A Hybrid Reconstructed Discontinuous Galerkin and Continuous Galerkin Finite Element Method for Incompressible Flows on Unstructured Grids

Aditya Pandare\(^1\) and Hong Luo\(^2\,*\)
North Carolina State University, Raleigh, NC, 27695, USA

A hybrid reconstructed discontinuous Galerkin and continuous Galerkin method based on an incremental pressure projection formulation, termed rDG(P\(_m\)P\(_n\))+CG(P\(_n\)) in this paper, is developed for solving the unsteady incompressible Navier-Stokes equations on unstructured grids. In this method, a reconstructed discontinuous Galerkin method (rDG(P\(_m\)P\(_n\))) is used to discretize the velocity and a standard continuous Galerkin method (CG(P\(_n\))) is used to approximate the pressure. The rDG(P\(_m\)P\(_n\))+CG(P\(_n\)) method is designed to increase the accuracy of the hybrid DG(P\(_n\))+CG(P\(_n\)) method and yet still satisfy Ladyženskaja-Babuška-Brezzi (LBB) condition, thus avoiding the pressure checkerboard instability. An upwind method is used to discretize the nonlinear convective fluxes in the momentum equations in order to suppress spurious oscillations in the velocity field. A number of incompressible flow problems for a variety of flow conditions are computed to numerically assess the spatial order of convergence of the rDG(P\(_m\)P\(_n\))+CG(P\(_n\)) method. The numerical experiments indicate that both rDG(P\(_0\)P\(_1\))+CG(P\(_1\)) and rDG(P\(_1\)P\(_2\))+CG(P\(_1\)) methods can attain the designed 2\(^{nd}\) order and 3\(^{rd}\) order accuracy in space for the velocity respectively. Moreover, the 3\(^{rd}\) order rDG(P\(_1\)P\(_2\))+CG(P\(_1\)) method significantly outperforms its 2\(^{nd}\) order rDG(P\(_0\)P\(_1\))+CG(P\(_1\)) and rDG(P\(_1\)P\(_1\))+CG(P\(_1\)) counterparts: being able to not only increase the accuracy of the velocity by one order but also improve the accuracy of the pressure.

Keywords: Reconstructed Discontinuous Galerkin, Incompressible flows, Projection method

1. Introduction

The discontinuous Galerkin methods\(^4,9,10,13-15,30,31,39\) (DGM) have become popular for the solution of systems of conservation laws in computational fluid dynamics in the past few decades. The discontinuous Galerkin methods combine two advantageous features commonly associated with finite element and finite volume methods. As in classical finite element methods, the DGM achieve high order accuracy by means of high-order polynomial approximation within an element rather than by use of wider stencils as in the case of finite volume methods. The physics of wave propagation is, however, accounted for by solving Riemann problems that arise from the discontinuous representation of the solution at element interfaces, which makes them similar to finite volume methods. The discontinuous Galerkin methods have many attractive features: (1) Their mathematical rigor implies useful mathematical properties with respect to conservation, stability and convergence; (2) The methods can be easily extended to higher-order (> 2\(^{nd}\)) approximations; (3) They are well suited for complex geometries since they can be applied on unstructured grids. In addition, the methods can also handle non-conforming elements, where the grids are allowed to have hanging nodes; (4) The methods are highly parallelizable, as they are compact and each element is independent; (5) 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 implementation and coding; (6) 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

---

\(^1\) Graduate Research Assistant, Department of Mechanical and Aerospace Engineering.
\(^2\) Professor, Department of Mechanical and Aerospace Engineering.
\(*\) Corresponding author: hong_luo@ncsu.edu

© 2016. This manuscript version is made available under the Elsevier user license
http://www.elsevier.com/open-access/userlicense/1.0/
elements; (7) The methods allow easy implementation of hp-refinement, for example, the order of accuracy, or shape, can vary from element to element. p-refinement can be achieved by simply increasing the order of the approximation polynomial. However, the DGM have their own weaknesses. In particular, compared to the finite element methods and finite volume methods, the DGM 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 especially in the context of implicit methods, where the memory requirement for the Jacobian matrix grows quadratically with the order of the DG methods, thus leading to a significant increase in computational cost.

In order to reduce high costs of the DGM, a new family of reconstructed discontinuous Galerkin methods, termed PhPm schemes by Dumbser et al. and referred to as rDG(P_m,P_m) in this paper, have been developed for solving the compressible Euler and Navier-Stokes equations. In the rDG(P_m,P_m) methods, P_n indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and P_m represents a reconstructed polynomial solution of degree of m (m ≥ n) that is used to compute the fluxes. The beauty of rDG(P_m,P_m) schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of rDG(P_m,P_m) schemes, and thus allow for a direct efficiency comparison. When n = 0, i.e. a piecewise constant polynomial is used to represent a numerical solution, rDG(P_0,P_m) is nothing but classical high order finite volume schemes, where a polynomial solution of degree m (m ≥ 1) is reconstructed from a piecewise constant solution. When m=n, the reconstruction reduces to the identity operator, and rDG(P_m,P_m) scheme yields a standard DG method. Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the rDG(P_m,P_m) schemes. In Dumbser’s work, a higher order polynomial solution is reconstructed using a L2 projection, requiring it to be indistinguishable from the underlying DG solutions in the contributing cells in the weak sense. The resultant over-determined system is then solved using a least-squares method that guarantees exact conservation, not only of the cell averages but also of all higher order moments in the reconstructed cell itself, such as slopes and curvatures. However, this conservative least-squares reconstruction approach is computationally expensive, as the L2 projection, i.e., the operation of integration, is required to obtain the resulting over-determined system. Furthermore, the reconstruction might be problematic for a boundary cell, where the number of the face-neighbor cells might not be enough to provide the necessary information to recover a polynomial solution of a desired order. Fortunately, the projection-based reconstruction is not the only way to obtain a polynomial solution of higher order from the underlying discontinuous Galerkin solutions. In a reconstructed DG method using a Taylor basis developed by Luo et al. for the solution of the compressible Euler and Navier-Stokes equations on arbitrary grids, a higher order polynomial solution is reconstructed by use of a strong interpolation, requiring point values and derivatives to be interpolated on the face-neighbor cells. The resulting over-determined linear system of equations is then solved in the least-squares sense. This reconstruction scheme only involves von Neumann neighborhood, and thus is compact, simple, robust, and flexible. Like the projection-based reconstruction, the strong reconstruction scheme guarantees exact conservation, not only of the cell averages but also of their slopes due to a judicious choice of the Taylor basis. A comparative study on these reconstructed discontinuous Galerkin methods rDG(P_2,P_2) to solve the compressible Euler equations on arbitrary grids indicates that the rDG methods can deliver the desired 3rd order of accuracy. They also significantly improve the accuracy of the underlying 2nd order DG method. Thus, it can be concluded that the least-squares reconstruction method provides the best performance in terms of both accuracy and robustness.

A number of the DGM have been developed for solving the incompressible Navier-Stokes equations. Bassi et al. presented a DG method for the incompressible Navier-Stokes equations written in conservative form based on an artificial compressibility formulation. Botti, Di Pietro and Kyriazis, Ekaternaris adopted the pressure-correction formulation to solve the incompressible Navier-Stokes equations written in a non-conservative form. They use a continuous Galerkin CG(P_n) and CG(P_{n+1}) discretizations for the pressure field and DG(P_n) discretization for the velocity field in order to satisfy the Ladyženskaja-Babuška-Brezzi (LBB) condition. They report that the velocity converges with the order of n+1 and the pressure at a convergence rate of n for both DG(P_n)+CG(P_n) and DG(P_n)+CG(P_{n+1}) spatial discretization. In addition, they observe a 2nd order of convergence in time for a range of Re (10^2, 10^3, 10^4).

Based on the success of the rDG methods for the compressible flows, the objective of this work is to develop a rDG method for solving the incompressible Navier-Stokes equations based on a projection formulation. The resulting algorithm is expected to be high-order accurate in space; be able to handle both steady and unsteady problems. Inspite of being high-order accurate, the algorithm must be computationally efficient and able to provide
stable solutions to the incompressible Navier-Stokes equations. A reconstructed discontinuous Galerkin approximation (rDG(P_nP_m)) is used for the velocity field and a continuous Galerkin approximation (CG(P_m)) is used to discretize the pressure field. The developed hybrid method, rDG(P_nP_m)+CG(P_m), inherently satisfies the so-called LBB condition and thus can effectively avoid the pressure checkerboard instability. In order to suppress spurious oscillations in the velocity field for the inviscid or high Reynolds number viscous flows, the nonlinear convective fluxes in the momentum equations are computed using an upwind method. The developed projection method is used to compute a variety of incompressible flow problems on unstructured grids. Only unstructured triangular meshes are used for the numerical tests in this work. The numerical experiments indicate that both rDG(P_0P_1)+CG(P_1) and rDG(P_1P_2)+CG(P_1) methods can attain the designed 2\textsuperscript{nd} and 3\textsuperscript{rd} order of accuracy, respectively and that the 3\textsuperscript{rd} order rDG(P_1P_2)+CG(P_1) method outperforms the 2\textsuperscript{nd} order rDG(P_0P_1)+CG(P_1) method. The remainder of the manuscript is organized as follows. The governing equations are briefly described in Section 2. The newly developed rDG(P_0P_m)+CG(P_m) method for solving the incompressible Navier-Stokes equations is presented and discussed in Section 3. Numerical experiments for a variety of incompressible flow problems on unstructured grids are reported in Section 4. Concluding remarks are given in Section 5.

2. Governing Equations

The incompressible Navier-Stokes equations governing unsteady incompressible viscous flows can be expressed in the conservative form as

\[
\begin{align*}
\frac{\partial \mathbf{v}}{\partial t} + \nabla \cdot (\mathbf{v} \mathbf{v}) - \nu \nabla^2 \mathbf{v} + \frac{1}{\rho} \nabla p &= \mathbf{f} , & \text{in } & \Omega \times (0, T_f) \\
\nabla \cdot \mathbf{v} &= 0 , & \text{in } & \Omega \times (0, T_f)
\end{align*}
\]

(2.1)

(2.2)

for a domain \( \Omega \) with the following initial and boundary conditions,

\[
\begin{align*}
\mathbf{v}(x, 0) &= \mathbf{v}_0 , & \text{in } & \Omega \\
\mathbf{v} &= \mathbf{v}_b , & \text{on } & \Gamma_D \\
\frac{\partial \mathbf{v}}{\partial \mathbf{n}} &= 0 , & \text{on } & \Gamma_N
\end{align*}
\]

(2.3)

where \( \Gamma \) is the boundary of the domain \( \Omega \) and subscripts D and N represent Dirichlet and Neumann boundary conditions respectively. In these equations, \( p \) denotes the pressure of the fluid, and \( \mathbf{v} \) and \( \mathbf{f} \) are the velocity of the flow and the body force vector, respectively. The incompressible Navier-Stokes equations is characterized by the incompressibility (divergence-free) constraint, which couples the velocity and pressure.
3. Numerical Methods

A major difficulty in solving the incompressible Navier-Stokes equations arises from the pressure-velocity coupling by the incompressibility (divergence-free) constraint. Chorin observed in the 1960’s that pressure does not carry any thermodynamic meaning for the incompressible flows, since there is no equation of state for an incompressible fluid. Rather, the pressure plays the role of a Lagrange multiplier, which adjusts its gradient to enforce the incompressibility constraint. This observation led to the development of an operator-splitting scheme, called the projection or fractional-step method\(^7,8,37\). This class of methods is characterized by a two-step predictor-corrector approach, resulting in a decoupling of the velocity and pressure. In the first step, an intermediate velocity is computed from the momentum equations, using an approximation for the pressure gradient and ignoring the incompressibility constraint. In the second step, the Helmholtz-Hodge decomposition is used to project the velocity into a space of divergence-free vector fields, to obtain the pressure and a correction to the intermediate velocity that makes it satisfy the incompressibility constraint. This decoupling makes the projection methods more efficient than the coupled monolithic methods, thus attracting a lot of attention in the past few decades and making them one of the most widely used methods for the solution of the unsteady incompressible Navier-Stokes equations. As a result, many modifications and improvements have been made in the projection methods\(^6,17,20,26,38\). We now briefly discuss the projection methods.

### 3.1 Projection methods

One of the most successful and widely used approach to solve the incompressible Navier-Stokes equations is the class of methods called the projection methods. Originally developed by Chorin\(^7,8\) and Temam\(^37\), these methods decouple the incompressibility constraint and the momentum equation, leading to a step-wise procedure to obtain the velocity and pressure at each time-step. They involve a predictor-corrector procedure for the velocity in which a prediction of a “non divergence-free” velocity is made from the momentum equations using an approximation for the pressure. This decoupling leads to a convection-diffusion equation for the velocity, and an elliptic (Poisson) equation for the pressure (or a related quantity). The predicted velocity is then corrected using the gradient of the solution of this Poisson equation. This way, the final velocity is guaranteed to satisfy the incompressibility condition.

The projection method originally proposed by Chorin\(^8\) and Temam\(^37\) has undergone numerous improvements. We consider the projection method introduced by Temam and further modified by Timmermans\(^38\) in this work, with a few modifications adopted from Botti and Di Pietro’s work\(^5\). Consider the momentum equations. Using an approximate value of pressure (viz. a time-lagged value), say \(q\), and considering a simple Euler backward difference formula for the time derivative, we get,

\[
\frac{v^* - v^n}{\Delta t} + \nabla \cdot (v^* v^*) - \nu \nabla^2 v^* + \frac{1}{\rho} \nabla q = f
\]  

(3.1)

where \(v^*\) is the intermediate non divergence-free velocity. Subtracting this from the original momentum equation using \(v^{n+1}\), we get the equation for the velocity correction,

\[
\frac{v^{n+1} - v^*}{\Delta t} + \frac{1}{\rho} \nabla (p^{n+1} - q) = 0
\]

\[
v^{n+1} = v^* - \frac{\Delta t}{\rho} \nabla (p^{n+1} - q)
\]

(3.2)

Here, note that \(p\) is the physical pressure at the current time-step \(n+1\) and \(q\) is an approximation to it. There are various ways to treat this pressure difference. One of the most common ways is to assume \(q = 0\) in the momentum equation. This gives rise to the non-incremental form of the pressure-correction methods. Another way is to approximate \(q\) by some form of pressure, a time-lagged pressure or an extrapolated value of the pressure\(^6,25\). This leads to the incremental form of the pressure-correction. We use the incremental method in this work. In doing so, we denote \(\Phi = p^{n+1} - q\). Further, we take the divergence of this equation. Noting that \(v^{n+1}\) is the actual physical velocity, and that it satisfies the divergence-free condition, gives us the \(\Phi\)-Poisson equation,
\[ \nabla^2 \Phi = \frac{\rho}{\Delta t} \nabla \cdot \mathbf{v}' \] (3.3)

with the boundary condition,

\[ \frac{\partial \Phi}{\partial n} = 0. \] (3.4)

This boundary condition changes when Neumann boundary condition needs to be enforced on the velocity field (viz. outflow BC). Dirichlet boundaries need to be set for the Poisson equation (3.3) for this case. The Neumann boundary condition (3.4) is enforced on the Poisson equation when any type of Dirichlet boundary condition for the velocity field (viz. inflow BC) needs to be prescribed. It is possible due to the nature of some specific test cases, that only Neumann boundary conditions are imposed on every boundary for the Poisson equation. It is known that the solution to the Poisson equation can be determined only up to a constant in such cases, i.e. a unique solution does not exist. The pressure at a specific node on one of the boundaries is fixed in this case. The choice of the node at which pressure is fixed is arbitrary. This fixes the constant value that cannot be uniquely determined for the Poisson equation, thus permitting a unique solution. It is important to understand that, as mentioned before, the value of the pressure itself plays no role in an incompressible fluid. It is the gradient of pressure that affects the flow. Thus, the constant that is fixed using this approach has no effect on the solution. Also note that when a discretization other than the 1st order Euler method (implicit or explicit) is used for the time derivative, the coefficient of \( \nabla \cdot \mathbf{v}' \) has to be modified accordingly. Since the 1st order Euler method is the only time-integration method used here, such modifications are unnecessary. The details of the time-integration are given in section 3.3.

### 3.2 Spatial discretization

A hybrid approach is used in this work for the spatial discretization of the pressure and velocity, i.e. the discrete pressure and discrete velocity are in different functional spaces. The pressure is in the continuous (or conforming) finite-element space, and is located at the nodal-points of the mesh. The velocity, on the other hand, is in the discontinuous (on non-conforming) finite-element space, and is located at the centroids of the elements of the mesh. See Fig. 1. The standard DG-method and the reconstructed-DG (or rDG) method are considered for discretizing the velocity. In the rDG method, the velocity is reconstructed from a piecewise-linear (P1) space to a piecewise-quadratic (P2) space, to increase its order of accuracy. The reconstruction is done by using the underlying P1 solution. A P2 solution of velocity is derived using the solution information from the neighborhood of each cell with a restriction that the scheme is conservative. Like the velocity, its derivatives determined by the reconstruction are also stored at the element centroids. These spatial discretizations for velocity and pressure are discussed in this section.

![Figure 1: Storage locations in the hybrid discretization](image)
A. Velocity:

The governing equations are discretized using a discontinuous Galerkin finite element method for the velocity field and a standard continuous Galerkin finite element method for the pressure field. This leads to the momentum equation to be in the DG vector space and the Poisson equation for pressure to be in the CG vector space. The momentum equation can be rewritten in a simplified form as,

$$\frac{\partial \mathbf{v}}{\partial t} + \frac{\partial \mathbf{F}_k(\mathbf{v})}{\partial x_k} + \frac{1}{\rho} \nabla p = \mathbf{f},$$  

(3.5)

where density can be taken outside the derivatives since it is constant for incompressible flows. $\mathbf{F}$ is the flux vector given by,

$$\mathbf{F}_j = v_j \mathbf{v} - \nu \frac{\partial \mathbf{v}}{\partial x_j}.$$

The weak formulation for the momentum equation after applying integration by parts on the flux terms is given by,

$$\int_{\Omega} \frac{\partial \mathbf{v}}{\partial t} \mathbf{W} d\Omega + \int_{\Gamma} \mathbf{F}_k(\mathbf{v}) \mathbf{n}_k \mathbf{W} d\Gamma - \int_{\Omega} \frac{\partial \mathbf{W}}{\partial x_k} \mathbf{v}_h d\Omega + \frac{1}{\rho} \int_{\Omega} \nabla p \mathbf{W} d\Omega = \int_{\Omega} \mathbf{f} \mathbf{W} d\Omega$$  

(3.6)

where $\Omega$ denotes the domain and $\Gamma$ is its boundary; and $\mathbf{n}$, the unit outward normal vector to $\Gamma$. The domain $\Omega$ is subdivided into a collection of $Nelem$ non-overlapping triangular elements $\Omega_e$. We introduce the following broken Sobolev space for the velocity field,

$$V_h^p = \left\{ \mathbf{W}_h \in [L_2(\Omega)]^m : \mathbf{W}_h|_{\Omega_e} \in \left[V^m_p \right] \quad \forall \quad \Omega_e \in \Omega \right\}$$  

(3.7)

which consists of discontinuous vector-valued polynomial functions of degree $p$, where $m$ is the dimension of the vector of unknowns $\mathbf{v}$ and,

$$V^m_p = \text{span}\{\prod x_i^{a_i} : 0 \leq a_i \leq p, \ 0 \leq i \leq d\}$$

where $\alpha_i$ denotes a multi-index and $d$ is the dimension of the physical space. Applying the weak formulation on each element $\Omega_e$, we get,

$$\frac{d}{dt} \int_{\Omega_e} \mathbf{v}_h \mathbf{W}_h d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{v}_h) \mathbf{n}_k \mathbf{W}_h d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{v}_h) \frac{\partial \mathbf{W}_h}{\partial x_k} d\Omega + \frac{1}{\rho} \int_{\Omega_e} \nabla p \mathbf{W}_h d\Omega = \int_{\Omega_e} \mathbf{f} \mathbf{W}_h d\Omega \quad \forall \quad \mathbf{W}_h \in V_h^p$$  

(3.8)

where $\mathbf{v}_h$ and $\mathbf{W}_h$ represent the finite element finite element approximations to the analytical solution $\mathbf{v}$ and test function $\mathbf{W}$ respectively. These are approximated by piecewise polynomial functions of degree $p$, which are discontinuous between the cell interfaces. For the Galerkin type of finite element methods, the test function is chosen as the basis $B_i$ of these polynomial functions that approximate the solution $\mathbf{v}_h$. This gives the following system of $N$ equations,

$$\frac{d}{dt} \int_{\Omega_e} \mathbf{v}_h B_i d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{v}_h) \mathbf{n}_k B_i d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{v}_h) \frac{\partial B_i}{\partial x_k} d\Omega + \frac{1}{\rho} \int_{\Omega_e} \nabla p B_i d\Omega = \int_{\Omega_e} \mathbf{f} B_i d\Omega, \quad 1 \leq i \leq N,$$  

(3.9)

where $N$ is the dimension of the polynomial space, which is equal to $(p+1)(p+2)/2$ in 2D. In the traditional DG methods, the numerical polynomial solutions $\mathbf{v}_h$ in each element are expressed using standard Lagrange finite element or hierarchical nodal basis functions, as a result of which, the unknowns solved are the variables located at the nodes. Following Luo et al., we express the numerical solutions in the form of a Taylor series expansion, with the unknown variables located at the cell center as shown in Fig. 2.
If we do a Taylor series expansion at the cell centroid, and take the cell averaged values of the unknowns and their derivatives at the center of the cell, we get,

\[
\begin{align*}
\mathbf{v}_h = \overline{\mathbf{v}} + \frac{\partial \mathbf{v}}{\partial x} |_{c} (x - x_c) + \frac{\partial \mathbf{v}}{\partial y} |_{c} (y - y_c) &+ \frac{\partial^2 \mathbf{v}}{\partial x^2} |_{c} \frac{(x - x_c)^2}{2} - \frac{1}{h_e} \int_{\Omega_e} \frac{(x - x_c)^2}{2} d\Omega \\
&+ \frac{\partial^2 \mathbf{v}}{\partial y^2} |_{c} \frac{(y - y_c)^2}{2} - \frac{1}{h_e} \int_{\Omega_e} \frac{(y - y_c)^2}{2} d\Omega \\
&+ \frac{\partial^2 \mathbf{v}}{\partial x \partial y} |_{c} (x - x_c)(y - y_c) - \frac{1}{h_e} \int_{\Omega_e} (x - x_c)(y - y_c) d\Omega 
\end{align*}
\]  

(3.10)

where \( \overline{\mathbf{v}} \) is the mean value of \( \mathbf{v} \) in the cell and \((x_c, y_c)\) are the coordinates of the cell centroid. The unknowns to be solved in this formulation are the cell-averaged variables and their derivatives at the center of the cells, regardless of the element shape. Using these expansions for the solution variable, the weak form of the system equation becomes,

\[
\left( \int_{\Omega_e} B_i B_j d\Omega \right) \frac{d \mathbf{v}_i}{dt} + \int_{\Omega_e} \mathbf{F}_k(\mathbf{v}_h) n_k B_i d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{v}_h) \frac{\partial B_i}{\partial x_k} d\Omega + \frac{1}{\rho} \int_{\Omega_e} \mathbf{v}_p B_i d\Omega = \int_{\Omega_e} \mathbf{f} B_i d\Omega, \quad 1 \leq i \leq N, 
\]

(3.11)

where \( \int_{\Omega_e} B_i B_j d\Omega \) is the mass-matrix of the system of equations and a repeated index denotes a sum over that index. We consider a three-dimensional polynomial space, resulting in a piecewise linear solution. The corresponding three basis functions are,

\[
B_1 = 1, \quad B_2 = \frac{x - x_c}{\Delta x}, \quad B_3 = \frac{y - y_c}{\Delta y} 
\]

(3.12)

after a normalization suggested by Luo et al.\textsuperscript{29} The linear solution is then written as,

\[
\mathbf{v}_h = \overline{\mathbf{v}} + \frac{\partial \mathbf{v}}{\partial x} x_c \Delta x B_2 + \frac{\partial \mathbf{v}}{\partial y} y_c \Delta y B_3 
\]

(3.13)

The only term connecting one element to the other is the interface flux term. The other terms are all uncoupled from the neighboring cells. This makes the DG system very compact, since the support extends only to the immediate neighborhood for each element. This is an important advantage of DG methods.

Further, we reconstruct the P1 polynomial solution to the P2 space to improve the accuracy following Luo et al.\textsuperscript{31} This is termed as the reconstructed-DG or rDG method. The underlying DG solution is reconstructed to a quadratic polynomial solution,
\[
V^R_h = \hat{V} + \frac{\partial V^R}{\partial x} \Delta x B_2 + \frac{\partial V^R}{\partial y} \Delta y B_3 + \frac{\partial^2 V^R}{\partial x^2} \Delta x^2 B_4 + \frac{\partial^2 V^R}{\partial y^2} \Delta y^2 B_5 + \frac{\partial^2 V^R}{\partial x \partial y} \Delta x \Delta y B_6.
\] (3.14)

The higher-order basis functions are defined as,

\[
B_4 = \frac{(x - x_c)^2}{2 \Delta x^2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(x - x_c)^2}{2 \Delta x^2} d\Omega,
\]
\[
B_5 = \frac{(y - y_c)^2}{2 \Delta y^2} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(y - y_c)^2}{2 \Delta y^2} d\Omega,
\]
\[
B_6 = \frac{(x - x_c)(y - y_c)}{\Delta x \Delta y} - \frac{1}{\Omega_e} \int_{\Omega_e} \frac{(x - x_c)(y - y_c)}{\Delta x \Delta y} d\Omega.
\] (3.15)

In order to maintain the compactness of the DG methods, the reconstruction is required to only involve information from the adjacent cells that share a face with the cell over which solution is being reconstructed. There are six degrees of freedom giving 6 unknowns. The first three can be determined trivially by requiring the reconstruction to be conservative, thus giving the first 3 DOF’s equal to the 3 DOF’s of the underlying DG solution at the cell centroid. The remaining DOF’s can be computed in a least-squares sense or using Green-Gauss theorem. Here, the least-squares method described by Luo et al.\(^3\) is used. Due to the incompressible nature of the flow, strong discontinuities are not encountered. Thus, there is no need of limiting the high-order solution.

The advective fluxes at the interface in Eq. (3.11) are computed by an Upwind method. The upwind direction is determined based on the direction of the normal velocity to the interface. The velocity in the upwind direction is then used to compute the flux. The discrete numerical flux is given as follows:

\[
\mathbf{F}_{ij} \cdot \mathbf{n} = \begin{cases} 
F^+_i \cdot \mathbf{n}, & \text{if } \mathbf{v}_j^+ \cdot \mathbf{n} > 0 \text{ and } \mathbf{v}_j^- \cdot \mathbf{n} > 0 \\
F^-_j \cdot \mathbf{n}, & \text{if } \mathbf{v}_j^+ \cdot \mathbf{n} < 0 \text{ and } \mathbf{v}_j^- \cdot \mathbf{n} < 0 \\
\frac{1}{2} (F^+_i + F^-_j) \cdot \mathbf{n}, & \text{if } \mathbf{v}_j^+ \cdot \mathbf{n} \geq 0 \text{ and } \mathbf{v}_j^- \cdot \mathbf{n} \leq 0
\end{cases}
\] (3.16)

Since the pressure is uniquely determined on the interfaces, there is no need to consider an element-interface \((I_e)\) and element-domain \((\Omega_e)\) integral separately; only integrating over the element as shown in Eq. (3.11) suffices. The diffusive fluxes for the rDG\((P_1P_1)\) spatial discretization are computed using the following formula from Mathur and Murthy\(^4\):

\[
\nabla v_{ij} = \frac{v_j - v_i}{\Delta r_{ij} \cdot \mathbf{n}} + \left( \nabla \frac{\overline{v_{ij}}}{\Delta r_{ij} \cdot \mathbf{n}} \cdot \Delta r_{ij} \cdot \mathbf{n} \right)
\] (3.17)

where \(\Delta r_{ij}\) is the displacement vector between the cell centroids \(i\) and \(j\). The first term in the equation is the component normal to the face \(ij\) and the second term is the tangential component computed by removing the normal component from the average gradient at the face \(\nabla \overline{v_{ij}}\). The BR2 method, developed by Bassi and Rebay\(^5\), is used for computing the viscous fluxes in the rDG\((P_1P_1)\) and rDG\((P_1P_2)\) methods.

B. Pressure:

The pressure is discretized using the standard continuous Galerkin finite element method. Introducing the following continuous Sobolev space for the pressure field,

\[
Q^p_h = \{ W_h \in [H^1_0(\Omega)]^m : W_h|_{\Omega_e} \in [Q^m_p] \quad \forall \quad \Omega_e \in \Omega \}
\] (3.18)
where the P1 or P2 polynomial space is used for the pressure field, the finite element formulation for the $\Phi$-Poisson equation (3.3) can be expressed as

$$
\sum_{i=1}^{N_{\text{poin}}} \left( \int_{\Omega} \nabla D_i \cdot \nabla D_j d\Omega \right) \phi_i = -\rho \int_{\Omega} \frac{\nabla \cdot \mathbf{v}^*}{\partial t} D_j d\Omega, \quad \forall \quad D_j \in Q_h^p \tag{3.19}
$$

where $D_j$ are the standard Lagrange basis functions and $N_{\text{poin}}$ is the number of grid-points, where the pressure and $\phi$ are stored.

Since the predicted velocity $\mathbf{v}^*$ is discontinuous at element interfaces, the domain integral has to take the interfacial “jumps” of the velocity into consideration. For this, we use integration by parts on the velocity-divergence term.

$$
\sum_{i=1}^{N_{\text{poin}}} \left( \int_{\Omega} \nabla D_i \cdot \nabla D_j d\Omega \right) \phi_i = -\rho \int_{\Omega} \frac{\nabla \cdot \mathbf{v}^*}{\partial t} \left( \sum_{D_e} \int_{r_e} \mathbf{v}^* \cdot \mathbf{n} D_j d\Gamma - \int_{\Omega_e} \nabla D_j \cdot \mathbf{v}^* d\Omega \right) \tag{3.20}
$$

where $N_i$ is the set of faces surrounding the point $i$. The values of $\mathbf{v}^*$ at the interface are not unique for DG methods, since two values $\mathbf{v}_{r_i}$ and $\mathbf{v}_{r_i}$ can be obtained from the elements on each side of the interface. Using the average of these values, and writing it as $\mathbf{v}_{r_{i}} = \frac{1}{2} (\mathbf{v}_{r_i} + \mathbf{v}_{r_i})$ and defining a jump operator $[\mathbf{v}^*] \equiv \mathbf{v}_{r_{i}} - \mathbf{v}_{r_{i}}$, we get,

$$
\sum_{i=1}^{N_{\text{poin}}} \left( \int_{\Omega} \nabla D_i \cdot \nabla D_j d\Omega \right) \phi_i = -\rho \int_{\Omega} \frac{\nabla \cdot \mathbf{v}^*}{\partial t} \left( \sum_{D_e \in N_i} \int_{r_e} \left( \frac{1}{2} [\mathbf{v}^*] \right) \cdot \mathbf{n} D_j d\Gamma - \int_{\Omega_e} \nabla D_j \cdot \mathbf{v}^* d\Omega \right). \tag{3.21}
$$

Returning $\mathbf{v}_{r_{i}}$ from the interface integral and the whole domain integral on the right-hand side to its original form using integration-by-parts in reverse, we get the final weak form of the Poisson equation for $\Phi$,

$$
\sum_{i=1}^{N_{\text{poin}}} \left( \int_{\Omega} \nabla D_i \cdot \nabla D_j d\Omega \right) \phi_i = -\rho \int_{\Omega} \frac{1}{2} \sum_{D_e \in N_i} \int_{r_e} [\mathbf{v}^*] \cdot \mathbf{n} D_j d\Gamma - \int_{\Omega} \nabla \cdot \mathbf{v}^* D_j d\Omega. \tag{3.22}
$$
3.3 Temporal discretization

The time-derivative in the momentum equation has been discretized using two temporal discretization schemes, the explicit Euler and the implicit Euler difference formulae, in this work. Depending on which one is used, the algorithm becomes an explicit or implicit one. Both these methods have been used in this work.

A. Explicit method:

The incremental projection method, using the forward Euler method for the temporal discretization is given by,

\[
\frac{\mathbf{v}^* - \mathbf{v}^n}{\Delta t} + \nabla \cdot (\mathbf{v}^n \mathbf{v}^n) - \nu \nabla^2 \mathbf{v}^n + \frac{1}{\rho} \nabla p^n = \mathbf{f}
\]

\[
\nabla^2 \phi = \frac{\rho}{\Delta t} \nabla \cdot \mathbf{v}^*, \quad \frac{\partial \phi}{\partial n}|_r = 0
\]

\[
p^{n+1} = p^n + \phi
\]

\[
\mathbf{v}^{n+1} = \mathbf{v}^* - \frac{\Delta t}{\rho} \nabla \phi
\]

The Poisson equation for \( \Phi \) leads to a symmetric positive-definite system matrix, which therefore solved by the Conjugate Gradient (CG) method, preconditioned by the Lower-Upper Symmetric Gauss Seidel (LU-SGS) method. Since all the fluxes are calculated using the velocities at the previous time-step \( \mathbf{v}^n \), they get absorbed into the right-hand side and a single explicit equation is obtained. Although this formulation is easy to implement, and obviates the need to deal with nonlinear implicit terms and iterative solvers; the time-step sizes which can be used by an explicit algorithm are extremely low. For time-independent flows, the steady-state is achieved after running the algorithm to a very long \( t_f \). Since this work focuses on the spatial accuracy of the rDG method, time-steps chosen are small enough so that errors from the temporal discretization do not dominate. Typically, this means that the \( \Delta t \)'s used are of the order of 0.001 (CFL ~ 0.05). Thus explicit algorithms are more suitable for unsteady flows. However, given this restriction on \( \Delta t \), the explicit algorithms are prohibitive for steady-state flows. Thus we resort to an implicit treatment of the velocity in the momentum equations, described in the following section.

B. Implicit method:

The incremental projection method, using the backward Euler method for the temporal discretization is given by,

\[
\frac{\mathbf{v}^* - \mathbf{v}^n}{\Delta t} + \nabla \cdot (\mathbf{v}^* \mathbf{v}^*) - \nu \nabla^2 \mathbf{v}^* = -\frac{1}{\rho} \nabla p^n + \mathbf{f}
\]

with the Poisson equation, pressure and velocity correction and boundary conditions same as the explicit method. Note that for the implicit method all the fluxes are computed using the velocities at the current time-step, \( \mathbf{v}^* \). Also note that the \( x \) and \( y \) momentum equations are coupled via the convection speed. The system is now a nonlinear and coupled implicit system. The methodology from Xia et al. is used for the linearization. First, we write the implicit system as,

\[
M \cdot \mathbf{v}^* - \mathbf{v}^n = R(\mathbf{v}^*),
\]

where,

\[
R(\mathbf{v}^*) = -\nabla \cdot (\mathbf{v}^* \mathbf{v}^*) + \nu \nabla^2 \mathbf{v}^* - \frac{1}{\rho} \nabla p^n + \mathbf{f}.
\]

Now, the vector \( R(\mathbf{v}^*) \) is linearized with respect to the solution vector \( \mathbf{v}^n \) using a Taylor series expansion,

\[
R(\mathbf{v}^*) = R(\mathbf{v}^n) + \left( \frac{\partial R}{\partial \mathbf{v}} \right)^n (\mathbf{v}^* - \mathbf{v}^n)
\]
where \( \left( \frac{\partial R}{\partial \nu} \right)^n \) is the flux-Jacobian. The flux-Jacobian is computed by taking the analytical derivatives of the flux function. Using this, and denoting \( \Delta \nu^n = \nu^* - \nu^n \), the linearized form of the implicit momentum equation is obtained

\[
A \cdot \Delta \nu^n = R(\nu^n),
\]

(3.27)

where \( A \) is the system matrix

\[
A = \frac{M}{\Delta t} - \left( \frac{\partial R}{\partial \nu} \right)^n,
\]

and \( M \) is the mass matrix from the discretized momentum equation (3.11). The linearized implicit momentum equations are solved by the Generalized Minimum Residuals method (GMRES) preconditioned by a block-Jacobi method\(^{28}\). The GMRES takes about 5 to 7 iterations for the residuals to drop by four orders of magnitude. This convergence criterion is checked to have no effect on the solution accuracy when the spatial order-of-convergence is determined. The pressure Poisson equation is solved using the Conjugate Gradient (CG) method using a Lower-Upper Symmetric Gauss Seidel (LU-SGS) preconditioner, i.e. using the same method as in the explicit time-stepping.

It is important to note that since only first-order accurate time discretizations have been used in this work, the timesteps used are small enough so that truncation errors from the time-discretization do not dominate the final solution errors. This ensures that the spatial order is judged correctly. Tests have been conducted to make sure that this is indeed the case. This is of importance for the transient problems, since the solution is time-dependent. Since small time-steps required for transient problems, the explicit method is preferred. On the other hand, for steady problems, largest possible time-steps have to be chosen to drive the transient solver to steady state. Thus the implicit method is preferred for steady problems.
4. Numerical Results

The developed rDG(P_1+P_1)+CG(P_1) method was used to compute a number of classical incompressible flow problems for a variety of flow conditions. The numerical results obtained were compared with the lower order rDG(P_0+P_1)+CG(P_1) and rDG(P_1+P_1)+CG(P_1) methods. An explicit time discretization was used for unsteady flow problems, whereas the implicit BDF1 scheme was used for steady state problems. These results are presented in this section. The errors with respect to the exact solution are measured in the $L^2$-norm, defined as,

$$
\|e_u\|_{L^2(\Omega)} = \sqrt{\int_{\Omega} (u_h - u_e)^2 \, d\Omega} = \sqrt{\sum_{i=1}^{\text{Nets}} \int_{\Omega_e} (u_h - u_e)^2 \, d\Omega}
$$

where the integrals are computed using Gaussian quadrature of sufficient order. The errors thus obtained are used to compute the spatial order of accuracy of the method.

A. Kovasznay flow

The Kovasznay flow is an exact solution to the stationary incompressible Navier-Stokes equations. It represents flow past a grid or an array of cylinders at low Reynolds numbers. This flow is considered in the two-dimensional domain (-0.5,1.5) x (0,2) with Dirichlet boundary conditions obtained from the exact solution and initial conditions corresponding to a uniform flow field. The velocity and pressure fields are given by,

$$
V_x = 1 - e^{4x} \cos(2\pi y)
$$
$$
V_y = \frac{\lambda}{2\pi} e^{4x} \sin(2\pi y)
$$
$$
p = \frac{1}{2} (1 - e^{2\lambda x})
$$

where $\lambda$ is defined by,

$$
\lambda = \frac{1}{2\nu} - \frac{1}{\sqrt{4\nu^2 + 4\pi^2}}
$$

We use $\nu = 0.025$ ($Re = 40$). Due to its steady nature, temporal-discretization errors do not affect the order of accuracy. Thus, it is a good case to assess the spatial order-of-convergence. We use unstructured triangular meshes with 554, 2216, 8864 elements (shown in Fig. 3) for these tests. Contours of the velocity magnitude and the streamlines are shown in Fig. 4.

![Figure 3: The meshes used for judging spatial order of accuracy. Coarse mesh with 554 (left), fine mesh with 2216 (center) and the finest mesh with 8864 (right) elements.](image)
Errors plotted against the element size on a logarithmic scale is shown in Fig. 5. The rDG(P₀P₁) method is clearly more accurate than the rDG(P₁P₁) method, but the real advantage of the DG method shows after its reconstruction to P₂-polynomial. The reconstruction clearly improves the order of accuracy of the 2nd order DG method to make it 3rd order accurate. It also significantly improves the accuracy of the method by reducing the errors by more than an order of magnitude. Fig. 6 shows the errors plotted against the total number of DOF’s in the mesh. Here it is observed that the rDG(P₁P₂) method is more effective than both the other methods, in that, the least number of DOF’s are required to attain a certain level of accuracy by the rDG(P₁P₂) method as compared to the rDG(P₀P₁) and rDG(P₁P₁) methods, illustrating the superiority of the rDG(P₁P₂) method.

Figure 5: Spatial convergence of velocity (left) and pressure (right) for Kovasznay flow test case
Figure 6: Errors in velocity for Kovasznay flow plotted against the total number of DOF's
### B. Taylor vortex

The Taylor vortex flow is an exact solution to the unsteady incompressible Navier-Stokes equations. Similar to the previous test case, the Dirichlet boundary conditions are obtained from the exact solution,

\[
\begin{align*}
V_x &= \frac{1}{10} \pi \cos(0.1 \pi x) \sin(0.1 \pi y) \ e^{-0.02 \pi^2 \nu t} \\
V_y &= -\frac{1}{10} \pi \sin(0.1 \pi x) \cos(0.1 \pi y) \ e^{-0.02 \pi^2 \nu t} \\
p &= -\frac{1}{400} \pi^2 (e^{-0.02 \pi^2 \nu t})^2 (\cos(0.2 \pi x) + \cos(0.2 \pi y) - 2) .
\end{align*}
\]  \hspace{1cm} (4.3)

Unlike the previous test case, however, since this is an unsteady flow, the initial conditions are obtained from the exact solution as well. This flow is considered on the two-dimensional domain \((0,10) \times (0,10)\). We use \(Re = 10\) and \(Re = 100\) to ensure that the spatial convergence is independent of the viscosity. A time-step of \(10^{-5}\) has been used to drive the simulation to a final time of \(1.0\). The same set of meshes used for the Kovasznay flow test were used for this test case. From the convergence results of velocity in Figs. 7, 8 and 9 the rDG(P₁,P₂) method is seen to outperform the other methods again.

**Figure 7:** Spatial convergence of velocity (left) and pressure (right) for Taylor vortex with \(Re = 100\) at \(t = 1.0\)

**Figure 8:** Spatial convergence of velocity (left) and pressure (right) for Taylor vortex with \(Re = 10\) at \(t = 1.0\)
Errors at different flow times have been considered to make sure that the reported order-of-accuracy does not deteriorate with time. This has been confirmed and for the sake of brevity, these results for different flow times have not been included here.

Figure 9: Errors in velocity plotted against the total number of DOF’s for Taylor vortex with $Re=100$
C. Forced flow problem

We consider a manufactured solution of the unsteady incompressible Navier-Stokes equations from Liu et al.\textsuperscript{26} as another test case. This is a forced flow problem with the following exact solution,

\[ V_x = A \pi \sin(t) \sin^2(\pi x) \sin(2\pi y) \]

\[ V_y = -A \pi \sin(t) \sin(2\pi x) \sin^2(\pi y) \]

\[ p = A \sin(t) \cos(\pi x) \sin(\pi y) \]

(4.4)

and the forcing terms can be computed from the momentum equations. The initial and boundary conditions can be obtained from the exact solution. We use a two-dimensional domain \((0,1) \times (0,1)\) for this test case. First, an amplitude of \(A = 1.0\) is considered, using \(Re = 10\) and 100 (\(v = Re^{-1} \); 0.1 and 0.01). A time-step of \(10^{-5}\) is used to drive the solution to a final time of 0.1. From Fig. 10, it can be seen that the rDG(P\(_1\)P\(_2\)) method is 3\textsuperscript{rd} order accurate, whereas the rDG(P\(_1\)P\(_1\)) method is only 2\textsuperscript{nd} order accurate, again outperforming the rDG(P\(_1\)P\(_1\)) method.

![Figure 10: Spatial convergence for velocity (left) and pressure (right) for forced-flow flow test case at t = 0.1](image-url)
Figure 11: Spatial convergence for velocity (left) and pressure (right) for forced-flow flow test case at \( t = 0.1 \) at \( A = 20 \)

Results for an amplitude of \( A = 20.0 \) (\( Re = 0.1 \)) are shown in Fig. 11. The order of accuracy for velocity remains unaffected; however the order of accuracy of pressure is slightly affected. Note that the error in pressure using \( rDG(P_1P_2) \) show a very steep drop after the first mesh-refinement. Due to this, the order of accuracy obtained only from the second to the third mesh has been mentioned in the legend. Thus the asterisk.
D. Flow over a flat plate

The laminar boundary layer flow across a flat plate is considered in this test case, over a domain bounded by $(0.5, 1) \times (0, 1)$, and the flat-plate starts at point $(0,0)$ and extends to $(1,0)$. This test is carried out at $Re = 100,000$, with a mesh suitably generated to capture the boundary-layer. The height of the first element is $0.3464E^{-03}$ and $0.8265E^{-03}$ at the leading and trailing edge of the plate respectively. The mesh is shown in Fig. 12. Numerical results obtained from the $rDG(P_0P_1)$, $rDG(P_1P_1)$ and $rDG(P_1P_2)$ are compared with the theoretical Blasius solution.

![Figure 12: Mesh used for the flat plate boundary layer problem](image)

The skin-friction coefficient (Fig. 13) and velocity profiles (Figs. 15, 16 and 17) across the boundary layer at $x=0.5$, $x=0.8$ and $x=1$ along the flat plate are compared with Blasius’s exact solution. Fig. 14 shows a blown up image of the coefficients of friction computed by the 3 methods. It is clear from these images that the higher-order $rDG(P_1P_2)$ method predicts the coefficient of friction more accurately that the lower-order methods after the boundary layer growth. At the initial stages of the plate, the $rDG(P_1P_1)$ seems to be less accurate than the $rDG(P_0P_1)$ method. Yet, in this region as well, the $rDG(P_1P_2)$ is more accurate than the other two. The accuracy of the higher-order reconstruction is evident in the resolution of the $y$-component of velocities, which are extremely difficult to compute accurately (Figs. 15, 16 and 17). Again, we observe that the $rDG(P_1P_1)$ method has inaccuracies, in that, the velocity profile seems to have some oscillations. This is rectified by the reconstruction in the $rDG(P_1P_2)$ method, which makes it more accurate than the lower order methods.
Figure 13: Coefficient of friction computed by $rDG(P_0P_1)$, $rDG(P_1P_1)$ and $rDG(P_1P_2)$ compared to Blasius's solution.

Figure 14: Coefficient of friction computed by $rDG(P_0P_1)$, $rDG(P_1P_1)$ and $rDG(P_1P_2)$ on a section of the plate $x = 0.01$ to $1$. 
Figure 15: X-velocity (left) and Y-velocity (right) at x=0.5

Figure 16: X-velocity (left) and Y-velocity (right) at x=0.8

Figure 17: X-velocity (left) and Y-velocity (right) at x=1.0
E. Driven cavity

The 2-dimensional driven cavity is a benchmark test case used to validate CFD codes. The computational domain is a square with edges of unit length. The top surface (edge in 2D) is imposed with a constant horizontal speed. This drives the fluid inside the square cavity into a vortical flow, resulting in complex flow fields consisting of a large central vortex accompanied by small recirculating zones in the corners. The number of these tiny corner vortices varies with Re. This test case therefore provides a means to test the rDG(P1,P2)+CG(P1) method for a range of Re's.

![Streamlines for 2D driven cavity flow at Re=400 (left), Re=1000 (center) and Re=3200 (right)](image)

Figure 18: Streamlines for 2D driven cavity flow at Re=400 (left), Re=1000 (center) and Re=3200 (right)

The finest mesh shown in Fig. 3 was used for these tests. An implicit time-integration is used to march the solution in time until a steady-state is reached. The streamlines for Re = 400, 1,000, 3,200 and Re = 10,000 are shown in Fig. 18. The velocity profiles are compared with the data published by Ghia et al.\textsuperscript{16} to ensure that the obtained results are accurate. Ghia et al. obtained these results using the vorticity-stream function formulation of the incompressible N-S equation using an implicit multigrid method. The x-component of velocity along the vertical centerline and the y-component along the horizontal centerline are both shown in Figs. 19, 20 and 21. These results demonstrate the high accuracy attainable for flows at high Reynolds numbers by the rDG(P1,P2)+CG(P1) discretization.

![X-component of velocity along the vertical centerline (left); and Y-component of velocity along the horizontal centerline (right) compared with results from Ghia et al.\textsuperscript{16} Re=400.](image)

Figure 19: X-component of velocity along the vertical centerline (left); and Y-component of velocity along the horizontal centerline (right) compared with results from Ghia et al.\textsuperscript{16} Re=400.
Figure 20: X-component of velocity along the vertical centerline (left); and Y-component of velocity along the horizontal centerline (right) compared with results from Ghia et al.\textsuperscript{16} Re=1000.

Figure 21: X-component of velocity along the vertical centerline (left); and Y-component of velocity along the horizontal centerline (right) compared with results from Ghia et al.\textsuperscript{16} Re=3200.

F. Kármán vortex street

The flow past circular cylinder is one of the most extensively studied cases, both experimentally and numerically. Thus, it is chosen as a standard validation test. The flow conditions chosen here correspond to those in Luo et al.\textsuperscript{42}; freestream Reynolds number $Re_{\infty}= 200$ based on the cylinder diameter of $d = 1.0$ with an angle of attack $\alpha = 0^\circ$. This test case is geometrically simple, yet aerodynamically complex due to vortices being shed periodically downstream of the cylinder. The mesh used for this test case, shown in Fig. 22, has 8,192 triangular elements, 4,224 grid points and 256 boundary faces. The potential flow solution around the cylinder is used as the initial condition. The explicit Euler time discretization is used for the computations, with a time-step of $\Delta t = 0.0001$ to resolve the unsteady nature of the flow accurately. The contours of velocity magnitude and streamlines for the resulting flow are shown in Fig. 23. The lift and drag coefficients obtained by the rDG(P$_1$P$_2$)+CG(P$_3$) method are shown in Fig. 24 and 25. Note that, the initiation of the vortex shedding in a numerical simulation will depend on the initial solution used. The quantity being compared is, therefore, the frequency of oscillations in the data and its amplitude; not the start of the vortex shedding. The Strouhal number and lift and drag coefficients obtained are listed in Table 1. As shown, these compare well with the results obtained by Luo et al.\textsuperscript{42}, Saiki and Biringen\textsuperscript{43} and Franke et al.\textsuperscript{44}
Figure 22: Triangle mesh used for computing viscous flow around a cylinder

Figure 23: Contours of velocity magnitude and streamlines for viscous flow around a cylinder at $Re = 200$
Table 1: Comparison of aerodynamic coefficients and Strouhal number with references 43 and 44

<table>
<thead>
<tr>
<th>Lift Coefficient</th>
<th>Strouhal number</th>
</tr>
</thead>
<tbody>
<tr>
<td>rDG(P1P2) + CG(P2)</td>
<td>± 0.687</td>
</tr>
<tr>
<td>+ CG(P2)</td>
<td>± 0.65</td>
</tr>
<tr>
<td>Luo et al.</td>
<td>0.19608</td>
</tr>
<tr>
<td>Sui and Biringen</td>
<td>0.187</td>
</tr>
<tr>
<td>Franke et al.</td>
<td>0.194</td>
</tr>
</tbody>
</table>

Figure 24: Time-history of the coefficient of lift for flow past a cylinder at Re = 200

Figure 25: Time-history of the coefficient of drag for flow past a cylinder at Re = 200
5. Conclusions

A hybrid reconstructed discontinuous Galerkin and continuous Galerkin method: rDG(Pm,Pn)+CG(Ps), based on an incremental pressure projection formulation has been developed for the numerical solution of the unsteady incompressible Navier-Stokes equations on unstructured grids. A number of benchmark test cases for a variety of flow conditions have been presented to assess the order of spatial convergence, accuracy, and performance of the rDG(Pm,Pn)+CG(Ps) method. The numerical experiments demonstrate that both rDG(Pm,Pn)+CG(Ps) and rDG(Pm,Pn)+CG(Ps) methods can attain the designed 2nd order and 3rd order accuracy in space for the velocity and pressure respectively. The 3rd order rDG(Pm,Pn)+CG(Ps) method significantly outperforms its 2nd order rDG(Pm,Pn)+CG(Ps) and rDG(Pm,Pn)+CG(Ps) counterparts: being able to not only increase the accuracy of the velocity by one order but also improve the accuracy of the pressure. The current development is focused on implementation of the rDG(Pm,Pn)+CG(Ps) method on heterogeneous CPU and GPU architecture, and the extension of the rDG(Pm,Pn)+CG(Ps) method for 3D and multi-phase flow problems is also under consideration.

Acknowledgments

This research was partially supported by the Consortium for Advanced Simulation of Light Water Reactors (http://www.casl.gov), an Energy Innovation Hub (http://www.energy.gov/hubs) for Modeling and Simulation of Nuclear Reactors under U.S. Department of Energy Contract No. DE-AC05-000R22725. The authors would also like to acknowledge the partial support for this work provided by the Basic Research Initiative program of the Air Force Office of Scientific Research. Dr. F. Fariba and Dr. D. Smith serve as the technical monitors. The authors are also grateful towards Dr. Lijun Xuan for helpful discussions.

References

33 Hong Luo, Yidong Xia, Robert Nourgaliev, and Chunpei Cai, A class of reconstructed discontinuous Galerkin methods for the compressible flows on arbitrary grids. AIAA-0199, 2011.