face-normal direction. This optimization techniques require different sets of enhanced solutions $\hat{u}^x$ and $\hat{u}^y$, which are solution-enhanced in the $x$-direction and $y$-direction, respectively.

This section describes the steps to acquire an enhanced recovered function on a vertical interface from $\hat{u}^y$, and because this is a factorization technique, the same steps can be applied to get an enhanced recovered function on a horizontal interface from $\hat{u}^x$. We first cycle through all cells to get a new enhanced solution $\hat{u}^y$. Figure 6 shows a $K = 1$ example beginning from the left. The subscripts $M$, $L$, and $R$, stand for middle, left and right,

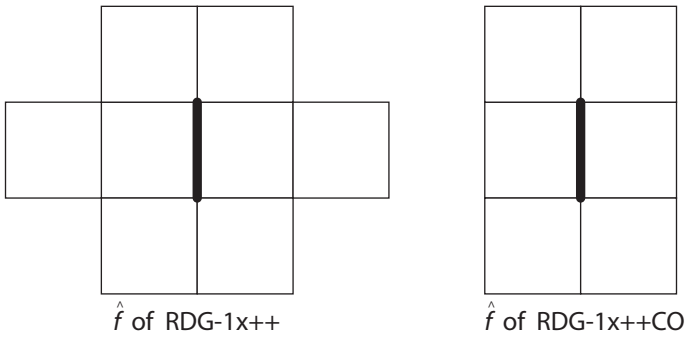


**Figure 5. The stencils of the enhanced recovered function for RDG-1x++ and RDG-1x++CO on the left and right, respectively.**

respectively. We require $\hat{u}_M^y$ to share all moments of $u_M$, and in addition, share the moments (without the $y$ basis) of the recovered functions on the top and bottom faces of the cell. The resulting $\hat{u}_M^y$ will be enhanced in the $y$-direction only, as shown in the middle of Figure 6, where the solution within the cell now contains quadratic and cubic information in the $y$-direction. Our next step is to recover the enhanced recovered function, $\hat{f}$, from the vertically enhanced solutions on the left and right of an interface. Using the standard binary recovery technique with $\hat{u}_L^y$ and $\hat{u}_R^y$ as inputs, the resulting $\hat{f}$ is more accurate in both $r$ and $s$ directions (see right-most frame of Figure 6). For higher $K$, recovery in the face-normal direction will still be more accurate than in the face-parallel direction; nevertheless, the slight improvement in the $s$-direction is sufficient for high-order accuracy.

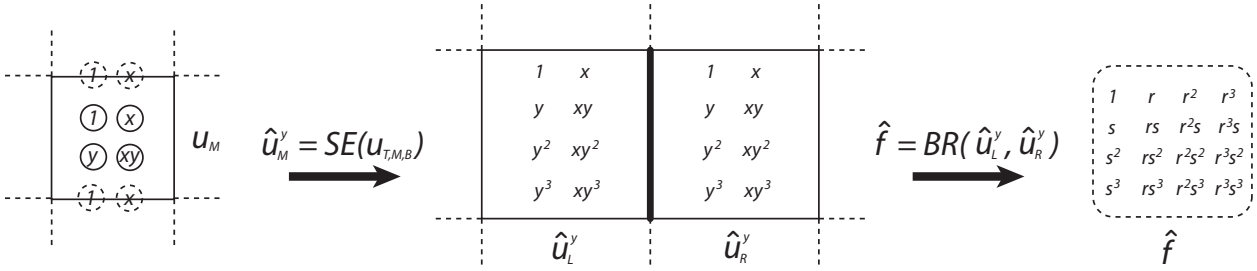


**Figure 6. Reduced-accuracy $y$-recovery, followed by standard $x$-recovery, to create an enhanced recovered function $\hat{f}$ for use at an interface along the $y$-direction.**

## IV. Solution enhancement on unstructured grids

In our 2009 conference paper on DG for diffusion[15] we discussed the concept of solution enhancement due to Nourgaliev et al.,[9] in which information from all neighboring cells (neighbors that share at least one point with the central cell) is drawn in via a large set of weak interpolation equations (cell-centered recovery). As is evident from the previous section, we have come away from this technique, for practical reasons, and are now only allowing enhancement using cell-face distributions obtained by interface-centered recovery.

What order of accuracy could we expect from such an approach on an unstructured grid? Such a grid does not agree with a tensor-product basis, so one would see the maximum order attainable drop from $3p+1$ or $3p+2$ to, perhaps, $2p+2$, which we found earlier[7,8] for a minimal basis of order $p$ used on a Cartesian grid. In the latter calculations, though, we did

not pull in corner-cell information through solution enhancement, so the order ceiling might be higher than $2p+2$ for a stencil including all neighbors.

An unstructured grid made of simplices lacks the chief benefit offered by a Cartesian grid, i.e., fortuitous cancelation of truncation errors among fluxes; other than that there seems to be no formal reason why the accuracy on an unstructured grid should be further lowered. On the contrary, on a simplex grid the information is packed more densely; for instance, a 2-D triangular cell has more direct neighbors than a Cartesian cell. Therefore, solution enhancement using all these neighbors would potentially lead to a higher order of accuracy than on a Cartesian grid. The big question is whether the information in the neighboring cells can be accessed and incorporated in a practical algorithm.

Consider Figure 7, taken from;[15] part (b) it shows the "butterfly" stencil for a 2-D flux calculation. Before the final flux between cells $\mathcal{A}$ and $\mathcal{B}$ is computed by recovery, the solution in these cells may first be enhanced using solution values recovered at the interfaces between the pairs of cells $(\mathcal{A},\mathcal{C})$, $(\mathcal{A},\mathcal{E})$, $(\mathcal{B},\mathcal{D})$, $(\mathcal{B},\mathcal{F})$ and even $(\mathcal{A},\mathcal{B})$. But neighbors of $\mathcal{C}$, $\mathcal{E}$, $\mathcal{D}$ and $\mathcal{F}$ that touch $\mathcal{A}$ or $\mathcal{B}$ are not included, leaving gaps in a stencil that at best would fully enclose the central cell. Further analysis and numerical experiments are now needed to uncover the effect of these gaps on the accuracy and stability of the unstructured DG method.

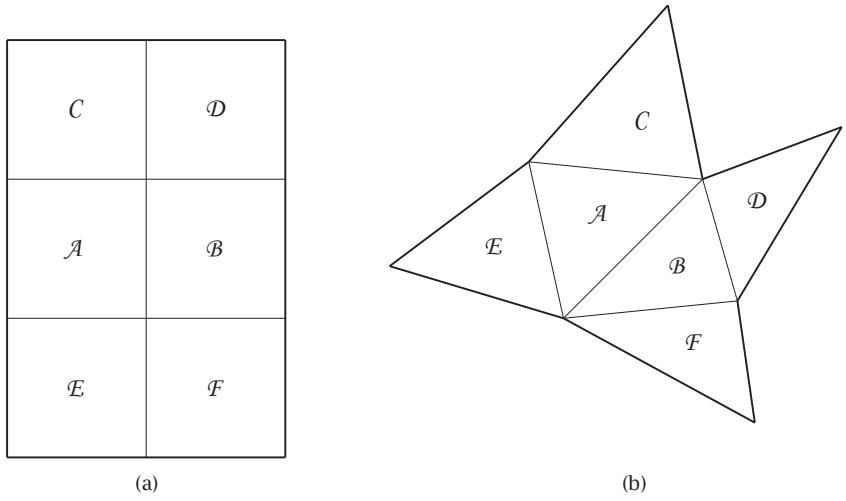


**Figure 7. Extended Cartesian (a) and triangular "butterfly" (b) stencils for more isotropic interface-based recovery.**

## IV.A. Numerical Results for 2-D Navier-Stokes Viscous Terms

In order to isolate the numerical scheme for diffusion, we remove the Euler terms from the Navier-Stokes equations and use RDG for the viscous fluxes. As a result the density equation drops out resulting in the following system:

$$\begin{pmatrix} \rho u \\ \rho v \\ \rho E \end{pmatrix}_t = \begin{pmatrix} \tau_{xx} \\ \tau_{yx} \\ u\tau_{xx} + v\tau_{xy} - q_x \end{pmatrix}_x + \begin{pmatrix} \tau_{xy} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} - q_y \end{pmatrix}_y , \tag{15}$$

Our numerical test case is for a manufactured solution with approximate physics; therefore, the values of the physical constants (such as Prandtl number, gas constant, and specific-heat

ratio) are fixed. The simplified equations are as follows. The shear stresses with Stokes' hypothesis are

$$\tau_{xx} = \frac{2}{3}\mu\left(2u_x - v_y\right), \tag{16}$$

$$\tau_{yy} = \frac{2}{3}\mu\left(2v_y - v_x\right), \tag{17}$$

$$\tau_{xy} = \tau_{yx} = \mu\left(u_y + v_x\right), \tag{18}$$

and the heat fluxes are,

$$q_x = \frac{19}{4}\mu\left(\frac{P}{\rho}\right)_x, \tag{19}$$

$$q_y = \frac{19}{4}\mu\left(\frac{P}{\rho}\right)_y. \tag{20}$$

We use a viscosity coefficient similar to that of Sutherland's law,

$$\mu = \left(\frac{2}{T+1}\right)T^{\frac{3}{2}}, \tag{21}$$

where $T$ is the temperature from the ideal gas law, $P = \rho RT$, with $R = 1$. We consider the following manufactured solutions,

$$\rho = U_0\left(x, y, t\right) = 2, \tag{22}$$

$$\rho u = U_1\left(x, y, t\right) = \sin\left(2\pi x\right)\sin\left(2\pi y\right)e^{-t}, \tag{23}$$

$$\rho v = U_2\left(x, y, t\right) = \sin\left(2\pi x\right)\sin\left(2\pi y\right)e^{-t}, \tag{24}$$

$$\rho E = U_3\left(x, y, t\right) = 5{+}\sin\left(2\pi x\right)\sin\left(2\pi y\right)e^{-t}, \tag{25}$$

and generate the appropriate source terms with Mathematica. We consider a periodic boundary unit square domain with $x \in [0, 1]$ and $y \in [0, 1]$. The RDG spatial discretizations are coupled with the 3rd, 4th, and 5th-order explicit Runge-Kutta temporal schemes for $K = 1, 2$, and 3, respectively. The Table 3 shows the $L_2$-error of the cell average for both RDG-1x++ and RDG-1x++CO at $t = 1$. It is worth adding that RDG-1x++CO is roughly three to four times faster than RDG-1x++.

### IV.B. Numerical Fourier Analysis of 2-D RDG Schemes

Now that all the new RDG schemes have been revealed, we present Fourier-analysis results for all RDG schemes on the 2-D Cartesian grid. The scalar equation of interest is the diffusion-shear equation including a Laplacian and a cross-derivative term,

$$u_t = u_{xx} + u_{yy} + \alpha u_{xy}, \tag{26}$$

where $\alpha$ is a constant, with the requirement $-2 \leq \alpha \leq 2$ for the PDE to be stable. The cross-derivative term is included to mimic the behavior of certain Navier-Stokes viscous terms. The numerical schemes should approximate the Fourier operator,

$$u_{xx} + u_{yy} + \alpha u_{xy} \simeq -\left(\frac{2\beta^2}{h^2} + \frac{\alpha\beta^2}{h^2}\right)u, \tag{27}$$

| RDG-1x++ | | | | RDG-1x++CO | | | |
|---|---|---|---|---|---|---|---|
| $K$ | Cells | $L_2$-error | O.O.A. | $K$ | Cells | $L_2$-error | O.O.A. |
| 1 | $6 \times 6$ | 0.000544 | | 1 | $6 \times 6$ | 0.000487 | |
| | $12 \times 12$ | $3.79e-05$ | 3.8 | | $12 \times 12$ | $3.66e-05$ | 3.7 |
| | $18 \times 18$ | $7.61e-06$ | 4.0 | | $18 \times 18$ | $7.49e-06$ | 3.9 |
| | $24 \times 24$ | $2.42e-06$ | 4.0 | | $24 \times 24$ | $2.40e-06$ | 4.0 |
| | | | | | | | |
| 2 | $12 \times 12$ | $1.91e-08$ | | 2 | $12 \times 12$ | $1.17e-09$ | |
| | $16 \times 16$ | $3.89e-09$ | 5.5 | | $16 \times 16$ | $7.52e-11$ | 9.5 |
| | $20 \times 20$ | $1.08e-09$ | 5.7 | | $20 \times 20$ | $1.27e-11$ | 8.0 |
| | $24 \times 24$ | $3.76e-10$ | 5.8 | | $24 \times 24$ | $4.68e-12$ | 5.5 |
| | | | | | | | |
| 3 | $6 \times 6$ | $1.32e-09$ | | 3 | $6 \times 6$ | $3.27e-10$ | |
| | $8 \times 8$ | $8.61e-11$ | 9.5 | | $8 \times 8$ | $2.39e-11$ | 9.1 |
| | $10 \times 10$ | $9.80e-12$ | 9.7 | | $10 \times 10$ | $2.92e-12$ | 9.4 |
| | $12 \times 12$ | $1.62e-12$ | 9.9 | | $12 \times 12$ | $4.93e-13$ | 9.8 |

**Table 3. $L_2$-error of the cell average for the RDG-1x++ and RDG-1x++CO schemes.**

where $\beta$ is the frequency of the Fourier mode assumed equal in the $x$- and $y$-directions, and $h$ is the cell width. Table 4 shows the results for $\alpha = 0$ (pure Laplacian), and Table 5 shows the results for $\alpha = 1$. The three numbers shown are the largest real eigenvalue $\max(\mathrm{Re}(\lambda))$, the largest imaginary eigenvalue $\max(\mathrm{Im}(\lambda))$, and the order of accuracy. In designing diffusion schemes, we want the maximum real eigenvalue to be as small as possible to allow for a larger time-step and the imaginary component to be as close to zero as possible.

$(\max|\mathrm{Re}(\lambda)|, \max(\mathrm{Im}(\lambda)), \text{O.O.A.})$

| $K$ | RDG-2x | RDG-1x+, RDG-1x++CO | RDG-1x++ |
|---|---|---|---|
| 0 | (7.9/0/2) | (7.9/0/2) | (20.2/0/2) |
| 1 | (30.0/0/4) | (30.0/0/4) | (82.5/0/4) |
| 2 | (66.0/0/8) | (66.0/0/8) | (183.3/0/6) |
| 3 | (134.9/11.1/10) | (134.9/11.1/10) | (318.8/0/10) |
| 4 | (217.2/18.4/14) | (217.2/18.4/14) | (477.1/0/14) |
| 5 | (302.3/21.4/16) | (302.3/21.4/16) | (649.9/20.5/16) |

**Table 4. Fourier-analysis results for $\alpha = 0$ (pure Laplacian). Note that RDG-1x+, RDG-1x++CO, and RDG-2x are identical.**

RDG-2x was first presented in 2005 and represents our best scheme for the scalar Laplacian, with the smallest real eigenvalues and the highest order of accuracy. For the scalar Laplacian, both RDG-1x+ and RDG-1x++CO reduces to the RDG-2x scheme. RDG-1x++ has larger real eigenvalues due to its larger stencil.

In the case with cross-derivative, the older generation RDG schemes, RDG-2x, RDG-1x, and RDG-1x+, fail to approximate the cross-derivative term; they result in zeroth-order schemes. Of the two remaining schemes, RDG-1x++CO allows a twice-as-large time-step

$(\max|\mathrm{Re}(\lambda)|, \max(\mathrm{Im}(\lambda)), \text{O.O.A.})$

| $K$ | RDG-1x++ | RDG-1x++CO |
|---|---|---|
| 0 | (20.2/0/2) | (7.9/0/2) |
| 1 | (82.5/0/4) | (33.7/0.4/4) |
| 2 | (183.6/0.76/6) | (87.1/3.0/8) |
| 3 | (319.2/7.59/10) | (169.6/10.7/10) |
| 4 | (495.3/30.43/10) | (278.6/23.8/14) |
| 5 | (803.6/105.5/14) | (412.8/43.4/16) |

**Table 5. Fourier analysis results for $\alpha = 1$ (with cross-derivative). The maximum real eigenvalue of RDG-1x++CO is about half of that of RDG-1x++.**

and obtains a higher order of accuracy for certain $K$.

## V. Future Works and Conclusion

RDG is a simple and intuitive method for DG diffusion. The only extra information that needs to be stored is the recovered function $f$ at each interface, which is much less than for other schemes (LDG, CDG, BR1, BR2) with multiple lifting functions. In this paper, we show that the recovery concept is now used for both diffusion flux and solution enhancement. The level of enhancement is dependent on the nature of the PDE; the current RDG-1x++ and RDG-1x++CO schemes are optimized for the Navier-Stokes equation. Numerical results clearly demonstrate super-convergence on the 2-D Cartesian grid for Navier-Stokes viscous terms.

Our future goal is to extend RDG-1x++ scheme to unstructured triangular grids. Our criterion for order of accuracy on the Cartesian grid is ambitious; however, for an unstructured triangular grid, we aim for a simple compact scheme with order of accuracy of at least $K + 1$. Similarly to how LDG is optimized into CDG, or BR1 is optimized into BR2, the current RDG schemes for Navier-Stokes must be optimized for parallelization by reducing connectivity between cells.

## References


[1]Reed, W. and Hill, T., "Triangular mesh methods for the neutron transport equation," *Tech. Rep. LA-UR 73-479, Los Alamos National Laboratory*, 1973.

[2]Arnold, D., "An interior penalty finite element method with discontinuous elements," *SIAM Journal on Numerical Analysis*, Vol. 19, 1982, pp. 742–760.

[3]van Leer, B., Lo, M., Gitik, R., and Nomura, S., *"A Venerable Family of Discontinuous Galerkin Schemes for Diffusion Revisited"*, World Scientific Review Volume 9, 2010.

[4]Bassi, F. and Rebay, S., ""A High-Order accurate discontinuous finite element method for the numerical solution of the compressible navier-stokes equations"," *Journal of Computational Physics*, Vol. 131, 1997, pp. 267–279.

[5]Cockburn, B. and Shu, C., "The local discontinuous Galerkin method for time-dependent convection-diffusion systems," *Siam, Journal on Numerical Analysis*, Vol. 35, 1998, pp. 2440–2463.

[6]van Leer, B. and Nomura, S., "Discontinuous Galerkin for diffusion," *17th AIAA Computational Fluid Dynamics Conference, Tornoto, Ontario, Canada, AIAA Paper 2005-5108*, 2005.

[7]van Leer, B., Lo, M., and van Raalte, M., "A discontinuous Galerkin method for diffusion based on recovery," *18th AIAA Computational Fluid Dynamics Conference, Miami, Florida, USA, AIAA Paper 2007-4083*, June 2007.

[8]van Leer, B. and Lo, M., "Unification of Discontinuous Galerkin methods for advection and diffusion," *47th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, Florida, USA, AIAA Paper 2009-400*, 2009.

[9]Park, H., Nourgaliev, R., Mousseau, V., and Knoll, D., "Recovery Discontinuous Galerkin - Jacobian-Free Newton Krylov (rDG-JFNK) Method for All-Speed Navier-Stokes Equations," *International Conference on Computational Fluid Dynamics (ICCFD), Seoul, Korea*, 2008.

[10]van Raalte, M. and van Leer, B., "Bilinear forms for the Recovery-based Discontinuous Galerkin method for diffusion," *Communications in Computational Physics*, Vol. 5, 2008, pp. 683–693.

[11]Huynh, H., "A Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin for Diffusion," *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition, AIAA Paper 2009-0403*, 2009.

[12]Peraire, J. and Persson, P., "The compact discontinuous galerkin (CDG) method for elliptic problems," *SIAM Journal of Scientific Computating*, Vol. 30, 2008, pp. 1806–1824.

[13]van Leer, B., "Multidimensional explicit difference schemes for hyperbolic conservation laws," *Computational Methods in Applied Sciences and Engineering, VI*, 1984, pp. 493–497.

[14]van Leer, B., "Computational methods for ideal compressible flow," *In Von Karman Inst. for Fluid Dynamics Computational Fluid Dyn.*, Vol. 1, 1984, pp. 45.

[15]Lo, M. and van Leer, B., "Analysis and implementation of Recovery-based Discontinuous Galerkin for diffusion," *19th AIAA Computational Fluid Dynamics Conference, San Antonio, Texas, USA, AIAA Paper 2009-3786*, 2009.