

A PARALLEL HIGH-ORDER DISCONTINUOUS GALERKIN SOLVER FOR
THE UNSTEADY INCOMPRESSIBLE NAVIER-STOKES EQUATIONS IN
COMPLEX GEOMETRIES

by

Khosro Shahbazi

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Mechanical and Industrial Engineering
University of Toronto

Copyright © 2007 by Khosro Shahbazi

Abstract

A Parallel High-Order Discontinuous Galerkin Solver For the Unsteady Incompressible
Navier-Stokes Equations in Complex Geometries

Khosro Shahbazi

Doctor of Philosophy

Graduate Department of Mechanical and Industrial Engineering

University of Toronto

2007

We develop a parallel method and corresponding code for the numerical solution of the unsteady incompressible Navier-Stokes equations, with application to the direct numerical simulation of transitional and turbulent flows through mechanical heart valves. The solver is based on a simple and efficient scheme, namely a high-order discontinuous Galerkin method on triangular and tetrahedral elements. Spatial discretization of the Stokes operator employed both equal-order ($P_k - P_k$) and mixed-order ($P_k - P_{k-1}$) velocity and pressure approximations. The interior penalty method and local Lax-Friedrichs fluxes are used for the discretizations of the viscous term and the nonlinear term in the divergence form, respectively. A second order approximate algebraic splitting is used to decouple the velocity and pressure calculations leading to an algebraic Helmholtz equation for each component of the velocity and a consistent Poisson equation for the pressure. The consistent Poisson operator is replaced by an equivalent operator, namely that arising from the interior penalty discretization of the standard Poisson operator with appropriate boundary conditions. An explicit lower bound is derived for the penalty parameter of the interior penalty method that ensures the coercivity of the bilinear form. Efficiency aspects of the scheme include knowing an explicit expression for the penalty parameter of the interior penalty method and compact stencil size for the discretizations of the velocity and pressure equations and the nonlinear term.

We verify the accuracy and stability of the method on several two- and three-dimensional benchmarking problems. On the challenging Orr-Sommerfeld test problem, the equal-order polynomial approximation of the velocity and pressure ($P_k - P_k$) leads to a stable and accurate solution, while the mixed-order method ($P_k - P_{k-1}$) yields a non-physical instability. In simulating vortex shedding past a square cylinder at $Re = 100$ and in simulating a three-dimensional backward-facing step flow using the equal-order method, excellent agreement with other computational and experimental results are obtained. The developed solver is used to study flow through a two-dimensional bileaflet mechanical heart valve geometry.

We conclude that the proposed discontinuous Galerkin method with the $P_k - P_k$ formulation is a suitable scheme for simulations of flows through mechanical heart valve geometries.

Acknowledgements

I am grateful to my advisor, Prof. Ross Ethier, for all his help and support during the course of this work. He has been a great mentor and always available to answer my questions even when he is very busy or out of town. His discipline and thoroughness have been very inspiring.

I gratefully acknowledge Dr. Paul Fischer for his constructive advice and suggestions on this work. He has been very helpful answering my questions with patience and understanding.

I acknowledge my supervising and examining committee, Profs. Christina Christara, Pierre Sullivan, Markus Bussmann and David Steinman.

I thank PETSc developers for their valuable responses to my questions regarding the use and features of the library.

I acknowledge many past and present students in the Computational Fluid Dynamics and Biomechanics laboratories at the University of Toronto, including Blake Charlebois, Cezary Niewiadomski, Lev Vaisman, Ji Zhang and Tao Xu.

I greatly appreciate my parents, sisters, brother and girl friend for their love and support throughout years.

Contents

1	Introduction	1
1.1	Motivation	1
1.1.1	Mechanical Heart Valves	1
1.1.2	Design and Clinical Impact	4
1.1.3	Synergy of Simulation and Experiment for Design	6
1.2	Mechanical Heart Valve Simulations	7
1.2.1	Previous Simulations	7
1.2.2	Complexities of Simulation	9
1.2.3	Direct Numerical Simulation	9
1.3	Methodology of Direct Numerical Simulation	14
1.3.1	Dispersion Error and High-Order Approximation	15
1.3.2	Choice of Spatial Discretization	16
1.4	Objectives	18
2	Overview of Discontinuous Galerkin Methods	20
2.1	DG Method for a Hyperbolic Equation	20
2.1.1	DG Method for an Elliptic Problem	22
2.2	Simplicity	25
2.2.1	Parallelization	25
2.2.2	Non-conforming hp-adaptivity	28

2.3	Weak Imposition of Dirichlet Boundary Conditions	30
2.4	Conservativity	30
2.5	Efficiency	31
2.5.1	Total Number of Unknowns	31
2.5.2	Block Diagonal Mass Matrix	33
2.6	Drawbacks of DG methods	35
3	A High-Order DG Method for the NS Equations	37
3.1	Introduction	37
3.1.1	Review of DG Discretization of the Convective Operator	38
3.1.2	Review of DG Discretization of the Stokes Operator	39
3.2	Navier-Stokes Discretization	41
3.2.1	Preliminaries	42
3.2.2	Nonlinear Treatment	43
3.2.3	Stokes Discretization	44
3.2.4	DG Formulation of the Unsteady Stokes Operator	48
3.3	Implementation	52
3.3.1	Differentiation	53
3.3.2	Inner-Product Evaluations	53
3.3.3	Quadrature for the Nonlinear Term	55
3.4	Verification on Two-dimensional Problems	55
3.4.1	Temporal Error Test	56
3.4.2	Taylor Vortex Problem	57
3.4.3	Orr-Sommerfeld Stability Problem	61
3.4.4	Flow Past a Square Cylinder	64
3.5	Flow through a Two-dimensional Mechanical Heart Valve	65

4	Verification on Three-dimensional Problems	72
4.1	Three-dimensional Taylor Vortex	72
4.2	Three-dimensional Orr-Sommerfeld Stability	76
4.3	Backward-facing step flow	80
4.3.1	$Re = 172$	84
4.3.2	$Re = 344$	85
4.4	Summary	90
5	An Explicit Expression for the Penalty Parameter	95
5.1	Introduction	95
5.2	Interior Penalty Method	96
5.3	Explicit Expression for the Penalty Parameter	100
6	Implementation and Performance Study	105
6.1	Linear System Solves	105
6.1.1	Projection Technique for the Pressure System	107
6.2	Some Coding Details	108
6.3	Performance Study	110
6.3.1	Performance Enhancement due to the Projection Method	110
6.3.2	Parallel Performance	113
7	Conclusions and Future Directions	120
7.1	Summary and Conclusions	120
7.1.1	First Objective	120
7.1.2	Second Objective	121
7.2	Contributions	123
7.3	Future Directions	123
7.3.1	Strategies for Efficiency Improvement	124
7.3.2	Performance Tuning on Large Parallel Computers	134

7.3.3 Curved Elements	134
Bibliography	135

List of Tables

2.1	Efficiency comparison of continuous and discontinuous Galerkin methods	34
3.1	Comparison of Strouhal number for vortex shedding in flow over a square cylinder	65
4.1	Convergence rates for solving three-dimensional Taylor vortex problem .	75
4.2	Perturbation growth rates in solving three-dimensional Orr-Sommerfeld stability problem	76
6.1	Parallel efficiency in solving three-dimensional backward-facing step problem	114
6.2	Profiling of the solver and the parallel performance of velocity solves, pressure solve and the nonlinear term evaluation	115
6.3	Profiling of the preconditioned conjugate gradient in solving three-dimensional backward-facing step problem	116
7.1	Number of floating point operations vs. approximation orders $k = 1, \dots, 10$ in evaluating the nonlinear term using two different approaches	129

List of Figures

1.1	Drawing of a human heart	2
1.2	Mechanical heart valve	3
1.3	Efficiency of high-order methods	17
2.1	Accuracy of discontinuous Galerkin method on Hyperbolic problem	22
2.2	Accuracy of discontinuous Galerkin method on Elliptic problem	24
2.3	Partitioning of an unstructured mesh	26
2.4	Communication patterns in continuous and discontinuous Galerkin methods	27
2.5	Example of a non-conforming mesh	29
2.6	Two dimensional nodal sets	32
2.7	Efficiency comparison of continuous and discontinuous Galerkin methods	35
3.1	Compact stencil of the interior penalty discretization of the Laplacian, and the DG treatment of the nonlinear term	48
3.2	L^2 norm of errors in velocity and pressure versus the time step size in solving an unsteady Stokes problem	57
3.3	Maximum errors in calculated velocity and pressure versus approximating polynomial degrees in solving the Taylor vortex problem	58
3.4	L^2 velocity errors vs. element size in solving the Taylor vortex problem .	59
3.5	Perturbation energy versus normalized time in solving the Orr-Sommerfeld problem for both equal- and mixed-order methods	62

3.6	The mesh for flow past square cylinder	63
3.7	Snapshot of vorticity contours in the vortex shedding past a square cylinder at $Re = 100$	64
3.8	Geometry of the two-dimensional mechanical heart valve problem	66
3.9	Mesh used for mechanical heart valve flow problem	67
3.10	Streamwise velocity contours for flow through a two-dimensional mechan- ical heart valve	70
3.11	Streamwise and spanwise velocity along different lines for flow through a two-dimensional mechanical heart valve	71
4.1	Mesh used for three-dimensional Taylor vortex problem	73
4.2	Convergence for a three-dimensional problem with exact solution	74
4.3	Eigenfunctions of the Orr-Sommerfeld equation	77
4.4	Mesh used for three-dimensional Orr-Sommerfeld problem	78
4.5	Perturbation growth rates for three dimensional Orr-Sommerfeld problem	79
4.6	Geometry of the backward-facing step flow problem	80
4.7	Spectral element distribution for backward-facing step problem	80
4.8	Mesh used for backward-facing step problem	81
4.9	Length of the first recirculation zone for backward-facing step problem . .	82
4.10	Velocity contours for backward-facing step flow at $Re = 172$ simulated using the discontinuous Galerkin Navier-Stokes solver	86
4.11	Velocity contours for backward-facing step flow at $Re = 172$ simulated using a spectral element Navier-Stokes solver	87
4.12	Streamwise velocity components in the lower part of the geometry of the backward-facing step problem at $Re = 172$	88
4.13	Spanwise velocity components in the lower part of the geometry of the backward-facing step problem at $Re = 172$	89

4.14	Velocity contours for backward-facing step flow at $Re = 344$ simulated using the discontinuous Galerkin Navier-Stokes solver	91
4.15	Velocity contours for backward-facing step flow at $Re = 344$ simulated using a spectral element Navier-Stokes solver	92
4.16	Streamwise velocity components in the lower part of the geometry of the backward-facing step problem at $Re = 344$	93
4.17	Spanwise velocity components in the lower part of the geometry of the backward-facing step problem at $Re = 344$	94
5.1	Effect of penalty parameter on the iterative solution of the system arising from the interior penalty discretization of the Poisson equation	99
5.2	Error versus penalty parameter in solving Poisson equation using the interior penalty method	103
6.1	Performance of the code in solving the three-dimensional Orr-Sommerfeld stability problem without the projection method for the pressure equation	111
6.2	Performance of the code in solving the three-dimensional Orr-Sommerfeld stability problem with the projection method for the pressure equation .	112

Chapter 1

Introduction

1.1 Motivation

1.1.1 Mechanical Heart Valves

The heart and the circulatory system together form the cardiovascular system. The heart works as a pump that pushes blood to the organs, tissues and cells of the body. It consists of four chambers: the upper two, called the left and right atria, and the lower two, called the left and right ventricles (Fig.1.1). A muscular wall called the septum separates the left and right ventricles. Four valves, namely the tricuspid, pulmonic, mitral and aortic, regulate blood flow through the heart (Fig.1.1). The tricuspid valve regulates blood flow between the right atrium and the right ventricle. The pulmonic (or pulmonary) valve controls blood flow from the right ventricle to the pulmonary artery, which carries the deoxygenated blood to the lung to pick up oxygen. The mitral valve lets oxygenated blood from the lungs pass from the left atrium into the left ventricle. The aortic valve controls blood flow from the left ventricle to the aorta, which carries the oxygenated blood to all parts of body. The tricuspid, pulmonic and aortic valves all have three leaflets, while the mitral valve has two leaflets. The general function of all these valves is to prevent retrograde blood flow and regurgitation.

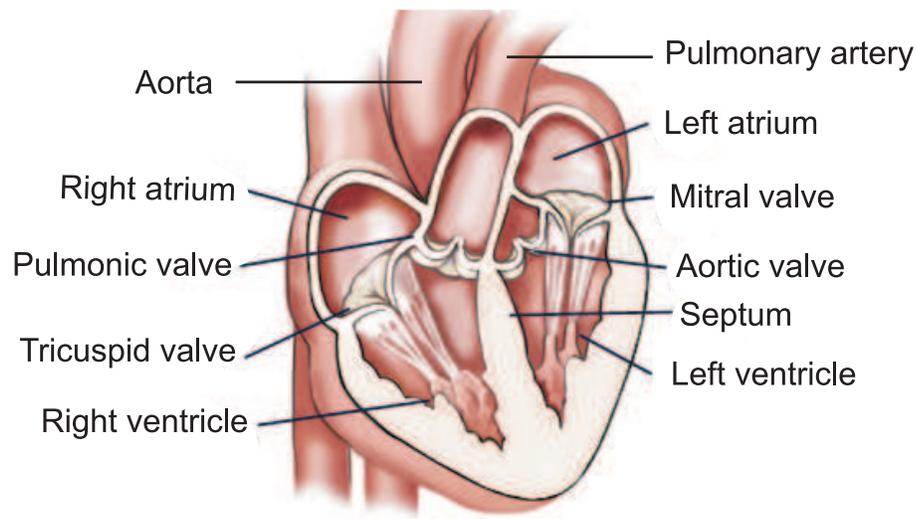


Figure 1.1: Drawing of a human heart [1]

However, these valves may malfunction due to valvular disease. When valvular disease becomes significant, the condition of the patient deteriorates and valve replacement becomes necessary. Approximately 180,000 valve replacements are performed each year worldwide [123]. Replacement valves are either tissue valves or mechanical heart valves (MHVs). Compared to mechanical heart valves, tissue valves resemble more closely the geometry and function of native physiological valves and result in better haemodynamic performance. However, the major drawback of tissue valves is their limited life-span, on average 10 – 15 years. This makes these valves suitable for elderly patients, but less attractive to younger individuals. MHVs, on the other hand, exhibit superior durability and are thus preferred in the younger patients with life expectancies of longer than 15 years.

The design of MHVs and their surgical implantation have evolved during the last several decades, representing a compromise between the ideal configuration and what is practicable. The common design includes a flow occluder system housed in a frame or a ring that is fitted with an appropriate sewing cuff (see Fig. 1.2a). From a hydrodynamic point of view, the major difference between the MHVs and the natural heart valves

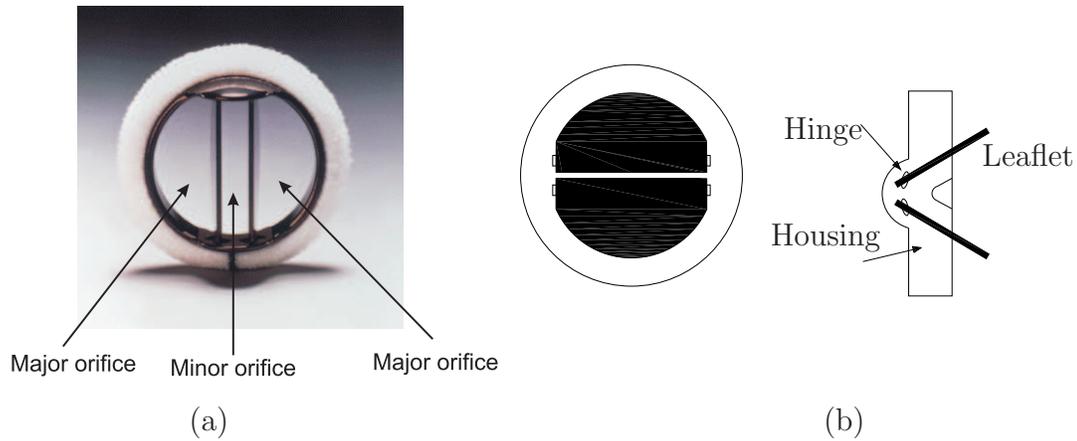


Figure 1.2: (a) Downstream view of a St. Jude Medical bileaflet heart valve [2], (b) schematic of a generic bileaflet heart valve, demonstrating the leaflets, hinges and housing.

is the positioning of the occluder system. In the natural design, the aortic valve opens completely to produce an unobstructed flow passage. On the other hand, a manufactured valve inevitably contains some obstructions embodying the occluder mechanism around which the blood must flow. This causes a smaller effective cross sectional area and thus non-physiological orifice (stenosis) flow with higher energy dissipation, locally higher kinetic energy, and greater turbulence.

The most recent and popular MHV design is the bileaflet valve, first introduced in 1977 by St. Jude with the standard model. In a bileaflet MHV, the occluder mechanism comprises two half discs that pivot to the center of the ring (see Fig.1.2). The leaflets, together with the housing frame, are coated with pyrolytic carbon. The shape of the discs, the opening angle and the hinge configuration vary in different designs. When open, the valve is divided into two major orifices on the sides and one minor orifice in the center (see Fig. 1.2a).

1.1.2 Design and Clinical Impact

Despite improvements in the performance of MHVs during the past five decades, MHVs still induce very significant complications due to thrombosis and hemolysis. Thrombosis is defined as the formation of a blood clot within a blood vessel during life and hemolysis is referred to as the lysis of red blood cells. These complications are promoted by non-physiological flows characterized by high incidence of flow stagnation, separation and turbulence. These non-physiological flow patterns are related to valve design non-optimality. Realizing this problem thus puts forward the need to improve current valve designs and/or strive for radical new designs.

Many studies have investigated the associations of fluid mechanics with thrombosis and hemolysis (e.g., [17, 96, 107, 109, 100, 91, 85, 76, 66, 84, 55]). Thrombus formation, which is a consequence of platelet activation, can be correlated with shear stress and turbulence and the exposure time of a platelet to these fluid mechanical effects. Employing a laminar rotational viscometer, Brown et al. [17] reported that a shear stress of 5Pa resulted in platelet aggregation. They also concluded that platelet lysis occurred at shear stresses of 10Pa, and a shear stress of 25Pa resulted in platelet fragmentation. On the other hand, in a laminar tube flow experiment, Ramstack et al. [96] considered higher shear stresses of 30, 75 and 100Pa and average residence times of 25 – 165ms. While they reported platelet activation, they did not observe any platelet lysis. This shows that exposure time affects platelet damage: longer exposure time requires lower level of stress to damage platelets and *mutatis mutandis*. To study the effects of turbulence, Stein and Sabbah [109] and Smith et al. [107] conducted *in vivo* experiments on dogs. In each dog, they established two arteriovenous shunts, one from each femoral artery to the contralateral vein. Only one shunt contained a turbulence-producing device; otherwise the shunts were identical in shape, size and material. They found that in each dog, more thrombi, by weight, accumulated in the turbulent shunt than in the laminar shunt. Moreover, they reported that the weight of thrombi accumulating in the turbulent system appeared to

be related to the turbulence intensity.

Analogously, researchers have correlated hemolysis with shear stress and Reynolds shear stress. In a laminar Couette flow, Paul et al. [91] used plasma hemoglobin to quantify red cell damage due to shear stress. For exposure times of 25–1250ms and shear stresses ranging from 25 to 450Pa, they reported a remarkably low level of hemolysis. However, a significant increase in red cell damage was reported for shear stresses greater than 450Pa and residence times longer than 620ms. Sallam and Hwang [100] using a two dimensional submerged turbulent jet experiment, presented a Reynolds shear stress of 400Pa as a threshold above which lethal red cell damage occurs.

The fact that current MHV designs cause non-physiological flow patterns has also been realized in the literature [15]. The evolution of valve configurations from the cage and ball, the earliest design, to the bileaflet, the most recent design, has been aimed at achieving more natural flow characteristics, although it has not been completely successful. The general features of flow through different MHV designs have been characterized by in vitro experiments (e.g., [124]). In the open position of a bileaflet valve, the three orifices (Fig. 1.2a) divide the forward flow into two lateral jets and one central jet. In a St. Jude Medical valve, the maximum velocity along these jets reaches approximately 2m/s at peak systole. After traveling 20 – 30mm downstream of the valve, these jets merge into a central flow region. Regions of flow separation lie around the jets adjacent to the channel wall. Small regions of low-velocity reverse flows occur adjacent to the pivot/hinge mechanism of the valve. Turbulent shear stresses as high as 170Pa along large velocity gradients approximately 10mm downstream of the leaflets have been measured. Higher turbulent shear stresses are expected closer to the valve. The leakage flow primarily occurs around the hinge mechanism and the peripheral gap between the housing and the leaflets in the closed positions. Maximum leakage velocities of approximately 1 – 4m/s and maximum turbulent shear stresses of approximately 200 – 700Pa around the hinge mechanism have been reported in different bileaflet valve designs at the mitral position

[124].

Current research related to bileaflet valve design is concentrated on examining critical valve parameters including leaflet shape, hinge mechanism, and clearance between the leaflet edges and the valve housing. For example, Grigioni et al. [51] have used a laser Doppler anemometer (LDA) to study the turbulence characteristics in the wake of two similar valve models, but with different leaflet curvatures. They have found distinct turbulence fields suggesting that leaflet curvature is a key design parameter. Leo et al. [75] have also used LDA measurements to study the hinge region flow during the closure period of the CarboMedics bileaflet valve. To assess the potential of blood cell damage, they have quantified the associated turbulent flow field and compared it with those of two other models, the St. Jude Medical and the Medtronic Parallel.

1.1.3 Synergy of Simulation and Experiment for Design

Refinement of current MHV designs as well as introduction of radical new designs, with the aim of zero or at least minimal thromboembolic complications, can be achieved effectively through both physical experimentations and computer simulations in a complementary manner. Computer simulations offer unique capabilities of obtaining full-field attributes and conducting inexpensive parametric studies which are needed for evaluating preliminary designs. Through the preliminary design stage, the overall bulk hydrodynamic performance of a proposed valve such as pressure drop and regurgitant volume (closing and leakage volumes) are examined. Since current MHV designs satisfy the ISO and GEN Heart Valves Standards, these tests are only necessary for valves with substantially new designs. However, all current valve designs, as well as any new designs, must be evaluated for their potential for blood cell damage and platelet activation. Using accurate computational techniques, in this stage, can reduce prototype fabrications as well as expensive and long physical experimental runs. Finally, to test the preliminary designs in physical situations and obtain design parameters unachievable by computer

simulations, in vitro and in vivo experiments must be employed.

Different experimental techniques are currently used for in vitro studies. These include flow visualization, LDA and particle image velocimetry (PIV). For in vivo testing, magnetic resonance imaging (MRI) is a popular approach. Since we concentrate in this work on computational methods, we will not explain these experimental techniques and refer the reader to the article by Yoganathan et al. [124] for the present state of the art.

1.2 Mechanical Heart Valve Simulations

In this section, we first review previous MHV flow simulations. Based on the complexities of the simulation, we then propose our approach, direct numerical simulation, and analyze the resolution requirements and the associated cost.

1.2.1 Previous Simulations

Several mechanical heart valve simulations, in two and three space dimensions, have been reported in the literature (see the review article [123]). We here review only four representative papers.

Although it was two dimensional and lacked real clinical input parameters such as Reynolds number, the pioneering work by McQueen and Peskin in 1983 [83] serves as an excellent paradigm demonstrating how computer simulations can actually be used in the design of prosthetic mitral heart valves. Their numerical procedure solved the coupled equations of blood flow and heart wall mechanics and heart valve motion. They chose two design parameters: the radius of curvature of the occluder, and the location of the pivot point. For different values of these design indices, they examined three performance criteria including the net stroke volume, the mean forward pressure difference and the peak anterior velocity, in the search for an optimum result. They concluded that the best overall valve had a radius of curvature equal to 1.5 times its diameter and a pivot

point located 0.39 mitral-ring diameters from the anterior border of the mitral annulus.

Motivated by the structural failure reports of prosthetic mitral heart valves [37], Lai et al. [74] used numerical simulations to analyze the role of bileaflet tip geometry in the incidence of a negative pressure transient on the atrial side. They employed a two dimensional finite volume method capable of capturing the leaflet motion via an arbitrary Lagrangian-Eulerian (ALE) technique. Their results did not show significant differences of pressure as a function of different leaflet tip angles. However, they reported the importance of the leaflet tip velocity on the fluid dynamics during the closing phase.

King et al. [70] conducted a parametric study on the design of bileaflet MHVs at an aortic site. They used a finite element based commercial code to study the effect of two leaflet opening angles on the time-dependent laminar flow through a three dimensional model of the valve. This model was geometrically symmetric and did not contain the hinge configuration. They found that as the leaflet opening angle increased the flow downstream of the valve became more centralized and the wakes of the leaflets decreased in size.

In an attempt to find out the effect of implantation techniques such as misalignment of the valve on the thromboembolic potential of MHVs, Bluestein et al. [16] simulated blood flow through a two dimensional model of an implanted St. Jude Medical bileaflet heart valve in the aortic position. They used a $k - \omega$ turbulence model and the FIDAP CFD package (Fluent) to study turbulence, shed vortices in the wake of the leaflets and the recirculation locations during the deceleration phase of systole. They found that a tilted valve generates narrower and stronger jets through the valve's orifices and a wider wake with larger vortices. They also reported the typical high shear layers at the interface of jets and recirculation zones which contribute to platelet activation.

1.2.2 Complexities of Simulation

The design and clinical implications of the above-cited efforts, as well as literature we have not reviewed, have been very limited because all studies have considered highly simplified models of the real situation. The complexities involved in the real problem are:

- a three dimensional complex geometry,
- pulsatility, transition and highly anisotropic and intermittent turbulence at Reynolds number $(Re) \approx \mathcal{O}(7000)$,
- flow-structure interaction due to the compliant walls of the heart and arteries, and motion of the valve leaflets and their possible fluttering,
- non-Newtonian behavior of blood at low shear rates,
- the particulate nature of the blood once turbulence scales become comparable to the blood cell sizes.

Any computer simulation aiming to reveal accurate and useful results must first fully address the individual difficulties and then integrate them all together. Simulations involving all these complexities not only require significant advances in computational techniques, but are also a challenging task for today's largest parallel computers. To be pragmatic, we focus, for now, on addressing the first two complexities, namely the transitional and turbulent flow through complex MHV geometry. For simulating such flow, we propose direct numerical simulation, which we will describe in the following subsection.

1.2.3 Direct Numerical Simulation

In simulating fully developed high Reynolds-number turbulent flows, the Reynolds-averaged Navier-Stokes equations (temporal averaged models) and large eddy simulations (spatial

averaged models) have proven to be effective. However, these averaged techniques are not amenable to simulating low Reynolds-number transitional and turbulent flows, since these flows lack an inertial subrange [41]. Hence, the only reliable approach is to resolve all active temporal and spatial scales through direct numerical simulations (DNS).

The most important question arising when one proposes to carry out DNS of a turbulent flow is what the computational costs associated with resolving all the active spatial and temporal scales of the turbulence are. To answer this, we need an estimate of the required resolution.

Length Scale Estimation and Resolution Requirement

An estimate of smallest length scales involved in a turbulent flow, which are also called Kolmogorov length scales, can be derived by using Kolmogorov's universal equilibrium theory [114]. The basis of this theory is the assumption that the time scale associated with the small scale motion of the turbulence is much smaller than that of the mean flow and large scale motions. The small scale only depends on the rate of energy cascading from the large scales and of course the kinematic viscosity ν . The former is balanced with the rate of energy dissipation ϵ occurring on the small scale because the net rate of change of the small scale energy depends on the mean flow. This leads to an expression for the small length scale η of the form

$$\eta \equiv \left(\frac{\nu^3}{\epsilon}\right)^{1/4}. \quad (1.1)$$

To proceed further, we need an estimate of ϵ . We use an inviscid estimate which is valid for large Reynolds numbers [114]. Due to diminished viscous dissipation of the large scale fluctuations at high Reynolds numbers, the rate of energy transferred to the small scales (which is balanced with ϵ) is proportional to the square of the velocity of the large scales u^2 and the inverse of their time scale u/l , where l denotes the length of large scales,

$$\epsilon \propto \frac{u^3}{l}. \quad (1.2)$$

Substituting 1.2 in 1.1 leads to

$$\frac{\eta}{l} \equiv (ul/\nu)^{-3/4} = (Re_l)^{-3/4}, \quad (1.3)$$

where Re_l is the Reynolds number associated with the largest (integral) turbulent scale. If we are to use equation 1.3 to calculate the resolution requirement for turbulence simulation through MHVs, we must assume that equation 1.1 is valid for pulsatile flow since time scales related to the pulsatile flow might be comparable with those of small scales, which might lead to violation of Kolmogorov's universal equilibrium theory. We calculate Re_l for the maximum blood flow rate of $30L/min$ occurring at the peak of the cardiac cycle (systole). Other researchers have also used this value for the maximum flow rate (see e.g [18]). The integral length scale l is chosen to be the typical diameter of a MHV (25mm) and the kinematic viscosity of blood is assumed to be $3.5 \times 10^{-6} m^2/s$. Based on the forgoing parameters Re_L is 7000. Thus,

$$\frac{\eta}{l} \equiv (7000)^{-3/4} = 0.0013,$$

or equivalently the smallest scale is $0.0013 \times 25mm = 33\mu m$.

We now discuss length scale estimations resulting from experiments. There have been only a few papers dealing with turbulence characteristics (in particular scale analysis) downstream of MHVs. Among those, the most recent and perhaps most reliable one is that of Liu et al. [78]. They conducted a series of in vitro measurements downstream of three bileaflet MHVs employing a bidimensional LDA. The sampling area of the LDA, which is a measure of the resolution, was an ellipsoidal region with a short axis of $50\mu m$ and a long axis of $400\mu m$. Liu et al. measured the mean (U) and fluctuating (u) velocities and then based on an autocorrelation coefficient they calculated the integral and Taylor micro time scales. Based on Taylor's hypothesis, which states that the entire flow remains unchanged as it passes by a certain point ($\frac{\partial}{\partial t} = -U \frac{\partial}{\partial x}$, see [61], section 1.8), they calculated the integral length scale and Taylor micro length scale λ . They also

assumed that the fluctuating scales are isotropic and used the formula

$$\epsilon = 15\nu \overline{\left(\frac{\partial u}{\partial x}\right)^2} = 15\nu \frac{\overline{u^2}}{\lambda^2} \quad (1.4)$$

to calculate the dissipation rate ϵ . Finally they used equation 1.1 to calculate the smallest length scales. They reported scales as small as $20 - 30\mu\text{m}$.

Since Liu et al.'s measurements led to $u/U > 0.1$, the Taylor hypothesis is not valid [114] despite it being used throughout the calculation. Moreover, due to the small Reynolds number turbulent flow, the isotropy assumption does not seem to yield accurate results. Liu et al. did not report any uncertainty for their results. To have a sense of uncertainties associated with LDA measurements, we refer to the recent work by DeGraaff and Eaton [33]. For their high resolution LDA (with sampling volume size of $35\mu\text{m}$ in diameter and $60\mu\text{m}$ in length) for turbulence at $Re = 1450$, they reported up to 10% uncertainty for the Reynolds shear stress. They also mentioned that near wall measurements are subjected to higher uncertainties. Considering the uncertainties involved, assumptions often made (e.g. isotropy and validity of the Taylor hypothesis) and the inherent order of magnitude estimate of equation 1.1, all we expect from this length scale analysis is just an order of magnitude estimate of the smallest scales. Therefore, based on the smallest length scale obtained from Kolmogorov theory ($\eta = 33\mu\text{m}$) and that obtained from the Liu et al.'s experiment ($\eta = 20 - 30\mu\text{m}$), we consider the smallest length scale η to be $20\mu\text{m}$ in all three directions. Using the integral length scale l_1 of 25 mm in the spanwise and radial directions and the computational domain size l_2 of 250mm in the streamwise direction leads to the total grid points N

$$N = \left(\frac{l_1}{\eta}\right)^2 \left(\frac{l_2}{\eta}\right) = \left(\frac{25 \times 10^3}{20}\right)^2 \left(\frac{250 \times 10^3}{20}\right) = 2750^3.$$

However, the resolution of actual DNS on simple geometries has typically been one order of magnitude less in each dimension than the resolution deduced from the Kolmogorov theory [86]. Moser and Moin [88] also noted that the actual dissipation in turbulent plane channel flow occurs on a scale 15 times larger than the Kolmogorov scale. Thus, based

on the preceding analysis the resolution requirement for the DNS simulation $\approx \mathcal{O}(300^3)$ or roughly 27 million grid points. We now compare this estimate with what is available as the resolution requirement for similar flows.

Using a spectral element solver with an explicit-implicit time integration scheme, Fischer et al. [41, 40] have recently performed DNS of weakly turbulent flows in two vascular bifurcations, namely a stenosed carotid artery and an arteriovenous graft. The size of geometries in these simulations were comparable with that in the proposed MHV simulation and Reynolds numbers based on the peak systole bulk velocity were 1350 and 1200 in the carotid artery and arteriovenous graft simulations, respectively. In the carotid artery case, Fischer et al. used a mesh of 2544 hexahedral elements with orders $k = 7, 9$, and 11 and time step size $\Delta t = 5 \times 10^{-6}$. They found resolution-independent results in calculating mean and root-mean-square (RMS) statistics collected for 30 flow-through times at $k = 9$, yielding roughly two million gridpoints as resolution requirements. The simulation at these resolution required 20 hours of CPU time on 256 processors of the Alpha-based TCS1 at the NSF Pittsburgh Supercomputing Center. In the arteriovenous graft simulation, the mean and RMS quantities were in good agreement with experimental data when using roughly four million gridpoints (2640 hexahedrons of order $k = 12$).

We note that if one uses the typical explicit-implicit (semi-explicit) time integration schemes (such as used in Fischer et al.'s solver), the simulation is well resolved for the physical time scales. This conjecture is supported by the fact that the time step restriction in an semi-explicit scheme is governed by the CFL condition which leads to a time step size often smaller than the Kolmogorov time scale

$$\tau = \sqrt{\nu L/u^3}.$$

(e.g. [122]). Moreover, in their carotid artery simulations, Fischer et al. reported physical time scales as small as 5×10^{-3} which is much larger than the time step size they used (5×10^{-6}).

These verify our estimate of the resolution requirements for DNS of MHV flows.

MHV flows occur at a few-fold higher Re than those considered by Fischer et al., and accordingly DNS of such flows require several times higher resolutions (approximately 27 million gridpoints). Consequently, we need computational resources consisting of several hundred to several thousand processors as well as highly efficient parallel flow solvers.

1.3 Methodology of Direct Numerical Simulation

The governing equations we seek to solve are the unsteady incompressible Navier-Stokes (NS) equations introduced in the third chapter. The speed of a particular solver for the NS equations depends on hardware, methodology (algorithm) and implementation. The last two factors, namely efficient schemes for the Navier-Stokes equation and their implementation, are the subjects of this thesis.

Solution of the unsteady NS equations consists of two parts: temporal discretization and spatial discretization. For temporal discretization several options are possible: fully implicit schemes, semi-Lagrangian schemes, and semi-explicit schemes with explicit treatment of the non-linear term and implicit treatment of the Stokes operator. The first method requires Newton iterations and the solution of nonsymmetric systems, rendering the solution algorithm inefficient for time-accurate calculations [122]. The semi-Lagrangian schemes do not exhibit good parallel performance, despite allowing large time step sizes based on large CFL numbers ($\approx \mathcal{O}(10)$) [122].

Semi-explicit methods, on the other hand, are simple, efficient and widely used in the DNS of transitional and turbulent flows. Thus, for temporal discretization, we propose to adopt a semi-explicit scheme. Below, we discuss the importance of high-order approximation, before evaluating possible techniques for the spatial discretizations.

1.3.1 Dispersion Error and High-Order Approximation

In transitional and turbulent MHV flows, physical dissipation is very small; thus, it is essential that the numerical solution scheme yield minimal dissipative and dispersive errors. Low order approximations leave high-wave number components of the solution unresolved. Due to the absence of physical dissipations, these errors accumulate and pollute the solution over long integration time, yielding high dispersion. However, high-order approximations, at the same resolution, accurately propagate a larger portion of the solution spectrum. Thus, they are capable of maintaining high accuracy over prolonged integration time (i.e., yielding low dispersive errors) [50, 36]. We now illustrate this fact with a simple example.

Let us consider an unsteady advection equation in the d -dimensional domain $\Omega = [0, 2\pi]^d$ with periodic boundary conditions in the form

$$\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{c}u) = 0 \quad \text{in } \Omega, \quad (1.5a)$$

$$u(\mathbf{x}, 0) = \cos(\boldsymbol{\omega} \cdot \mathbf{x}) \quad \text{in } \Omega, \quad (1.5b)$$

with exact solution $u(\mathbf{x}, t) = \cos(\boldsymbol{\omega} \cdot (\mathbf{x} - \mathbf{c}t))$. For simplicity, we assume isotropic wave number and wave velocity vectors, i.e., $\boldsymbol{\omega} = (\omega, \dots, \omega)$ and $\mathbf{c} = (c, \dots, c)$. We consider solving eq. 1.5 using a finite difference scheme of order k over q periods in time; that is $t = 2\pi q/\omega c$. The approximate solution has the form

$$u_h = \cos(\boldsymbol{\omega} \cdot (\mathbf{x} - \mathbf{c}_k t)), \quad (1.6)$$

where the \mathbf{c}_k depends on the number of grid points N_k and the approximation order k . Following [53], one can easily show that to achieve a phase error equal to or smaller than ϵ , the number of grid points per wavelength, M_k , scales as

$$M_k = \frac{N_k}{\omega} \propto \left(\frac{q}{\epsilon}\right)^{\frac{d}{k}}, \quad k = 2, 4, \dots \quad (1.7)$$

This yields the computational work per wavelength, W ,

$$W \propto (k)^d M_k \propto (k)^d \left(\frac{q}{\epsilon}\right)^{\frac{d}{k}}. \quad (1.8)$$

The above relation, for approximation orders $k = 2, 4, 8$ and 16 , is depicted in Figs. 1.3a, b and c for $d = 1, 2$ and 3 , respectively. It is clear from the figures that as q/ϵ grows, higher order approximations become increasingly efficient. However, there is a limit as to how high one should choose the approximation order; in this case the limit is approximately $k = 8$. Also notable from Figs. 1.3a, b and c is the fact that the efficiency gain due to higher order methods is greater for multidimensional problems than for one-dimensional problems. For instance, comparing Fig. 1.3a with Fig. 1.3c, we observe that while in the one-dimensional problem for $q/\epsilon = 1000 - 10000$, the cost of order $k = 8$ is approximately ten-fold lower than the cost of order $k = 2$, in the three-dimensional problem for the same range of q/ϵ , the cost of order $k = 8$ is approximately one hundred- to one thousand-fold lower than the cost of order $k = 2$.

Although the efficiency of high-order methods has been demonstrated using the simple advection equation, a similar result holds for more complicated problems where multidimensional simulations with high accuracy over long integration times are required. This is certainly the case for the problem we are aiming to solve, namely the direct numerical simulation of transitional and turbulent flows through three-dimensional mechanical heart valve geometries. Since these flows involve small and large scales, and since all these scales must be resolved in direct numerical simulations, a high level of accuracy is definitely required. Moreover, long integration times (at least ten-fold higher than the cardiac cycle [40]) are also essential to damp out the artificial transients caused by uncertainties in initial and boundary conditions and to obtain accurate statistics.

1.3.2 Choice of Spatial Discretization

Spectral methods have been traditionally used for the DNS of transition and turbulence [90]. However, despite their high accuracy and efficiency, they are limited to very simple geometries such as boxes. Among methods suitable for complex geometries are finite volume methods, continuous Galerkin (CG) methods and discontinuous Galerkin (DG)

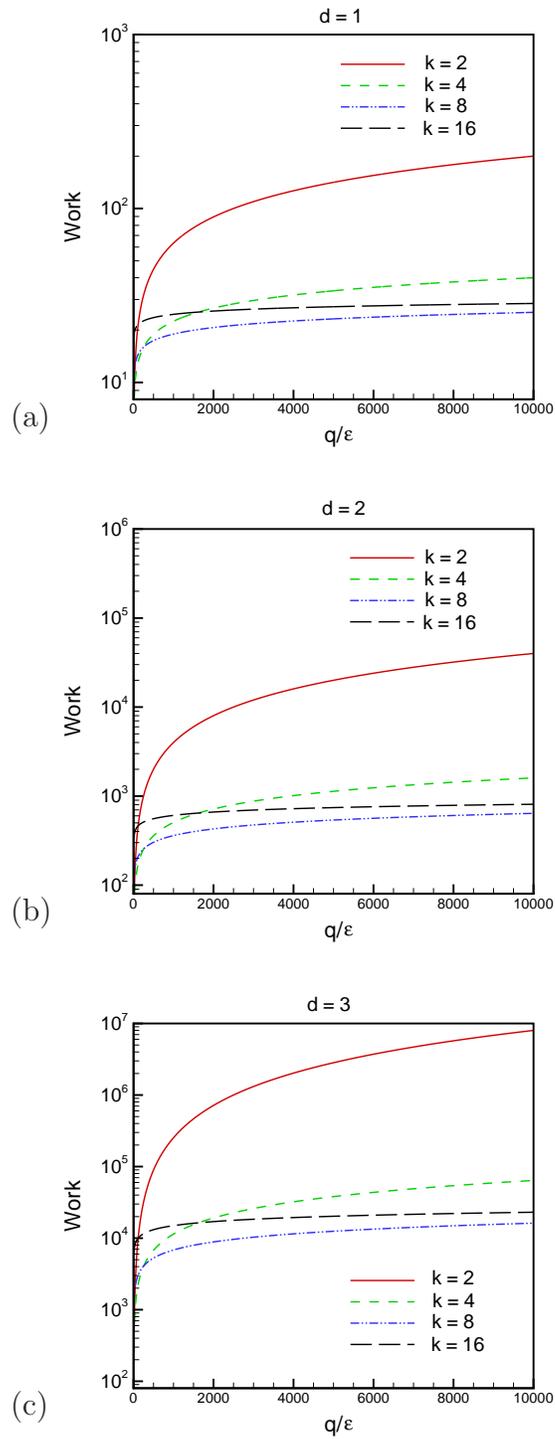


Figure 1.3: Work vs. q/ϵ in solving the d -dimensional advection equation (1.5) using a finite difference method with orders 2, 4, 8 and 16; (a) $d = 1$; (b) $d = 2$; (c) $d = 3$. q/ϵ represents the ratio of number of periods in time (q) to a given accuracy tolerance (ϵ).

methods.

While finite volume methods are locally conservative and suitable for convection-dominated flows, their high-order versions (developed by Barth [11]) require large stencil sizes, yielding poor parallel efficiency.

High-order continuous Galerkin (CG) methods, on the other hand, offer both geometric flexibility and high parallel efficiency. However, the implementation of these methods is rather complex, in particular in parallel and adaptive algorithms. Moreover, in convection-dominated flows, they yield spurious oscillations if mesh elements are not sufficiently fine to yield a grid Reynolds number of $\mathcal{O}(1)$ [117]. The grid Reynolds number is defined based on the local element size and the local velocity. As the (global) Reynolds number increases the restriction on the element sizes become more severe, rendering the numerical scheme inefficient. To circumvent this drawback, such schemes require special modifications such as filtering [36] or stabilizations [45, 64].

High-order DG methods are simple methods that combine the advantages of both finite volume and continuous Galerkin methods, namely robustness at high Reynolds numbers flows (i.e., yielding stable solutions even on coarse meshes with large grid Reynolds numbers), geometric flexibility, high-parallel efficiency, local conservativity and adaptivity. In view of these advantages, we propose to adopt high-order DG methods for the spatial discretization of the NS equations.

1.4 Objectives

Discontinuous Galerkin methods for the *stationary* incompressible Navier-Stokes equations have been recently developed [48, 23]. However, the corresponding numerical solution procedure for the *unsteady* incompressible Navier-Stokes equations have not yet been proposed. Therefore, the objectives of this thesis are:

- to develop an efficient high-order discontinuous Galerkin scheme for solving the

unsteady incompressible Navier-Stokes equations using triangular and tetrahedral elements in two and three space dimensions, respectively; and

- to implement the scheme in parallel and verify the accuracy and stability of the method by solving popular benchmarking problems in two and three space dimensions.

The ultimate goal of the work is to use the developed code to carry out parametric studies of contemporary mechanical valve designs and even new configurations, with the aim of suggesting optimal designs based on performance criteria such as the desired pressure drop, regurgitant volume, and minimal thromboembolism. However, this solver will have much broader application. It can be used in simulating blood flow in other parts of the vascular system for both laminar and transitional flow regimes. These simulations may have implications in medical device design [73], disease research (such as investigating the effect of blood flow on blood cells and arterial walls) [65, 112, 92, 79] and surgical planning (such as analysis of different options in arterial bypass surgery) [111].

The remainder of the thesis is structured as follows. In chapter 3, we introduce our scheme and its validation for two dimensional problems. This is followed by validation in three dimensions in chapter 4. In chapter 5, we address the issue of choosing the penalty parameter for the interior penalty method. In chapter 6, some implementation details and a performance study are presented. Finally, in the conclusions chapter, we summarize the work and explore some future directions. Note that chapters 3 and 5 are based on the articles [104, 105], respectively. Although this may cause some overlap among different chapters, it contributes to the readability of each individual chapter.

Chapter 2

Overview of Discontinuous Galerkin Methods

In this chapter, we introduce DG methods for hyperbolic and elliptic problems and present some results on their accuracy in comparison with the CG method. We then give details of their important features, again as compared with the CG methods. We end the chapter by mentioning some drawbacks of DG methods.

2.1 DG Method for a Hyperbolic Equation

The first DG method was introduced in 1973 by Reed and Hill [97] in solving a hyperbolic equation (the neutron transport equation)

$$u + \nabla \cdot (\mathbf{c}u) = 0 \quad \text{in } \Omega, \quad (2.1)$$

where \mathbf{c} is the characteristic velocity vector. To develop the DG method, the above equation is multiplied by a test function v and then integrated over a subset of Ω , K . After integration by parts we obtain

$$\int_K u v d\mathbf{x} + \int_K \nabla v \cdot (\mathbf{c}u) d\mathbf{x} - \int_{\partial K} \mathbf{n} \cdot (\mathbf{c}u) v ds = 0, \quad (2.2)$$

where \mathbf{n} is the unit outward normal of ∂K . We now construct a triangulation of Ω , $\mathcal{T}_h = \{K\}$. The discontinuous approximate solution u_h , belonging to a polynomial space P_K^k of degree at most k , is defined as the solution of the following system:

$\forall K \in \mathcal{T}_h$:

$$\int_K u_h v_h d\mathbf{x} + \int_K \nabla v_h \cdot (\mathbf{c} u_h) d\mathbf{x} - \int_{\partial K} (\widehat{\mathbf{n} \cdot \mathbf{c} u_h}) v_h ds = 0, \quad \forall v_h \in P_K^k. \quad (2.3)$$

To complete the definition of the DG method, we define the numerical flux $(\widehat{\mathbf{n} \cdot \mathbf{c} u_h})$

$$(\widehat{\mathbf{n} \cdot \mathbf{c} u_h})(\mathbf{x}) = \mathbf{n} \cdot \mathbf{c} \lim_{s \downarrow 0} u_h(\mathbf{x} - \mathbf{c}s). \quad (2.4)$$

The term $\lim_{s \downarrow 0} u_h(\mathbf{x} - \mathbf{c}s)$ represents the upstream value of u_h , where “upstream” is defined with reference to the characteristic direction \mathbf{c} . Consequently, the degrees of freedom of the approximate solution u_h in the element K can be calculated using the values of u_h upstream of the characteristics impinging on ∂K . If elements are suitably sorted according to the characteristic directions, the solution can be calculated on an element-by-element basis.

We note that besides the above upwind numerical flux, there are other possible choices for the numerical flux including the Godunov flux, the Engquist-Osher flux and the Lax-Friedrichs flux [27]. The Lax-Friedrichs flux is particularly attractive due to its simplicity for arbitrary mesh configurations. This flux is used for the discretization of the nonlinear term in the Navier-Stokes equations as shown in chapter 3.

This feature of the DG formulation, namely that the direction of the convection is taken into account, is absent in the CG formulation, which does not respect the direction of characteristics. This is why DG methods lead to superior results in problems where convection plays an important role. To illustrate this fact, we present the numerical results of Giraldo [47] on passive advection of a cylindrically shaped perturbation on the surface of a sphere. The perturbation represented a geopotential height, and the test was carried out in the context of solving the shallow water equations with relevance in climate modeling. Giraldo compared the performance of both DG and CG discretizations. In

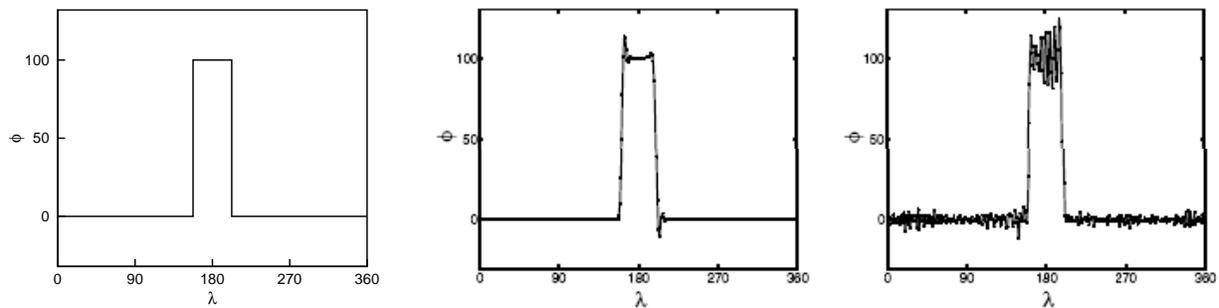


Figure 2.1: Profile of geopotential height, Φ , along the equator undergoing passive advection for 12 days; the exact profile (left); the profile calculated using the DG method (center); the profile calculated using the CG method (right). From Giraldo [47].

both discretizations, a mesh consisting of 64 triangular element of order $k = 1$ was used. The results after 12 days of integration are shown in Fig. 2.1. It is clear that the DG method yielded a superior solution compared to the CG method. While the DG method yields relatively smooth solution with only minimal, and more importantly, localized overshoots and undershoots, the CG method yielded a spurious solution over the entire domain (global effect). We note that similar undesirable results may be also observed when simulating incompressible flows using a CG method, despite the absence of physical discontinuities. In an unresolved (or marginally resolved) solution, sharp gradients can introduce artificial discontinuities and similar global spurious oscillations will pollute the solution.

2.1.1 DG Method for an Elliptic Problem

We here describe the DG formulation of the elliptic problem

$$u - \Delta u = f \quad \text{in } \Omega, \quad (2.5a)$$

$$u = 0 \quad \text{on } \partial\Omega, \quad (2.5b)$$

The Laplacian is first written as two first-order operators by introducing the auxiliary variable $\boldsymbol{\sigma}$:

$$\nabla u = \boldsymbol{\sigma}, \quad (2.6a)$$

$$u - \nabla \cdot \boldsymbol{\sigma} = f. \quad (2.6b)$$

Following the same procedure as the hyperbolic case, we then multiply the above equations with test functions $\boldsymbol{\tau}$ and v and integrate over the element K . After integration by parts we obtain

$$-\int_K \nabla \cdot \boldsymbol{\tau} u d\mathbf{x} + \int_{\partial K} \boldsymbol{\tau} \cdot \mathbf{n} u ds = \int_K \boldsymbol{\sigma} \cdot \boldsymbol{\tau} d\mathbf{x}, \quad (2.7a)$$

$$\int_K u v d\mathbf{x} + \int_K \boldsymbol{\sigma} \cdot \nabla v d\mathbf{x} - \int_{\partial K} \boldsymbol{\sigma} \cdot \mathbf{n} v ds = \int_K f v d\mathbf{x}, \quad (2.7b)$$

Now the approximate solutions $u_h \in P_K^k$ and $\boldsymbol{\sigma}_h \in (P_K^k)^d$ are defined as the solution of the system

$\forall K \in \mathcal{T}_h$:

$$-\int_K \nabla \cdot \boldsymbol{\tau}_h u_h d\mathbf{x} + \int_{\partial K} \boldsymbol{\tau}_h \cdot \mathbf{n} \hat{u}_h ds = \int_K \boldsymbol{\sigma}_h \cdot \boldsymbol{\tau}_h d\mathbf{x} \quad \forall \boldsymbol{\tau}_h \in (P_K^k)^d, \quad (2.8a)$$

$$\int_K u_h v_h d\mathbf{x} + \int_K \boldsymbol{\sigma}_h \cdot \nabla v_h d\mathbf{x} - \int_{\partial K} \hat{\boldsymbol{\sigma}}_h \cdot \mathbf{n} v_h ds = \int_K f v_h d\mathbf{x} \quad \forall v_h \in P_K^k. \quad (2.8b)$$

The numerical fluxes \hat{u}_h and $\hat{\boldsymbol{\sigma}}_h$ are defined as

$$\hat{u}_h(\mathbf{x}) = \frac{1}{2}(u(\mathbf{x}^+) + u(\mathbf{x}^-)), \quad (2.9a)$$

$$\hat{\boldsymbol{\sigma}}_h(\mathbf{x}) = \frac{1}{2}(\nabla u_h(\mathbf{x}^+) + \nabla u_h(\mathbf{x}^-)) - \mu(u(\mathbf{x}^+) - u(\mathbf{x}^-))\mathbf{n}, \quad (2.9b)$$

where $\mathbf{x}^+ = \lim_{s \downarrow 0}(\mathbf{x} + s\mathbf{n})$ and $\mathbf{x}^- = \lim_{s \downarrow 0}(\mathbf{x} - s\mathbf{n})$. For $\partial K \in \partial\Omega$, the numerical fluxes are defined as

$$\hat{u}_h(\mathbf{x}) = u(\mathbf{x}^+), \quad (2.10a)$$

$$\hat{\boldsymbol{\sigma}}_h(\mathbf{x}) = \nabla u_h(\mathbf{x}^+) - \mu u(\mathbf{x}^+)\mathbf{n}, \quad (2.10b)$$

We note that unlike the hyperbolic case where the fluxes depend on the solution on one side of ∂K , here the numerical fluxes depend on the value of u on both sides of ∂K . This

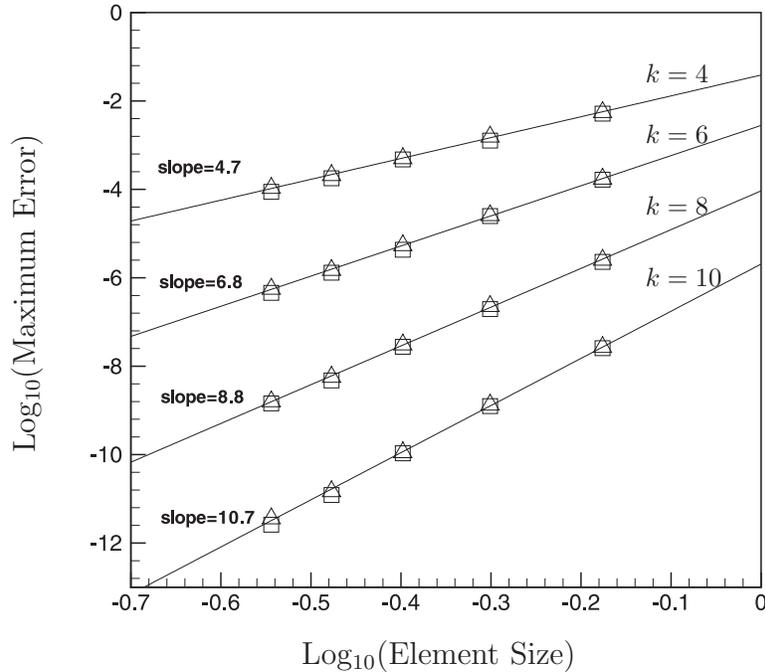


Figure 2.2: Maximum nodal error vs. element size for different orders k using two methods to solve the Helmholtz equation on the unit square: triangles are results of the IP method, and squares are results of the CG method reported in [119]. A best-fit line with associated slope is shown for the IP method results for each k .

definition of fluxes correspond to the interior penalty (IP) method introduced by Arnold [4]. The second term in eq. 2.10b is referred to as the penalty (stabilization) term and it is added to enforce the coercivity (stability) of the method. μ is called the penalty parameter and must be chosen large enough. While other choices for the numerical fluxes are possible [5], we prefer the IP method due to its simplicity, compact stencil size, stability and optimal rate of convergence. However, one drawback to this scheme is the lack of an explicit expression for the penalty parameter. This issue will be the focus of the fifth chapter, where an explicit lower bound for μ will be derived. For now, we assume μ is known for a given mesh and approximation order k .

We compared the accuracy of the IP method with the CG method in solving the Helmholtz equation with $f = (1 + 2\pi^2) \sin(\pi x) \sin(\pi y)$, corresponding to the (smooth)

exact solution $u = \sin(\pi x) \sin(\pi y)$. The domain was $[-1, 1] \times [-1, 1]$ and was partitioned into $N = 2M^2$ structured triangles. Fig. 2.2 shows how the maximum error depended on mesh size for orders $k = 4, 6, 8$ and 10 . It can be seen that the IP method yielded results almost identical to those obtained by the CG method using the same nodal sets, as reported by Warburton et al. [119]. We also calculated the convergence rates for different k , obtaining a convergence order of approximately $k + 1$, similar to that of the CG method reported in [119]. We conclude that the IP method is competitive with classical CG methods for elliptic problems.

2.2 Simplicity

DG methods offer simplicity in design and implementation of important strategies, namely parallelization, and non-conforming geometric (h) and functional (p) adaptivities. The simplicity results from the fact that in the DG setting no continuity constraint is enforced at element interfaces, a property that is not offered by the CG method.

2.2.1 Parallelization

In the DG method we develop in the following chapters, the degrees of freedom of an element are coupled only with those of its immediate neighbors. (Two elements are defined as immediate neighbors if they share an edge or a face in two or three space dimensions, respectively.) This means that in parallel implementations of DG algorithms, two processors communicate with each other only if they share an edge (face) in two (three) space dimensions. As a result, parallelization of the DG method is highly efficient and simple, which we now demonstrate with an example.

An unstructured mesh consisting of 72 triangles of order $k = 3$ is partitioned into four submeshes (Fig. 2.3), and each is assigned to a processor. Fig. 2.4a demonstrates the communication pattern of the DG method in evaluating the second order Laplacian term.

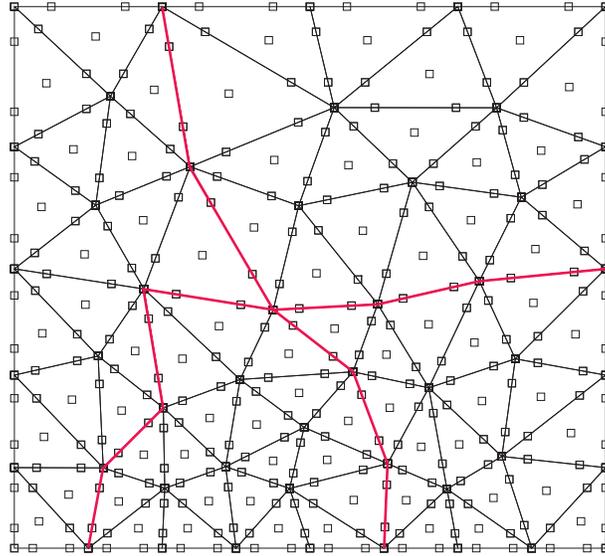
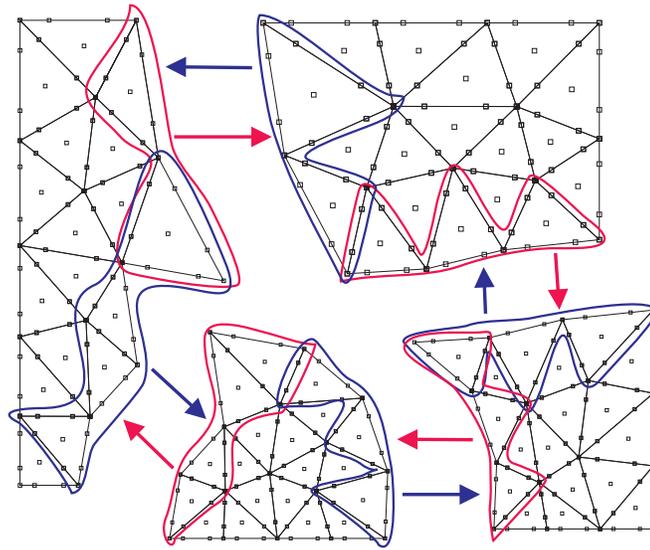


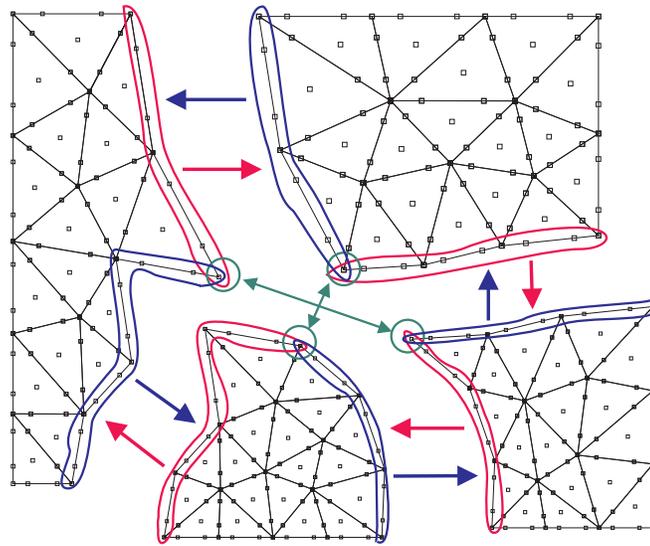
Figure 2.3: Partitioning of an unstructured mesh consisting of 72 triangles of order 3 into four sub-meshes using [99]. The squares signify the nodal points of order 3; the red polylines trace the partition boundaries.

Each processor communicates with two other processors, and sends only the information for those triangles that have an edge on the partition boundaries. Due to large start up costs, it is important to pack all the information destined to a given processor into a single message. Consequently, each processor sends only two separate messages in this example.

Fig. 2.4b shows the communication pattern for the direct stiffness summation procedure [36] required in evaluating the Laplacian term using the CG method. Here, each processor sends three separate messages. In the CG method, two processors exchange information if they share a vertex, or an edge (or a face in three space dimensions) on the partition boundary. Since in three space dimensions, the number of elements surrounding a given element can be large, the communication cost of a CG algorithm can be high. For example, assume a computational mesh is divided into several hexahedral regions and



(a), DG method



(b), CG method

Figure 2.4: Communication pattern on an unstructured mesh (Fig. 2.3), distributed among four processors. (a) In the DG method, each processor sends two separate messages containing information about *elements* inside the red and blue perimeters. (b) In the CG method, each processor sends three separate messages containing information about *nodes* inside the red, blue and green perimeters. Arrows are colored according to the perimeters showing the direction of the messages sent.

each region is assigned to a processor. Since a given region has typically 26 neighbors, based on the procedure of the preceding example, the direct stiffness step would require roughly 26 messages to be sent [36]. This is high, particularly when compared to the DG algorithm which requires roughly sending only 6 messages. Note that although in the CG method shorter messages are exchanged compared to the DG method, the communication cost of the CG method would still be higher. This is due to the fact that the start up time is much higher than the message transfer rate; a message can be as long as 100-1500 words before a two-fold increase in the communication cost will be realized [36].

There have been several efforts to devise efficient communication strategies for the CG method [36]. For instance, Tufo and Fischer [116] developed a general directional exchange algorithm suitable for unstructured meshes. This strategy has proved to be successful in simulations carried out on several thousands processors [36].

In a nutshell, while a simple communication algorithm can yield an efficient parallelization in a DG discretization, efficient communication strategies for a CG discretization, though available, are rather sophisticated and their implementations are cumbersome.

2.2.2 Non-conforming hp-adaptivity

Non-conforming h-adaptivity is a series of non-conforming local mesh refinement (or coarsening) in regions where the solution exhibits singularities or sharp gradients, to effectively achieve a desired level of accuracy. Non-conforming refinements result in meshes with hanging nodes (i.e., meshes with elements that are not face-to-face in three dimensions and are not edge-to-edge in two dimensions). An example of a non-conformingly refined mesh is shown in Fig. 2.5. Compared to conforming refinement, non-conforming refinement has the advantage of maintaining the original isotropy of the mesh.

P-adaptivity, on the other hand, is local order enrichment (or reduction) in regions

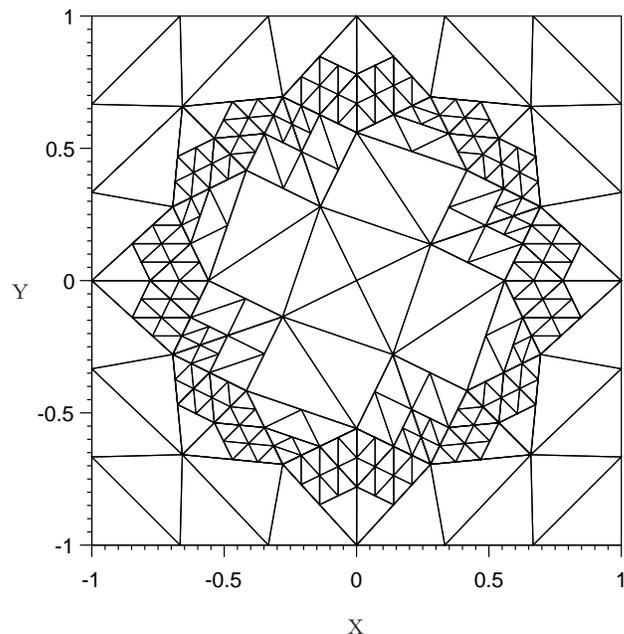


Figure 2.5: A non-conformingly refined mesh consisting of 1160 triangles. The original conforming mesh consisted of 72 triangles. DG methods handle such refinements easily.

with a smooth solution to efficiently achieve a given accuracy. hp-adaptivity is the simultaneous use of both h-adaptivity and p-adaptivity to achieve an exponential rate of convergence [34].

As the complexity of the problem grows, so does the need for adaptation. Considering the geometric complexity of MHVs, characterized by thin leaflets and a small hinge mechanism, non-conforming mesh refinement, easily supported in DG methods, is an effective tool to resolve large gradients of vorticity at leaflet edges and the micro structure of flow in the vicinity of the hinge mechanism.

2.3 Weak Imposition of Dirichlet Boundary Conditions

In DG methods, Dirichlet boundary conditions are imposed weakly, yielding boundary operators similar to the flux operators defined on the elemental interfaces. In the classical CG methods, BCs have traditionally been enforced strongly; however, extending the weak concept to the CG method is also possible.

Several studies have compared the performance of weakly imposed Dirichlet boundary conditions with the strongly imposed conditions, and concluded that the former are more advantageous than the latter. In direct numerical simulations and large eddy simulations of turbulent channel flows using high-order DG methods, Collis and Ramakrishnan [28, 95] showed that the weak imposition of the wall boundary conditions prevents the need to resolve the viscous sublayer. Barzilevs and Hughes [13] also compared weak and strong enforcement of the no-slip boundary conditions for a convection-diffusion equation and for the incompressible Navier-Stokes equations, but in the continuous Galerkin discretization. They found that weakly enforced conditions are effective and superior to the strongly enforced conditions. Moreover, similar to the results in the DG context [28, 95], they reported that weakly imposed conditions act like a wall function and thus avoid the need for high resolution close to the wall.

Based on the above evidence, we expect that imposing boundary conditions weakly in our DG scheme will have beneficial efficiency implications in blood flow simulations through mechanical heart valves.

2.4 Conservativity

Practitioners who numerically solve nonlinear conservation laws overwhelmingly prefer locally conservative methods. [25]. In climate and atmospheric chemistry applications,

both local and global conservations of quantities such as mass and total kinetic energy are important [35, 77]. There is also evidence in incompressible flow simulations (at high Reynolds number), using finite difference schemes, that conservative schemes lead to more accurate results than those of non-conservative forms (e.g., [87]).

Most DG methods are locally and globally conservative. The former implies elemental conservation and the latter implies conservation over the entire computational domain. Local conservation stems from the property that the test function can be set to value one over the element of interest and zero on the rest of the computational domain. Due to the discontinuous nature of test function spaces, this is possible in the DG methods. This is also true in the global sense. On the other hand, in most continuous Galerkin methods, this is neither possible element-wise due to the continuity constraint along the element boundaries nor over the entire domain due to the strong enforcement of the Dirichlet boundary conditions. CG methods are globally conservative only in the absence of Dirichlet boundary conditions [62]. The non-conforming Crouzeix-Raviart elements is among a few CG methods with the element-wise mass conservation property [30].

In an attempt to prove the conservation property of CG methods, Hughes et al. [62] introduced a modified formulation. While in the DG method the approximate solution alone satisfies the conservation law directly, this is not the case in this modified form of the CG method. An additional variable, a numerical flux on the Dirichlet part of the boundary (or element interfaces) calculated in a postprocessing stage, is required to be able to write a conservation statement.

2.5 Efficiency

2.5.1 Total Number of Unknowns

Due to the multiple values for the degrees of freedoms on elemental interfaces, DG methods yield larger number of unknowns compared to CG methods, and consequently they

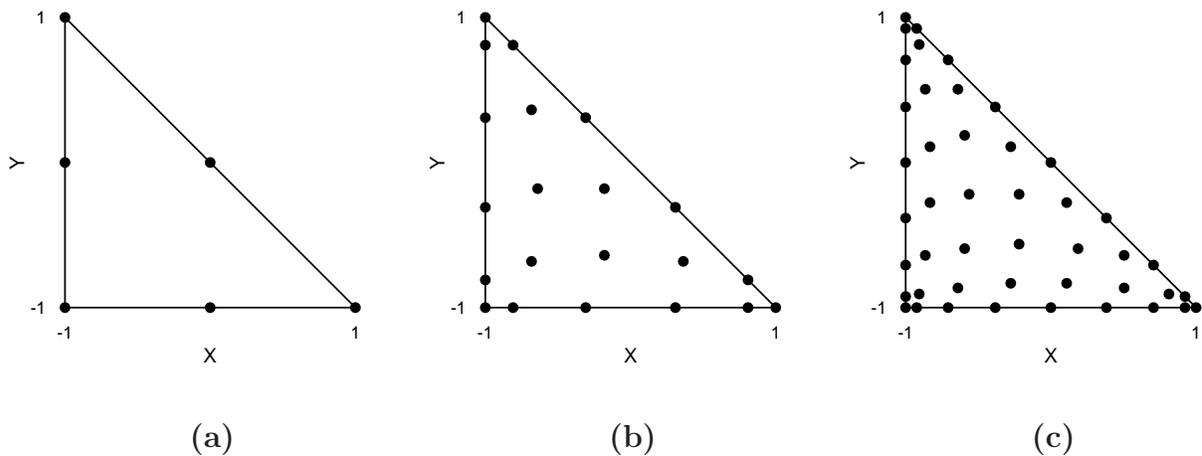


Figure 2.6: The standard triangle and its nodal sets for (a) $k = 2$, (b) $k = 5$, (c) $k = 8$ [57]; the corresponding percentages of the internal nodes are 0%, 29% and 40%.

may be considered less efficient than CG methods. This argument is only valid for low-order approximations where the ratio of the number of unknowns of the DG method (N_{DG}) to those of the CG method (N_{CG}), N_{DG}/N_{CG} , is large. While in low order approximations, the degrees of freedom are mostly located on the boundary of an element, in high order approximations, the number of internal degrees of freedoms increases and so does the percentage of internal degrees of freedoms. Fig. 2.6 shows the nodal set distributions for orders $k = 2, 5$ and 8 . The percentage of number of internal nodes has increased from 0% at $k = 2$ to 40% at $k = 8$. Therefore, as the percentage of the internal degrees of freedom increases, the percentage of the interfacial degrees of freedoms decreases, and N_{DG} approaches N_{CG} . To illustrate this fact, we measure N_{DG} and N_{CG} as a function of k on a structured mesh. The domain is a square (or cube in three dimensions), and is partitioned into $2n^2$ ($6n^3$) semi-structured triangular (tetrahedral) elements, where n is

the number of subdivisions in each direction. Then one may easily show that

$$N_{DG} = 2n^2(k+1)(k+2)/2 \quad d = 2, \quad (2.11a)$$

$$N_{DG} = 6n^3(k+1)(k+2)(k+3)/6 \quad d = 3, \quad (2.11b)$$

$$\begin{aligned} N_{CG} &= (n+1)(n+1) + (3n^2 + 2n)(k-1) \\ &\quad + 2n^2((k+1)(k+2)/2 - 3k) \quad d = 2, \quad (2.11c) \end{aligned}$$

$$\begin{aligned} N_{CG} &= (n+1)(n+1)(n+1) + \{(6n^2 + 3n)(n+1) + n^3\}(k-1) \\ &\quad + (6(n+1)n^2 + 6n^3)\{(k+1)(k+2)/2 - 3k\} \\ &\quad + 6(n^3)\{(k+1)(k+2)(k+3)/6 - 2(k^2 + 1)\} \quad d = 3. \quad (2.11d) \end{aligned}$$

Table 1.1 shows N_{DG}/N_{CG} (for $n \gg 1$) for $k = 1, \dots, 10$ and ∞ for both triangular and tetrahedral meshes. It is clear from the table that even for moderately high orders ($5 \leq k \leq 10$), N_{DG} is only a few times higher than N_{CG} ($N_{DG}/N_{CG} < 3$).

Furthermore, in high-order CG methods, the cost is not directly proportional to the total number of unknowns. Since the implementation of high-order CG methods is often based on elemental operations [36, 67], the cost scales as $E k^\alpha$ with E the total number of elements and $\alpha = 2d$ (or $d+1$ if tensor product forms are available). This cost is identical to the cost of a DG method.

In view of the above analysis and some evidence, such as that in chapter 3 that reveals that for some convection dominated flows we may require smaller resolution using a DG method than the CG method, we conclude that using (moderately) high-order DG methods can be computationally competitive with their continuous counterparts.

2.5.2 Block Diagonal Mass Matrix

In fully explicit or semi-explicit time integration schemes, it is required to invert a global mass matrix to directly solve a system or to precondition a system. Consequently, it is important to have a mass matrix which is diagonal or block diagonal.

Order	N_{DG}/N_{CG}	
	Triangular mesh	Tetrahedral mesh
1	6	23.93
2	3	7.49
3	2.22	4.44
4	1.87	3.28
5	1.70	2.69
6	1.56	2.33
7	1.47	2.1
8	1.41	1.93
9	1.36	1.81
10	1.32	1.72
∞	1	1

Table 2.1: The ratio of total number of unknowns for a DG method to those for a CG method, N_{DG}/N_{CG} , vs. approximation orders for both triangular and tetrahedral meshes.

Some CG methods, such as spectral element methods, yield diagonal mass matrices only if quadrilateral or hexahedral meshes are used. However, on triangular and tetrahedral elements, they lead to a sparse global mass matrix whose inversion is expensive. DG methods, on the other hand, lead to block diagonal mass matrices. Each block corresponds to an elemental mass matrix of size $(k + 1) \times (k + 1)$ which is easily invertible. In the case of straight-sided triangles or planar-faced tetrahedrons, only one elemental mass matrix corresponding to the standard element is inverted and the inverse of each block is recovered through a constant scaling.

There are some numerical results in the literature verifying the importance of a block diagonal mass matrix for explicit time integration schemes. In solving a hyperbolic equation on a sphere with a discontinuous solution, Giraldo [47] compared the performance of

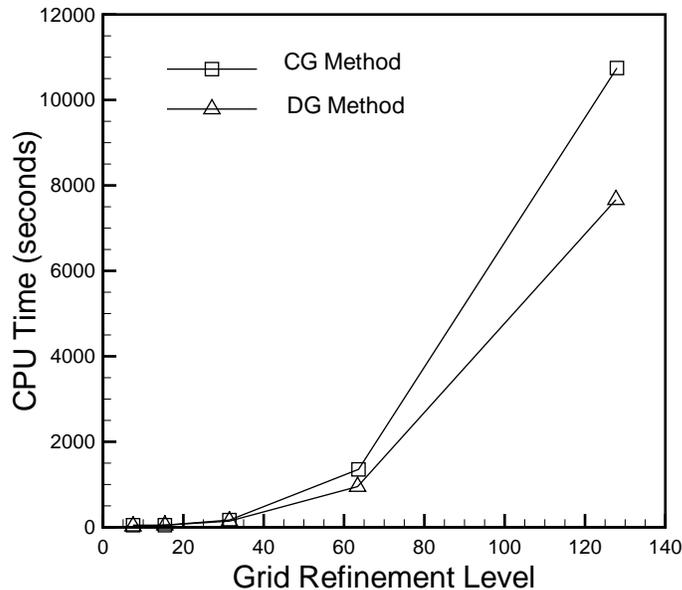


Figure 2.7: CPU time vs. refinement levels in solving a hyperbolic equation using CG and DG methods [47].

a triangular spectral element method with a triangular DG discretization. In the spectral element method, strong filtering was used for stabilization and a state-of-the-art solver (GMRES with a fast projection method [47]) was used to solve the global mass matrix. The results favored the DG method in both accuracy and computational time. The timing results are depicted in Fig. 2.7, demonstrating the higher speed of the DG solver over the CG solver, in particular as the problem size grows.

2.6 Drawbacks of DG methods

In the previous sections, we have described the benefits of DG methods, we now mention two drawbacks of these methods. First, as mentioned in subsection 2.5.1, for low approximation orders, DG methods yield larger number of unknowns than the CG methods, and possibly are less efficient than CG methods. Second, the solution strategies for the systems arising from DG discretizations are not as mature as those for the CG discretizations. In particular, the optimal overlapping elemental (multigrid) Schwarz pre-

conditioners for the spectral element discretization of the elliptic equations [80, 43] are not applicable to the DG setting. Also, the applicability of recently developed optimal two-level overlapping Schwarz preconditioners for the high-order CG methods on triangular and tetrahedral elements [101] to the DG discretization needs to be investigated.

Chapter 3

A High-Order DG Method for the NS Equations

This chapter presents our proposed high-order discontinuous Galerkin scheme for the solution of the unsteady incompressible Navier-Stokes equations and its verification on two-dimensional benchmarking problems.

3.1 Introduction

DG methods for pure elliptic problems and hyperbolic conservation laws have been extensively developed and analyzed during the past three decades (e.g, see the review articles [5] for elliptic problems and [27] for hyperbolic systems). Only recently, however have the DG methods been extended to the numerical solution of incompressible flows, including the Stokes and the incompressible Navier-Stokes equations (see articles [54, 102, 24] for the Stokes problem and [48, 23] for the Navier-Stokes equations). All of the above work has considered only the stationary Stokes or Navier-Stokes equations. Therefore, the objective of this paper is to propose an efficient DG scheme for the unsteady incompressible Navier-Stokes equations. Our approach is tailored to convection-dominated regimes encountered in transitional and turbulent flows. The approach is based on a semi-explicit

temporal discretization in which the convective term is treated explicitly and the Stokes operator is treated implicitly. It employs a high-order DG spatial discretization on triangular and tetrahedral elements in two and three space dimensions, respectively. To put our methodology in perspective, we review related work on the DG treatment of the convective and Stokes operators.

3.1.1 Review of DG Discretization of the Convective Operator

Several DG methods for the spatial discretization of the convective term have recently been proposed. Cockburn et al. [22] provided an *a priori* error estimate for the DG solution of the Oseen problem by treating the linear convective term with an upwinding scheme. For the nonlinear Navier-Stokes equations with nonoverlapping domain decompositions, Girault et al. [48] devised a stable method by discretizing the convective term in a skew-symmetric form. To assure that the DG formulation yields a locally conservative discretization, a property that is not offered by the above two methods, Cockburn et al. [23] proposed two strategies. In the first, they linearized the convective term and then used the results of [22] for the Oseen problem to prove the stability of the discrete solution. Through an iterative procedure they then recovered a locally conservative velocity field. In their second formulation, the pressure p was replaced with the Bernoulli pressure $p + \frac{1}{2}|\mathbf{u}|^2$; hence, local conservativity was attained. With this method, one can prove the boundedness of the approximate solution for the case of Dirichlet boundary conditions. For the case of outflow boundary conditions, however, this type of formulation leads to an unphysical solution at the outlet, as previously shown in the context of the continuous Galerkin approximation [60]. Thus, for engineering problems that are typically formulated on truncated domains with outflow boundary conditions, this method is not suitable.

We here propose a new strategy. We discretize the nonlinear term in the divergence form and use the local Lax-Friedrichs numerical fluxes to obtain stable results. Discretiz-

ing the nonlinear term in the divergence form immediately yields local conservativity, a property that the two previously proposed methods either do not offer (the method of Girault et al. [48]) or require an extra iteration procedure to attain (the first method of Cockburn et al. [23]). Unlike the second method of Cockburn et al. [23], our method applies for any boundary conditions, including Dirichlet, periodic, and outflow conditions.

3.1.2 Review of DG Discretization of the Stokes Operator

For the DG discretization of the Stokes operator, several studies have been reported in the literature, beginning with Hansbo and Larson [54]. For simplicial triangulations, they used the interior penalty (IP) method [4] for the viscous term and approximating polynomial degrees k and $k - 1$ for the velocity and the pressure, respectively ($P_k - P_{k-1}$, mixed-order formulation). Cockburn et al. [24] used the so-called local DG method for the viscous term [12, 26] and proved an inf-sup condition for equal approximating polynomial degree, k for the velocity and pressure ($P_k - P_k$, equal-order formulation), by adding a stabilization term to the discretized divergence-free constraint. Schötzau et al. [103] similarly employed the equal-order formulation but with an IP discretization of the Laplacian. In our methodology, we consider the IP method for the viscous term and both the $P_k - P_{k-1}$ and $P_k - P_k$ formulations. We prefer the IP method over the local DG method for its simplicity and its compact stencil size.

Employing a semi-explicit temporal discretization and the DG spatial discretization leads to an algebraic Stokes system to solve at each time step. For this system, we propose a new class of second-order approximate splitting methods. Applying algebraic splitting procedures introduced earlier in the context of the finite volume, finite element or spectral element methods (e.g., [93, 56, 29], respectively) yields a Helmholtz system for the velocity and a consistent Poisson equation with an extended stencil size for the pressure variable to be solved at each time step. In the DG setting, however, we are able to replace this pressure operator with an equivalent operator that is simpler and

computationally more efficient. This operator offers a compact stencil size and arises from the IP discretization of the Poisson equation with appropriate boundary conditions.

We note that the application of algebraic splitting for solving the Stokes system on triangular or tetrahedral meshes is more advantageous in the DG setting than in the continuous Galerkin method. In the DG setting, the pressure operator has compact stencil-size, and the mass matrix is block diagonal. The block diagonal structure of the mass matrix permits a simple and efficient preconditioner for the iterative solution of the Helmholtz system of the velocity as explained in Section 3.2. None of these properties exists in the continuous counterpart. Furthermore, our algebraic splitting approach is superior to the Chorin-Temam projection scheme (differential splittings) [21, 113] in the sense that our scheme avoids unphysical boundary conditions for the pressure equation inherent in the differential splittings.

Below, we describe details of our solution procedure. We then present some implementation details in Section 3.3. These are followed in Section 3.4 by numerical experiments to demonstrate the temporal and spatial accuracy of the method. In Section 3.5, the developed solver is used to simulate flow through a two-dimensional mechanical heart valve.

3.2 Navier-Stokes Discretization

We seek the numerical solution of the unsteady incompressible Navier-Stokes equations

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \frac{1}{Re} \nabla^2 \mathbf{u} - \nabla p + \mathbf{f} \quad \text{in } \Omega \times [0, T], \quad (3.1a)$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega \times [0, T], \quad (3.1b)$$

$$\mathbf{u}(t = 0) = \mathbf{u}_0 \quad \text{in } \Omega, \quad (3.1c)$$

$$\mathbf{u} = \mathbf{g}_D \quad \text{on } \partial\Omega_D, \quad (3.1d)$$

$$\frac{1}{Re} \frac{\partial \mathbf{u}}{\partial n} - p \mathbf{n} = 0 \quad \text{on } \partial\Omega_N, \quad (3.1e)$$

$$s(\mathbf{x}) = s(\mathbf{x}') \quad \mathbf{x}, \mathbf{x}' \in \partial\Omega_P, \quad (3.1f)$$

where \mathbf{u} , p and t are the non-dimensionalized velocity vector, pressure and time, respectively, and \mathbf{f} is a known body force. The Reynolds number is $Re = (UL)/\nu$, with U a characteristic velocity, L a length scale, and ν the kinematic viscosity. Equation 3.1c represents an appropriate initial condition, and eqs. 3.1d, e, and f represent Dirichlet, outflow, and periodic boundary conditions (BCs), respectively. Note that $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N \cup \partial\Omega_P$. s represents any component of the velocity vector or the pressure, and \mathbf{x} and \mathbf{x}' are two periodic points. Ω is a polygonal domain of dimension $d = 2$, or 3, and T is the total integration time.

The numerical solution of the above system consists of two parts: temporal discretization and spatial discretization. For temporal discretization, we use a semi-explicit scheme, in which the nonlinear term is treated explicitly and the Stokes operator is treated implicitly. We use a third-order backward differentiation (BD3) scheme for the unsteady term and a third-order extrapolation (EX3) for the nonlinear term, as proposed by Karniadakis et al. [68]. Let the total integration time T be divided into uniform time steps

of size Δt . Then the semi-discretized forms of eqs. 3.1a and 3.1b at time step n become

$$\begin{aligned} \left(-\frac{1}{Re}\nabla^2 + \frac{\beta_0}{\Delta t}\right)\mathbf{u}^{n+1} + \nabla p^{n+1} &= \left(\frac{\beta_1}{\Delta t}\mathbf{u}_1^n + \frac{\beta_2}{\Delta t}\mathbf{u}_2^{n-1} + \frac{\beta_3}{\Delta t}\mathbf{u}_2^{n-2}\right) \\ &\quad - (\gamma_1\mathbf{c}^n + \gamma_2\mathbf{c}^{n-1} + \gamma_3\mathbf{c}^{n-2}) + \mathbf{f}^{n+1} \quad \text{in } \Omega, \end{aligned} \quad (3.2a)$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \quad \text{in } \Omega. \quad (3.2b)$$

Here \mathbf{c} represents the nonlinear term; $\beta_0 = 11/6$, $\beta_1 = 3$, $\beta_2 = -3/2$, and $\beta_3 = 1/3$ are coefficients associated with BD3; and $\gamma_1 = 3$, $\gamma_2 = -3$, and $\gamma_3 = 1$ are coefficients associated with EX3. For ease of notation, we will drop the superscripts referring to the time steps and absorb the right-hand side of eq. 3.2a into \mathbf{f} .

Due to the explicit treatment of the convective term, time steps are limited by a CFL condition. We choose Δt based on the estimate

$$\Delta t \approx \mathcal{O}\left(\frac{\mathcal{L}}{\mathcal{U}k^2}\right), \quad (3.3)$$

reported in [67] for an advection model problem. Here \mathcal{L} is an integral length scale (typically the mesh element size) and \mathcal{U} is a characteristic velocity.

Below, we introduce some notation and approximate spaces, then describe the spatial discretization including the DG treatment of the nonlinear and Stokes operators.

3.2.1 Preliminaries

Let Γ_I denote the collection of all interior faces ¹. Then $\Gamma_{IDP} = \Gamma_I \cup \partial\Omega_D \cup \partial\Omega_P$, $\Gamma_{INP} = \Gamma_I \cup \partial\Omega_N \cup \partial\Omega_P$, and $\Gamma_{IDNP} = \Gamma_I \cup \partial\Omega_D \cup \partial\Omega_N \cup \partial\Omega_P$. On a face $e \in \Gamma_I$ shared with two elements K^+ and K^- , we permanently associate e with a unit normal vector \mathbf{n}_e directed from K^+ to K^- , and define the jump and average operators of a function ϕ by

$$[[\phi]] := (\phi|_{K^+})|_e - (\phi|_{K^-})|_e \quad \{\phi\} = \frac{1}{2}(\phi|_{K^+})|_e + \frac{1}{2}(\phi|_{K^-})|_e.$$

¹The terms “face” and “surface integral” denote edge and line integral in two space dimensions as well.

For $e \in \partial\Omega_P$, we use the same definitions except that if K^+ contains e , K^- is an element containing the periodic face of e . On a Dirichlet or outflow face e , \mathbf{n}_e is the unit normal vector \mathbf{n} outward to Ω , and the jump and average of the operator ϕ coincide with the trace of ϕ .

The discontinuous approximate spaces we use are

$$\mathcal{V}_k := \{v \in L^2(\Omega) | v|_K \in P_k(K), \forall K \in \mathcal{T}_h\} \quad (3.4)$$

and its vector version \mathcal{V}_k^d . $P_k(K)$ is the set of polynomials of total degree at most k on K , $k \geq 1$, with K being a simplicial element of the geometrically conforming triangulation \mathcal{T}_h of the domain Ω . While our methodology applies to a geometrically and functionally nonconforming approximation, for simplicity, we consider only conforming triangulations and uniform polynomial degrees over all elements.

3.2.2 Nonlinear Treatment

Using the divergence free constraint $\nabla \cdot \mathbf{u} = 0$, we write the nonlinear term in the divergence form

$$\mathbf{u} \cdot \nabla \mathbf{u} = \mathbf{u} \cdot \nabla \mathbf{u} + \mathbf{u} \nabla \cdot \mathbf{u} \equiv \nabla \cdot (\mathbf{u} \otimes \mathbf{u}),$$

where $\mathbf{u} \otimes \mathbf{v} := u_i v_j$, $i, j = 1, \dots, d$. We can now use ideas for the DG discretization of the nonlinear term previously developed in the context of hyperbolic conservation laws [27]. Let \mathbf{u} be approximated by $\mathbf{u}_h \in \mathcal{V}_k^d$, and for notational simplicity we denote $\mathbf{u}_h \otimes \mathbf{u}_h$ by $\underline{\underline{g}}$ hereafter. Multiplying the nonlinear term by a test function $\mathbf{v}_h \in \mathcal{V}_k^d$, integrating over the whole domain Ω , and carrying out integration by parts, we obtain

$$\int_{\Omega} \mathbf{v}_h \cdot (\nabla \cdot \underline{\underline{g}}) d\mathbf{x} = - \sum_K \int_K (\underline{\underline{g}} \cdot \nabla) \cdot \mathbf{v}_h d\mathbf{x} + \sum_{\Gamma_{IDNP}} \int_e \mathbf{n}_e \cdot \llbracket \underline{\underline{g}} \cdot \mathbf{v}_h \rrbracket ds.$$

To complete the discretization, we replace the integrand $\mathbf{n}_e \cdot \llbracket \underline{\underline{g}} \cdot \mathbf{v}_h \rrbracket$ in the surface integral with the local Lax-Friedrich fluxes $\overbrace{\mathbf{n}_e \cdot \llbracket \underline{\underline{g}} \cdot \mathbf{v}_h \rrbracket}$

$$\int_{\Omega} \mathbf{v}_h \cdot (\nabla \cdot \underline{\underline{g}}) d\mathbf{x} = - \sum_K \int_K (\underline{\underline{g}} \cdot \nabla) \cdot \mathbf{v}_h d\mathbf{x} + \sum_{\Gamma_{IDNP}} \int_e \overbrace{\mathbf{n}_e \cdot \llbracket \underline{\underline{g}} \cdot \mathbf{v}_h \rrbracket} ds, \quad (3.5)$$

where

$$\overbrace{\mathbf{n}_e \cdot \llbracket \underline{\underline{g}} \cdot \mathbf{v}_h \rrbracket} = \mathbf{n}_e \cdot \{\underline{\underline{g}}\} \cdot \llbracket \mathbf{v}_h \rrbracket + \frac{1}{2} \Lambda_{K,e} \llbracket \mathbf{u}_h \rrbracket \cdot \llbracket \mathbf{v}_h \rrbracket. \quad (3.6)$$

In eq. 3.6, and only in eq. 3.6, and for $e \in \Omega_D$, the operators $\{\}$ and $\llbracket \rrbracket$ have slightly different interpretations than those previously defined. Specifically, for $e \in \Omega_D$, $\{\underline{\underline{g}}\} = \frac{1}{2}((\mathbf{u}_h \otimes \mathbf{u}_h)|_e + (\mathbf{g}_D \otimes \mathbf{g}_D)|_e)$ and $\llbracket \mathbf{u}_h \rrbracket = (\mathbf{u}_h|_e - \mathbf{g}_D|_e)$. To define $\Lambda_{K,e}$, let λ^+ and λ^- be the largest eigenvalue (in absolute value) of the Jacobians $(\partial/\partial \mathbf{u})(\underline{\underline{g}} \cdot \mathbf{n}_e)|_{\bar{\mathbf{u}}_{K^+}}$ and $(\partial/\partial \mathbf{u})(\underline{\underline{g}} \cdot \mathbf{n}_e)|_{\bar{\mathbf{u}}_{K^-}}$, respectively, with $\bar{\mathbf{u}}_{K^+}$ and $\bar{\mathbf{u}}_{K^-}$ being the mean values of \mathbf{u}_h over the entire element K^+ and K^- , respectively. Then,

$$\Lambda_{K,e} = \max(\lambda^+, \lambda^-). \quad (3.7)$$

For $e \in \Omega_P$ and $e \in \Omega_D$, $\Lambda_{K,e}$ is defined similarly. Specifically, for $e \in \Omega_P$ and K^+ containing e , K^- contains the periodic face of e . For $e \in \Omega_D$ and $\bar{\mathbf{u}}_{K^+}$ being the mean of \mathbf{u}_h on K^+ containing e , $\bar{\mathbf{u}}_{K^-} = \mathbf{g}_D$. For $e \in \Omega_N$, $\Lambda_{K,e} = 0$.

Remark 1. The terms $(\underline{\underline{g}} \cdot \nabla) \cdot \mathbf{v}_h$ and $\mathbf{n}_e \cdot \{\underline{\underline{g}}\} \cdot \llbracket \mathbf{v}_h \rrbracket$ in eqs. 3.5 and 3.6 are evaluated in index notation as:

$$(\underline{\underline{g}} \cdot \nabla) \cdot \mathbf{v}_h := g_{ij} \frac{\partial v_{hi}}{\partial x_j} \quad i, j = 1, \dots, d, \quad (3.8a)$$

$$\mathbf{n}_e \cdot \{\underline{\underline{g}}\} \cdot \llbracket \mathbf{v}_h \rrbracket := n_{ej} \{g_{ij}\} \llbracket v_{hi} \rrbracket \quad i, j = 1, \dots, d, \quad (3.8b)$$

where repeated indices imply summation.

Remark 2. This choice of the numerical fluxes leads to a compact stencil size. As shown in Fig. 3.1a, the degrees of freedom (DOF) of a reference element (black triangle) couple only with those of its immediate neighbors (dark grey triangles).

3.2.3 Stokes Discretization

To set the stage for describing our solution procedure for the unsteady Stokes system 3.2, we first review two DG discretizations of the Poisson problem: the IP method of Arnold

[4] and the method of Bassi and Rebay [12], further developed in [26]. The latter method is referred to as the local DG method.

We seek the IP and the local DG formulations of the Poisson equation with Dirichlet, Neumann, and periodic boundary conditions:

$$-\Delta u = f \quad \text{in } \Omega, \quad (3.9a)$$

$$u = g_D \quad \text{on } \partial\Omega_D, \quad (3.9b)$$

$$\nabla u \cdot \mathbf{n} = g_N, \quad \text{on } \partial\Omega_N, \quad (3.9c)$$

$$u(\mathbf{x}) = u(\mathbf{x}') \quad \mathbf{x}, \mathbf{x}' \in \partial\Omega_P. \quad (3.9d)$$

IP Formulation of the Poisson Equation

In the IP formulation, the discontinuous approximation to the exact solution u , u_h , is a member of the finite element space \mathcal{V}_k . The approximate solution is defined by requiring that

$$a_h(u_h, v_h) = f_h(v_h) \quad \forall v_h \in \mathcal{V}_k,$$

where

$$a_h(u, v) = \sum_K \int_K \nabla u \cdot \nabla v d\mathbf{x} - \sum_{\Gamma_{IDP}} \int_e \left[\mathbf{n}_e \cdot \{\nabla u\} [[v]] + \mathbf{n}_e \cdot \{\nabla v\} [[u]] \right] ds + \sum_{\Gamma_{IDP}} \int_e \mu [[u]] [[v]] ds, \quad (3.10a)$$

$$f_h(v) = \int_{\Omega} f v d\mathbf{x} + \int_{\partial\Omega_N} g_N v ds + \int_{\partial\Omega_D} g_D (\mu v - \nabla v \cdot \mathbf{n}) ds. \quad (3.10b)$$

The last term in eq. 3.10a is called the penalty term. It is added to enforce the coercivity of the bilinear form, which requires the choice of a sufficiently large value for the penalty parameter μ . The last integral in eq. 3.10b is due to the weak imposition of the Dirichlet boundary conditions.

The minimum acceptable value for μ depends on the triangulation and the approximating polynomial degree. Although an explicit expression for μ is not known for a

general mesh topology, we will derive an expression for the case of simplicial elements in chapter 5 [104]. Specifically, here we have used

$$\mu = \frac{(k+1)(k+d)}{d} \max_K \left(\frac{S_K}{V_K} \right), \quad (3.11)$$

where S_K and V_K represent the surface area (perimeter in two dimensions) and volume (area in two dimensions) of the element K , respectively. This is a slightly simplified version of eq. 5.8. Note that instead of the global expression in eq. 3.11, the local penalty parameter derived in eq. 5.8 can also be used.

Local DG Formulation of the Poisson Equation

In the local DG method, the Laplacian is first written as two first-order operators by introducing the auxiliary variable $\boldsymbol{\sigma}$:

$$\nabla u = \boldsymbol{\sigma}, \quad (3.12a)$$

$$-\nabla \cdot \boldsymbol{\sigma} = f. \quad (3.12b)$$

In previous work, the approximations to u and $\boldsymbol{\sigma}$, u_h and $\boldsymbol{\sigma}_h$, have belonged to \mathcal{V}_k and \mathcal{V}_k^d , that is, spaces with equal polynomial degrees (equal-order method). In addition to equal-order polynomial degree spaces, here we allow $u_h \in \mathcal{V}_k$ and $\boldsymbol{\sigma}_h \in \mathcal{V}_{k+1}^d$, spaces with mixed polynomial degrees (mixed-order method). This approach is similar to our DG method for the Stokes operator, where both equal- and mixed-order methods are allowed.

The approximate solutions u_h and $\boldsymbol{\sigma}_h$ are then defined by requiring that

$$d_h(\boldsymbol{\tau}_h, u_h) = b_h(\boldsymbol{\sigma}_h, \boldsymbol{\tau}_h) + g_h(\boldsymbol{\tau}_h) \quad \forall \boldsymbol{\tau}_h \in \mathcal{V}_k^d / \mathcal{V}_{k+1}^d, \quad (3.13a)$$

$$-d_h(\boldsymbol{\sigma}_h, v_h) + e_h(u_h, v_h) = f_h(v_h) \quad \forall v_h \in \mathcal{V}_k, \quad (3.13b)$$

where

$$d_h(\boldsymbol{\tau}, u) = - \sum_K \int_K u \nabla \cdot \boldsymbol{\tau} d\mathbf{x} + \sum_{\Gamma_{INP}} \int_e \{u\} \llbracket \boldsymbol{\tau} \rrbracket \cdot \mathbf{n}_e ds, \quad (3.14a)$$

$$b_h(\boldsymbol{\sigma}, \boldsymbol{\tau}) = \int_{\Omega} \boldsymbol{\sigma} \cdot \boldsymbol{\tau} d\mathbf{x}, \quad (3.14b)$$

$$e_h(u, v) = \sum_{\Gamma_{IDP}} \int_e \alpha \llbracket u \rrbracket \llbracket v \rrbracket ds, \quad (3.14c)$$

$$g_h(\boldsymbol{\tau}) = - \int_{\partial\Omega_D} g_D \boldsymbol{\tau} \cdot \mathbf{n} ds, \quad (3.14d)$$

$$f_h(v) = \int_{\Omega} f v_h d\mathbf{x} + \int_{\partial\Omega_N} g_N v ds + \int_{\partial\Omega_D} \alpha g_D v ds. \quad (3.14e)$$

Here $e_h(u, v)$ is the penalty (stabilization) term and $\alpha = \eta/h_e$, with η any positive number, and $h_e \equiv \text{diam}(e)$. Since $\eta \ll 1$ and $\eta \gg 1$ yield ill-conditioned matrices, as shown by Castillo [20], moderate values of η ($\approx \mathcal{O}(1)$) should be chosen in practice. Note that the original formulation of Bassi and Rebay [12] lacks stabilization, i.e., $e_h(u, v) = 0$.

Remark 3. In the weak imposition of periodic BCs, for the IP and the local DG formulation, ∇u is also assumed to be periodic, so as to yield a symmetric discretization.

Remark 4. Applying the nodal high-order basis (described in Section 3.3) in eqs. 3.13a and 3.13b yields

$$\begin{bmatrix} \mathbf{B} & \mathbf{D}^T \\ \mathbf{D} & \mathcal{E} \end{bmatrix} \begin{bmatrix} \underline{\boldsymbol{\sigma}}_h \\ \underline{\mathbf{u}}_h \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{z}}_1 \\ \underline{\mathbf{z}}_2 \end{bmatrix}, \quad (3.15)$$

where $\underline{\mathbf{z}}_1$ and $\underline{\mathbf{z}}_2$ represent the given right-hand side and boundary data. Here, a bold matrix consists of d identical blocks. If n_u and $n_{\boldsymbol{\sigma}}$ are the number of degrees of freedom of a component of u_h and $\boldsymbol{\sigma}_h$, respectively, the matrix \mathbf{B} consists of d diagonal blocks of size $n_{\boldsymbol{\sigma}} \times n_{\boldsymbol{\sigma}}$ each, corresponding to the mass term, and the matrix \mathbf{D} consists of d blocks of size $n_u \times n_{\boldsymbol{\sigma}}$ each, corresponding to the divergence term. The matrix \mathcal{E} is $n_u \times n_u$, corresponding to the penalty term. After elimination of $\boldsymbol{\sigma}_h$ from the above system, the matrix equation for finding $\underline{\mathbf{u}}_h$ becomes

$$(-\mathbf{D} \mathbf{B}^{-1} \mathbf{D}^T + \mathcal{E}) \underline{\mathbf{u}}_h = \underline{\mathbf{z}}, \quad (3.16)$$

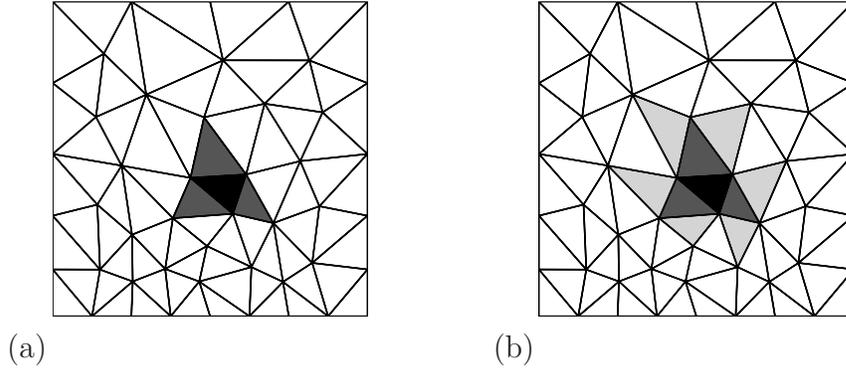


Figure 3.1: (a) Compact stencil of the IP discretization of the Laplacian, and the DG treatment of the nonlinear term discussed in Section 3.2 on an unstructured triangular mesh (DOF of the black triangle couple with those of dark grey triangles); (b) the extended stencil size of the local DG discretization of the Laplacian on the same mesh (DOF of the black triangle couple with those of dark grey as well as light grey triangles). The reference triangle is shown in black, the immediate neighbors of the reference triangle are denoted in dark grey, and the second layer of the neighbors is in light grey.

where \underline{z} corresponds to the given right-hand side and boundary data.

Remark 5. While the IP method offers a compact stencil size (Fig. 3.1a), the local DG method yields an extended stencil size (Fig. 3.1b). The larger stencil size leads to higher computations and communications per global stiffness matrix construction and global stiffness matrix-vector product.

3.2.4 DG Formulation of the Unsteady Stokes Operator

Following [54, 103], we first present the DG discretization of the system 3.2. The discontinuous approximations \mathbf{u}_h and p_h are defined by requiring that

$$A_h(\mathbf{u}_h, \mathbf{v}_h) + B_h(\mathbf{u}_h, \mathbf{v}_h) + D_h(\mathbf{v}_h, p_h) = F_h(\mathbf{v}_h) \quad \forall \mathbf{v}_h \in \mathcal{V}_k^d, \quad (3.17a)$$

$$D_h(\mathbf{u}_h, q_h) = G_h(q_h) \quad \forall q_h \in \mathcal{V}_k / \mathcal{V}_{k-1}, \quad (3.17b)$$

where

$$A_h(\mathbf{u}, \mathbf{v}) = \sum_K \int_K \frac{1}{Re} \nabla \mathbf{u} : \nabla \mathbf{v} dx - \sum_{\Gamma_{IDP}} \int_e \frac{1}{Re} \left[\mathbf{n}_e \cdot \{\nabla \mathbf{u}\} \cdot \llbracket \mathbf{v} \rrbracket + \mathbf{n}_e \cdot \{\nabla \mathbf{v}\} \cdot \llbracket \mathbf{u} \rrbracket \right] ds + \sum_{\Gamma_{IDP}} \int_e \frac{\mu}{Re} \llbracket \mathbf{u} \rrbracket \cdot \llbracket \mathbf{v} \rrbracket ds, \quad (3.18a)$$

$$B_h(\mathbf{u}, \mathbf{v}) = \sum_K \int_K \frac{\beta_0}{\Delta t} \mathbf{u} \cdot \mathbf{v} dx, \quad (3.18b)$$

$$D_h(\mathbf{v}, q) = - \sum_K \int_K q \nabla \cdot \mathbf{v} dx + \sum_{\Gamma_{IDP}} \int_e \{q\} \llbracket \mathbf{v} \rrbracket \cdot \mathbf{n}_e ds, \quad (3.18c)$$

$$F_h(\mathbf{v}) = \int_{\Omega} \mathbf{f} \cdot \mathbf{v} dx + \int_{\Omega_D} \frac{1}{Re} \left[-\mathbf{g}_D \cdot \nabla \mathbf{v} \cdot \mathbf{n} + \mu \mathbf{g}_D \cdot \mathbf{v} \right] ds, \quad (3.18d)$$

$$G_h(q) = \int_{\Omega_D} q \mathbf{g}_D \cdot \mathbf{n} ds. \quad (3.18e)$$

For the viscous term, the IP method is used because of its simplicity and compact stencil size. The discretization of the two operators ∇p and $\nabla \cdot \mathbf{u}$ are similar to that of ∇u and $\nabla \cdot \boldsymbol{\sigma}$ in eqs. 3.13a and 3.13b. The only difference is that the roles of the Dirichlet and outflow boundary conditions are switched in the surface integrals of $D_h(\mathbf{v}, q)$ and $d_h(u, \boldsymbol{\tau})$.

Remark 6. Analogous to Remark 1, the terms $\nabla \mathbf{u} : \nabla \mathbf{v}$ and $\mathbf{n}_e \cdot \{\nabla \mathbf{u}\} \cdot \llbracket \mathbf{v} \rrbracket$ in eq. 3.18a are evaluated as:

$$\nabla \mathbf{u} : \nabla \mathbf{v} := \frac{\partial u_i}{\partial x_j} \frac{\partial v_i}{\partial x_j} \quad i, j = 1, \dots, d, \quad (3.19a)$$

$$\mathbf{n}_e \cdot \{\nabla \mathbf{u}\} \cdot \llbracket \mathbf{v} \rrbracket := n_{ej} \left\{ \frac{\partial u_i}{\partial x_j} \right\} \llbracket v_i \rrbracket \quad i, j = 1, \dots, d. \quad (3.19b)$$

Now, let the matrix form of the discretized Stokes system be

$$\begin{bmatrix} \mathbf{H} & \mathbf{D}^T \\ \mathbf{D} & 0 \end{bmatrix} \begin{bmatrix} \underline{\mathbf{u}}^{n+1} \\ \underline{p}^{n+1} \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{f}}^{n+1} \\ \underline{g}^{n+1} \end{bmatrix}, \quad (3.20)$$

where $\mathbf{H} = (1/Re)\mathbf{A} + (\beta_0/\Delta t)\mathbf{B}$ with \mathbf{A} and \mathbf{B} denoting the Laplacian and block diagonal mass matrices, respectively. The descriptions of the matrices are similar to those in eq. 3.15. $\underline{\mathbf{f}}^{n+1}$ and \underline{g}^{n+1} correspond to the given right-hand-side and boundary data.

Applying an LU factorization procedure to the above system and writing the system for the pressure increment variable $\underline{p}^{n+1} - \underline{p}^n$, instead of \underline{p}^{n+1} , yields

$$\begin{bmatrix} \mathbf{H} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{H}^{-1}\mathbf{D}^T \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{H}^{-1}\mathbf{D}^T \\ \mathbf{0} & I \end{bmatrix} \begin{bmatrix} \underline{\mathbf{u}}^{n+1} \\ \underline{p}^{n+1} - \underline{p}^n \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{f}}^{n+1} \\ \underline{g}^{n+1} \end{bmatrix} + \begin{bmatrix} -\mathbf{D}^T \underline{p}^n \\ 0 \end{bmatrix}.$$

By introducing the auxiliary vector $\underline{\tilde{\mathbf{u}}}^{n+1}$, the solution procedure for the above system can be written in three steps as follows:

1. $\mathbf{H}\underline{\tilde{\mathbf{u}}}^{n+1} = \underline{\mathbf{f}}^{n+1} - \mathbf{D}^T \underline{p}^n,$
2. $(-\mathbf{D}\mathbf{H}^{-1}\mathbf{D}^T) (\underline{p}^{n+1} - \underline{p}^n) = -\mathbf{D}\underline{\tilde{\mathbf{u}}}^{n+1} + \underline{g}^{n+1},$ (3.21)
3. $\underline{\mathbf{u}}^{n+1} = \underline{\tilde{\mathbf{u}}}^{n+1} - \mathbf{H}^{-1}\mathbf{D}^T (\underline{p}^{n+1} - \underline{p}^n).$

The first and second steps involve linear system solves, and the preferred approach is an iterative method. For the pressure increment solution, step 2, each iteration requires extra inner iterations associated with the inversion of the matrix \mathbf{H} . The inner iterations can be avoided by replacing \mathbf{H}^{-1} with the computationally more efficient matrix $\mathbf{H}_I = (\Delta t/\beta_0)\mathbf{B}^{-1}$, where \mathbf{B} is block diagonal, and easily invertible. As shown in [56], this choice leads to a second-order accurate approximation in time. Replacing \mathbf{H}^{-1} with \mathbf{H}_I in 3.21, we obtain the approximate split solution procedure

1. $\mathbf{H}\underline{\tilde{\mathbf{u}}}^{n+1} = \underline{\mathbf{f}}^{n+1} - \mathbf{D}^T \underline{\hat{p}}^n,$
2. $(-\mathbf{D}\mathbf{H}_I\mathbf{D}^T) (\underline{\hat{p}}^{n+1} - \underline{\hat{p}}^n) = -\mathbf{D}\underline{\tilde{\mathbf{u}}}^{n+1} + \underline{g}^{n+1},$ (3.22)
3. $\underline{\hat{\mathbf{u}}}^{n+1} = \underline{\tilde{\mathbf{u}}}^{n+1} - \mathbf{H}_I\mathbf{D}^T (\underline{\hat{p}}^{n+1} - \underline{\hat{p}}^n)$

where $\underline{\hat{\mathbf{u}}}$ and $\underline{\hat{p}}$ are the approximations to $\underline{\mathbf{u}}$ and \underline{p} , respectively.

The first and second steps are solved iteratively by using the conjugate gradient method. Since $Re/\Delta t \gg 1$, the Helmholtz solves are effectively preconditioned by the block diagonal mass matrix, leading to a small number of iterations typically of $\mathcal{O}(1)$. On the other hand, the pressure solve requires a more sophisticated and more expensive preconditioner, and thus it is the dominant computation in terms of cost. Moreover,

the cost of a single iteration without preconditioning is also higher for the pressure case, since the pressure operator $(-\mathbf{D}\mathbf{H}_I\mathbf{D}^T)$ has an extended stencil similar to Fig. 3.1b. To alleviate this cost, one may attempt to reduce the pressure stencil size. In the DG setting, this strategy appears plausible.

Careful inspection of the pressure operator $(-\mathbf{D}\mathbf{H}_I\mathbf{D}^T)$ reveals that this operator is identical (to within a multiplicative constant) to the operator in eq. 3.16 with $\mathcal{E} = 0$, except that the roles of Dirichlet and Neumann BCs are switched in \mathbf{D} and \mathcal{D} (compare the definition of the divergence in eqs. 3.18c and 3.14a). In other word, $(-\mathbf{D}\mathbf{H}_I\mathbf{D}^T)$ results from the application of the local DG method (with zero stabilization) to a Laplacian with the following BCs:

$$\nabla v \cdot \mathbf{n} = 0 \quad \text{on } \partial\Omega_D \quad (3.23a)$$

$$v = 0 \quad \text{on } \partial\Omega_N \quad (3.23b)$$

$$v(\mathbf{x}) = v(\mathbf{x}') \quad \mathbf{x}, \mathbf{x}' \in \partial\Omega_P. \quad (3.23c)$$

Having realized this, we propose to replace the pressure operator $(-\mathbf{D}\mathbf{H}_I\mathbf{D}^T)$ with the operator arising from the IP discretization of the (negative) Laplacian with the above BCs:

$$\begin{aligned} \mathcal{A}_h(\mathbf{u}, \mathbf{v}) &= \sum_K \int_K \nabla \mathbf{u} : \nabla \mathbf{v} dx - \sum_{\Gamma_{INP}} \int_e \left[\mathbf{n}_e \cdot \{\nabla \mathbf{u}\} \cdot \llbracket \mathbf{v} \rrbracket + \mathbf{n}_e \cdot \{\nabla \mathbf{v}\} \cdot \llbracket \mathbf{u} \rrbracket \right] ds \\ &\quad + \sum_{\Gamma_{INP}} \int_e \mu \llbracket \mathbf{u} \rrbracket \cdot \llbracket \mathbf{v} \rrbracket ds. \end{aligned} \quad (3.24)$$

The justification is that the IP method and the local DG method are asymptotically similar for stability, boundedness, and the optimal rate of convergence as shown by Arnold et al. [5] in a unified analysis of the DG methods for elliptic problems. Note that since the replacement is applied at the algebraic level, no unphysical BCs have been introduced. Denoting the matrix form of the operator 3.24 with \mathcal{A} and then replacing

$(-\mathbf{D}\mathbf{H}_I\mathbf{D}^T)$ with \mathcal{A} in 3.22 yields

1. $\mathbf{H}\tilde{\mathbf{u}}^{n+1} = \underline{\mathbf{f}}^{n+1} - \mathbf{D}^T\hat{\underline{p}}^n,$
2. $\frac{\Delta t}{\beta_0}\mathcal{A}(\hat{\underline{p}}^{n+1} - \hat{\underline{p}}^n) = \mathbf{D}\tilde{\mathbf{u}}^{n+1} - \underline{\mathbf{g}}^{n+1},$
3. $\hat{\underline{u}}^{n+1} = \tilde{\underline{u}}^{n+1} - \mathbf{H}_I\mathbf{D}^T(\hat{\underline{p}}^{n+1} - \hat{\underline{p}}^n).$

(3.25)

Using the solution procedure 3.25 instead of 3.22 simplifies the whole method, in the sense that we use the same scheme (the IP method) for the velocity and pressure operators. Moreover, it enhances the overall efficiency of the scheme by reducing the cost per iteration of a pressure solve.

3.3 Implementation

For the approximating polynomial space for the velocity or pressure restricted to each element, $P_k(K)$, we choose a high-order nodal basis consisting of Lagrange interpolating polynomials defined on a reference simplex and the nodal set introduced in [57, 58], in two and three space dimensions. More specifically, let $\Xi = \{\boldsymbol{\xi}_i \in O : 0 \leq i \leq N\}$ denote the nodal set, where O is the reference element, and $N + 1 = (k + 1)(k + 2)/2$ or $N + 1 = (k + 1)(k + 2)(k + 3)/6$ for triangular or tetrahedral elements, respectively. Then, the nodal basis is a set of Lagrange interpolating polynomials with

$$L_i(\boldsymbol{\xi}_j) = \delta_{ij}, \quad \forall i, j = 0, \dots, N,$$

where δ_{ij} denotes the Kronecker delta. The interpolation representation of a function $f \in P_k(K)$ is

$$f(\boldsymbol{\xi}) = \sum_{j=0}^N f(\boldsymbol{\xi}_j)L_j(\boldsymbol{\xi}). \quad (3.26)$$

The Lagrange polynomials are the solution of the following system:

$$\sum_{j=0}^N b_i(\boldsymbol{\xi}_j)L_j(\boldsymbol{\xi}) = b_i(\boldsymbol{\xi}), \quad \forall i = 0, \dots, N, \quad (3.27)$$

with $\mathcal{J} = \{b_i(\boldsymbol{\xi}) | \boldsymbol{\xi} \in O, 0 \leq i \leq N\}$ being an orthonormal basis consisting of multivariate analogues of the Jacobi polynomials (see [71, 38]). The coefficient matrix in eq. 3.27 is called Vandemonde matrix and is denoted by V hereafter.

Below, we first briefly present the derivative and inner-product calculations following [119, 59], before describing a quadrature scheme for the nonlinear term evaluation.

3.3.1 Differentiation

For differentiation of a function, let us consider $\frac{\partial f}{\partial x}$, where x is a coordinate direction defined over an arbitrary tetrahedron K . Taking the derivative of the interpolation representation of $f(\mathbf{x})$, eq. 3.26, at the nodal point \mathbf{x}_i leads to

$$\frac{\partial f(\mathbf{x}_i)}{\partial x} = \sum_{j=0}^N f(\mathbf{x}_j) \frac{\partial L_j(\mathbf{x}_i)}{\partial x}.$$

Then, expanding the derivatives for the reference coordinates (ξ, η, ζ) defined on the reference tetrahedron O yields

$$\frac{\partial L_j(\mathbf{x}_i)}{\partial x} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \xi} \frac{\partial \xi}{\partial x} + \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \eta} \frac{\partial \eta}{\partial x} + \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \zeta} \frac{\partial \zeta}{\partial x}.$$

Let us define the coefficients of the above expression as

$$D_{ij}^{\xi} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \xi}, \quad D_{ij}^{\eta} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \eta}, \quad D_{ij}^{\zeta} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \zeta},$$

where $\partial \xi / \partial x$, $\partial \eta / \partial x$, and $\partial \zeta / \partial x$ result from the affine map between O and K . Then

$$D_{ij}^{\xi} = \sum_{k=0}^N V_{jk}^{-1} \frac{\partial b_k(\boldsymbol{\xi}_i)}{\partial \xi}, \quad D_{ij}^{\eta} = \sum_{k=0}^N V_{jk}^{-1} \frac{\partial b_k(\boldsymbol{\xi}_i)}{\partial \eta}, \quad D_{ij}^{\zeta} = \sum_{k=0}^N V_{jk}^{-1} \frac{\partial b_k(\boldsymbol{\xi}_i)}{\partial \zeta},$$

where derivatives of $b_i(\boldsymbol{\xi})$ are analytically given (e.g., [110]). V^{-1} is the inverse of the Vandermonde matrix, V .

3.3.2 Inner-Product Evaluations

For DG methods, we need to evaluate inner products of two functions on an element and on the boundary of an element, as well as inner products of the derivatives of functions.

Here, we describe only the procedure for calculating the inner product of two functions

$$(f, g)_K = \int_K f(\mathbf{x})g(\mathbf{x})d\mathbf{x}.$$

Other inner-product evaluations follow similarly. Using the interpolation approximation, eq. 3.26, we obtain

$$(f, g)_K = \int_K \sum_{i=0}^N f_i L_i(\mathbf{x}) \sum_{j=0}^N g_j L_j(\mathbf{x}) d\mathbf{x} = \sum_{i=0}^N \sum_{j=0}^N f_i g_j \int_K L_i(\mathbf{x}) L_j(\mathbf{x}) d\mathbf{x}.$$

The integral is evaluated on the reference element by using the Jacobian of the transformation J_K

$$(f, g)_K = \sum_{i=0}^N \sum_{j=0}^N f_i g_j \int_O L_i(\boldsymbol{\xi}) L_j(\boldsymbol{\xi}) J_K d\boldsymbol{\xi}.$$

Using eq. 3.27 yields

$$\begin{aligned} (f, g)_K &= \sum_{i=0}^N \sum_{j=0}^N f_i g_j \int_O \sum_{k=0}^N V_{ik}^{-1} b_k(\boldsymbol{\xi}) \sum_{l=0}^N V_{jl}^{-1} b_l(\boldsymbol{\xi}) J_K d\boldsymbol{\xi} \\ &= \sum_{i=0}^N \sum_{j=0}^N f_i g_j \sum_{k=0}^N \sum_{l=0}^N V_{ik}^{-1} V_{jl}^{-1} \int_O b_k(\boldsymbol{\xi}) b_l(\boldsymbol{\xi}) J_K d\boldsymbol{\xi}. \end{aligned}$$

In matrix form

$$(f, g)_K = [f_0 \dots f_N] \begin{bmatrix} V_{00}^{-1} & \dots & V_{0N}^{-1} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ V_{N0}^{-1} & \dots & V_{NN}^{-1} \end{bmatrix} W \begin{pmatrix} \begin{bmatrix} b_0(\boldsymbol{\xi}) \\ \cdot \\ \cdot \\ \cdot \\ b_N(\boldsymbol{\xi}) \end{bmatrix} \\ [b_0(\boldsymbol{\xi}) \dots b_N(\boldsymbol{\xi})] \end{pmatrix} \begin{bmatrix} V_{00}^{-1} & \dots & V_{N0}^{-1} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ V_{0N}^{-1} & \dots & V_{NN}^{-1} \end{bmatrix} \begin{bmatrix} g_0 \\ \cdot \\ \cdot \\ \cdot \\ g_N \end{bmatrix}, \quad (3.28)$$

where

$$W_{kl}(bb^T) = \int_O b_k(\boldsymbol{\xi}) b_l(\boldsymbol{\xi}) J_K d\boldsymbol{\xi}.$$

For each element, the matrix $V^{-1}W(V^{-1})^T$ is calculated in a preprocessing step. In the case of straight-sided triangles or tetrahedrons with planar faces, the Jacobian is constant.

Thus, $W = J_K I$, where I is the identity matrix. In this case, the matrix $V^{-1}(V^{-1})^T$

may be precalculated. Then the inner product for element K becomes $J_K f^T V^{-1} (V^{-1})^T g$. For the case of curved elements with nonconstant Jacobians, the integrals $W_{kl}(bb^T)$ are evaluated exactly using numerical quadrature [32].

3.3.3 Quadrature for the Nonlinear Term

The nonlinear term is evaluated by using quadrature of sufficiently high order to ensure accurate and stable integration. Specifically, to eliminate any possible quadrature effects in the following tests, for $k \leq 6$, a quadrature order $q = 3k$ has been used, while for $k = 7$, and 8, $q = 19$ has been used. This yields exact integral evaluation for $k \leq 6$ and a highly accurate integration scheme for $k = 7$, and 8. Let $R = \{\zeta_i \in O : 0 \leq i \leq M\}$ denote a set of $(M + 1)$ quadrature points and $W = \{w_i : 0 \leq i \leq M\}$ be the corresponding set of quadrature weights. Then, the numerical integration of the first term on the right-hand side of eq. 3.5 is carried out as

$$\int_K \underline{\underline{g}} \cdot \nabla \cdot \mathbf{v}_h d\mathbf{x} \approx \sum_{i=0}^M J_K w_i \underline{\underline{g}}(\zeta_i) \cdot \nabla \cdot \mathbf{v}_h(\zeta_i), \quad (3.29)$$

where

$$\underline{\underline{g}}(\zeta_i) = \sum_{j=0}^N L_j(\zeta_i) \underline{\underline{g}}(\xi_j), \quad (3.30a)$$

$$\nabla \cdot \mathbf{v}_h(\zeta_i) = \sum_{j=0}^N L_j(\zeta_i) \nabla \cdot \mathbf{v}_h(\xi_j). \quad (3.30b)$$

The numerical integration of the surface integral in eq. 3.5 is carried out in a similar manner and the quadrature rule of $q \approx 3k$ is also used. For the following tests, we have used R and W reported in [108].

3.4 Verification on Two-dimensional Problems

Below, we present some benchmarking tests to verify the accuracy of the proposed method. We will solve an unsteady Stokes problem to confirm temporal convergence

and will then examine the spatial accuracy and the stability of the scheme using equal- and mixed-order methods for the Navier-Stokes equations for three tests: the Taylor vortex problem, the Orr-Sommerfeld plane channel stability problem and flow past a square cylinder at $Re = 100$.

3.4.1 Temporal Error Test

Using a second-order backward differentiation temporal discretization and our proposed DG scheme, we solved the unsteady Stokes problem (eqs. 3.1a and 3.1b without the nonlinear term and the body force, $Re = 1.0$), having the exact solution

$$\begin{aligned} \mathbf{u} &= (\sin(x)(a \sin(ay) - \cos(a) \sinh(y))\mathbf{i} \\ &\quad + \cos(x)(\cos(ay) + \cos(a) \cosh(y))\mathbf{j}) \exp(-\lambda t), \end{aligned} \quad (3.31a)$$

$$p = \lambda \cos(a) \cos(x) \sinh(y) \exp(-\lambda t), \quad (3.31b)$$

where $a = 2.883356$ and $\lambda = 9.313739$ [82]. The computational domain was $\Omega = [-1, 1]^2$, and Dirichlet BCs and initial conditions were based on the above exact solution. We used equal interpolation orders of 8 for the velocity and pressure, and the mesh consisted of 72 semi-structured triangles (similar to that in Fig. 3.3a). The following results were essentially identical to those obtained using a lower interpolation order $k = 7$, confirming that spatial errors were dominated by the temporal errors, as required for a temporal convergence study. We measured the errors in the L^2 norms:

$$\|e_u\| = \frac{\|\mathbf{u}(n\Delta t) - \mathbf{u}_h(n\Delta t)\|_{L^2(\Omega)}}{\|\mathbf{u}^n\|_{L^2(\Omega)}}, \quad (3.32a)$$

$$\|e_p\| = \frac{\|p(n\Delta t) - p_h(n\Delta t)\|_{L^2(\Omega)}}{\|p^n\|_{L^2(\Omega)}}. \quad (3.32b)$$

The L^2 norms were calculated based on the nodal values. For this problem test and the next two tests, namely the Taylor vortex and Orr-Sommerfeld stability problems, we used absolute tolerances of 10^{-11} for the preconditioned conjugate gradient iterations of the velocity and pressure calculations. For a total integration time $T = 0.1$ during which

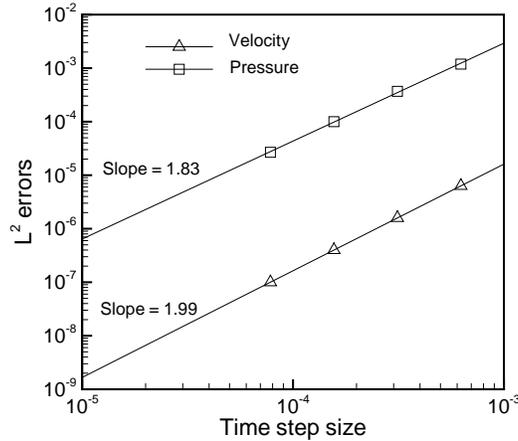


Figure 3.2: L^2 norm errors in velocity and pressure calculated based on eq. 3.32 vs. the time step size in solving an unsteady Stokes problem with the DG method described in Section 3.2, a semi-structured mesh consisting of 72 triangles (Fig. 3.3a), and equal interpolation orders of 8. For each data set, the best linear fit and its slope are also shown.

the initial solution decayed approximately three-fold, the results are depicted in Fig. 3.2. The slope of the best linear fit for the velocity is 1.99, verifying the expected convergence rate. For the pressure the slope is 1.83, slightly smaller than, but close to, the expected theoretical value.

3.4.2 Taylor Vortex Problem

For the first spatial error test, we solved the unsteady Navier-Stokes equations on the square domain $[-1, 1]^2$ with $Re = 100$, Dirichlet BCs and initial conditions based on the exact solution:

$$\mathbf{u} = (-\cos(\pi x) \sin(\pi y) \mathbf{i} + \sin(\pi x) \cos(\pi y) \mathbf{j}) \exp\left(\frac{-2\pi^2 t}{Re}\right), \quad (3.33a)$$

$$p = -\frac{\cos(2\pi x) + \cos(2\pi y)}{4} \exp\left(\frac{-4\pi^2 t}{Re}\right). \quad (3.33b)$$

We carried out convergence studies for both successive approximation order enrichment (p-convergence) and successive mesh refinements (h-convergence).

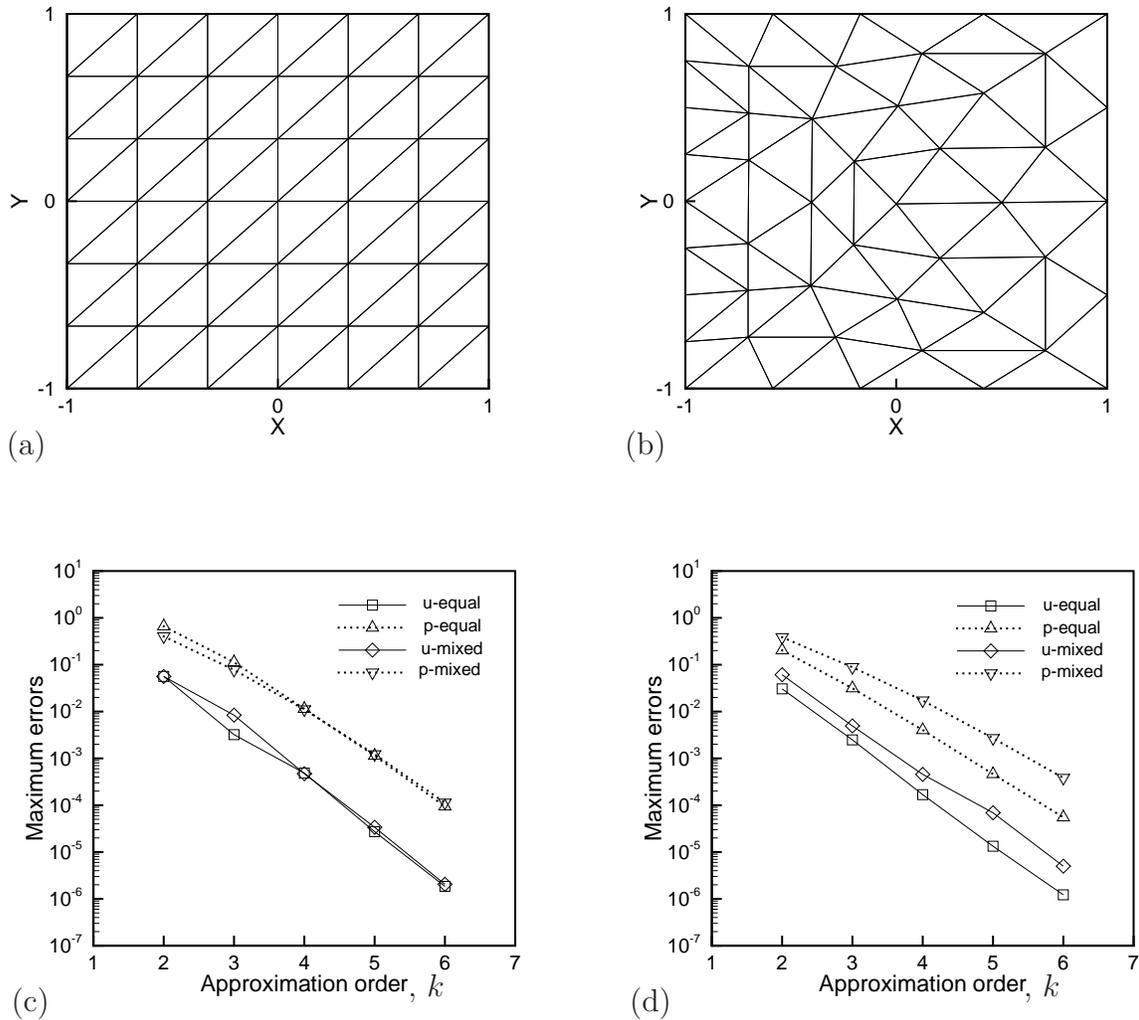


Figure 3.3: (a) Computational domain used for tests partitioned into 72 semi-structured triangular elements; (b) the same domain partitioned into 72 unstructured triangular elements generated using Gmsh software [46]; (c) and (d) maximum errors in calculated velocity and pressure vs. approximating polynomial degrees in solving the Taylor vortex problem using meshes in Figs. 3.3a and b, respectively. In the legend, “u-equal” refers to the error in velocity for equal order interpolation. “p-equal” is the corresponding error in the pressure. “mixed” refers to the $P_k - P_{k-1}$ formulation.

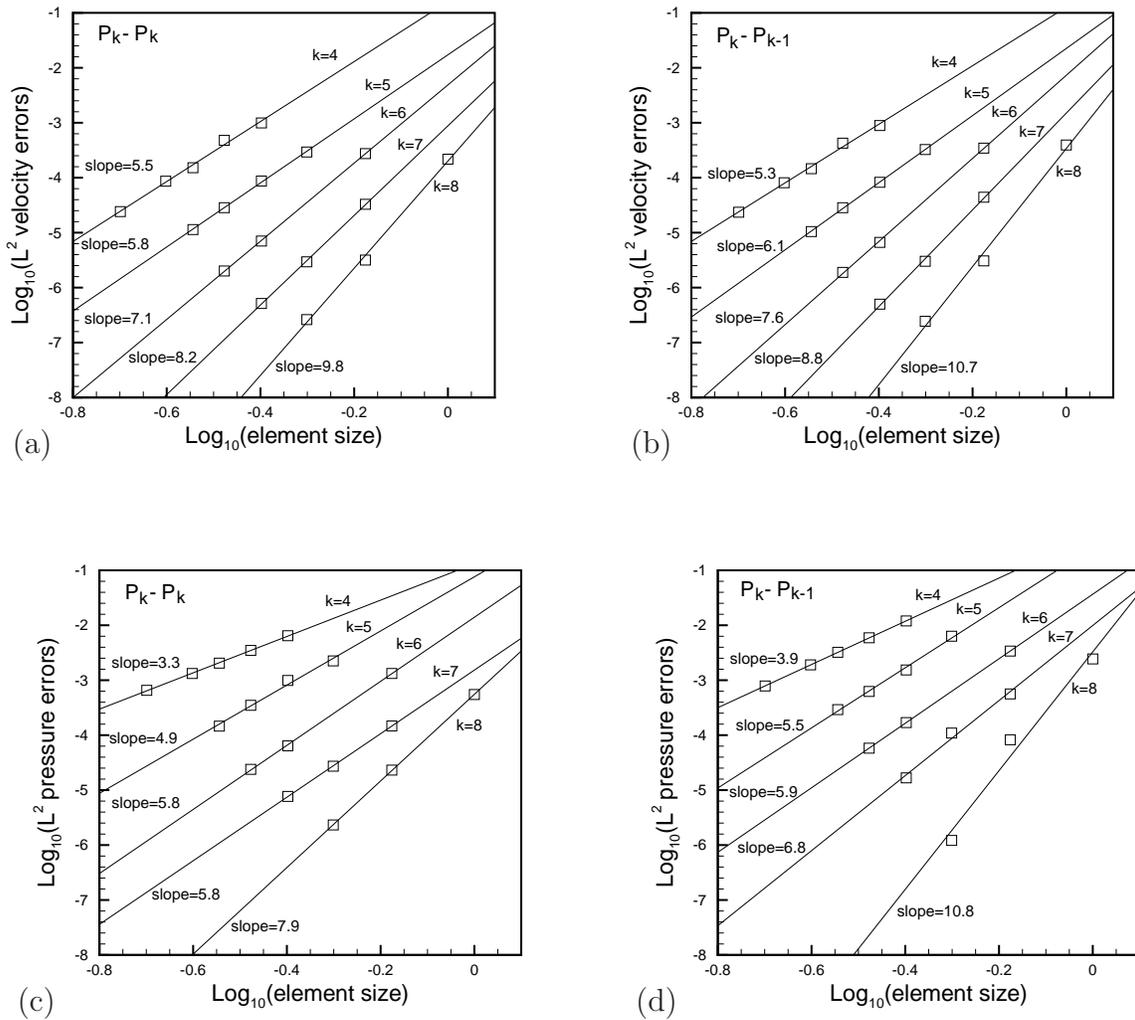


Figure 3.4: (a) and (b) L^2 velocity errors vs. element size of semi-structured meshes for interpolation order $k = 4, \dots, 8$, in solving the Taylor vortex problem using equal- and mixed-order methods, respectively. (c) and (d) the corresponding equal- and mixed-order data for the pressure. For each data set, the best linear fit and its slope are also shown.

For the p-convergence test, two meshes were used: a semi-structured mesh (Fig. 3.3a) and a fully unstructured one (Fig. 3.3b). The time step size $\Delta t = 10^{-4}$ was chosen to satisfy the CFL condition and to ensure that the dominant error was the spatial error. (This was verified by choosing a larger time step $\Delta t = 2 \times 10^{-4}$, which gave virtually identical results). The relative maximum errors in the calculated velocity and pressure for both equal- and mixed-order methods at $T = 5$ (corresponding to an approximately three-fold decay of the initial solution) and for a range of polynomial degrees $k = 2, \dots, 6$ are depicted in Figs. 3.3c and d for the semi-structured and unstructured meshes, respectively. Several points about the results are notable. First, a spectral rate of convergence was obtained with respect to the approximating polynomial degree for both velocity and pressure and for both mixed- and equal-order formulations. Second, for the semi-structured mesh, equal- and mixed-order methods led to results with very similar accuracy. On the other hand, for the unstructured mesh, using $P_k - P_k$ interpolations yielded more accurate results in both velocity and pressure than those resulting from $P_k - P_{k-1}$ interpolations.

To test the h-convergence, we only used a series of semi-structured meshes (similar to Fig. 3.3a). For the same time step size and the total time as the former case, the relative L^2 errors in calculated velocity versus element size for a range of polynomial degrees $k = 4, \dots, 8$ for both $P_k - P_k$ and $P_k - P_{k-1}$ formulations are depicted in Figs. 3.4a and b, respectively. Figs. 3.4c and d show the corresponding results of equal- and mixed-order methods for the calculated pressure. Similar to the p-convergence test, $P_k - P_k$ methods led to slightly more accurate results than those of $P_k - P_{k-1}$ for most cases. Moreover, we observed optimal rates of convergence in velocity for both $P_k - P_k$ and $P_k - P_{k-1}$ formulations. For the pressure, on the other hand, optimal rates of convergence were only obtained for the mixed-order method, as expected. The equal-order method led to suboptimal rates for the pressure.

Note that although both equal- and mixed-order methods led to stable results for this simple problem, the mixed-order method leads to unstable results for more challenging

(high Reynolds number) tests such as Orr-Sommerfeld stability problem as shown below.

3.4.3 Orr-Sommerfeld Stability Problem

We further investigated the spatial accuracy as well as the stability of our proposed method by solving the Orr-Sommerfeld stability problem. This is a suitable benchmarking test in that it is an unforced time-dependent solution of the Navier-Stokes equations for which an accurate solution is available from linear stability analysis. The geometry was a two-dimensional channel $[x = 0, x = 2\pi] \times [y = -1, y = 1]$. Dirichlet BCs were imposed in the spanwise direction (at $y = -1$ and $y = 1$) and periodic BCs were applied in the streamwise direction (at $x = 0$ and $x = 2\pi$). The initial conditions were

$$u = 1 - y^2 + \epsilon \hat{u}, \quad (3.34a)$$

$$v = \epsilon \hat{v}, \quad (3.34b)$$

where (u, v) represent the velocity components in the (x, y) directions. Here (\hat{u}, \hat{v}) (Tollmien-Schlichting waves, T-S waves) correspond to the only unstable eigensolution of the Orr-Sommerfeld equation with wave number unity at $Re = 7500$. We set ϵ to 10^{-4} . More details of this test can be found in [19].

According to linear stability theory, the perturbation energy

$$E(t) = \int_0^{2\pi} \int_{-1}^1 [(1 - y^2 - u)^2 + v^2] dy dx \quad (3.35)$$

should grow as $e^{2\omega_i t}$, where $\omega_i = 0.002234976$ is the growth rate.

For both $P_k - P_k$ and $P_k - P_{k-1}$ formulations with $k = 6$ and 8 and for a semi-structured mesh consisting of 128 triangles, and $\Delta t = 10^{-3}$ (smaller than Δt arising from the CFL condition) we plot the computed perturbation energy and its growth rate versus the normalized time (T/T_0) in Figs. 3.5a and b, respectively. In the same figures, the corresponding results from linear stability theory are also depicted. In the $P_k - P_k$ formulation for $k = 6$, we observed some dissipation (Fig. 3.5a); however, increasing the

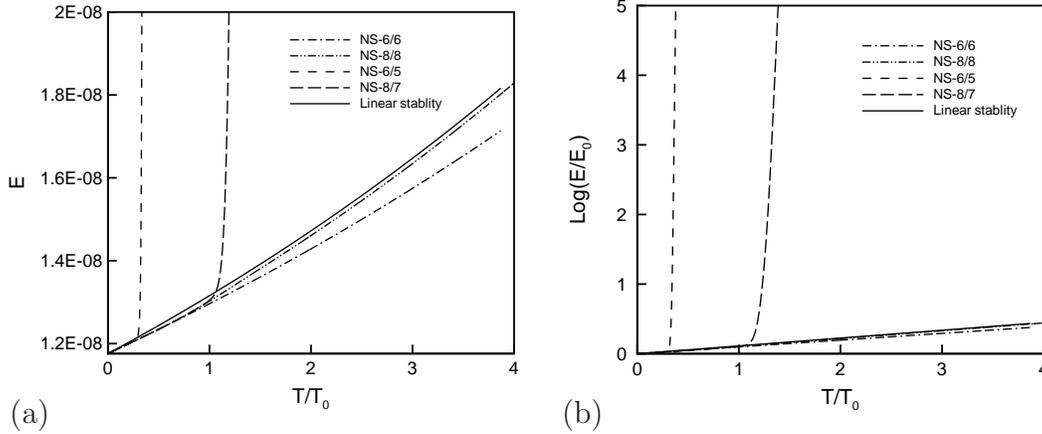


Figure 3.5: (a) Perturbation energy vs. normalized time in solving the Orr-Sommerfeld problem for both equal- and mixed-order methods; (b) the same test, showing the energy growth rate. The symbols NS-6/6 and NS-6/5 (NS-8/8 and NS-8/7) represent the computed Navier-Stokes solutions using $P_k - P_k$ and $P_k - P_{k-1}$ with $k = 6$ ($k = 8$). Scaling parameters are $E_0 = E(t = 0)$, and $T_0 = 25.1437$, the time taken for a T-S wave to travel the channel length, 2π .

resolution to $k = 8$ led to perturbation energy growth almost identical to the theoretical one. We also calculated the error in the growth rate at $T = 60$. For $k = 6, 7$, and 8 , we obtained growth rates $\omega = 0.001936496, 0.002156142$, and 0.002234850 . The corresponding errors calculated as $e_g = |\omega - \omega_i|/\omega_i$ were $e_g = 1.34e - 1, 3.53e - 2$, and $5.62e - 5$. A spectral rate of convergence was clearly obtained.

In the $P_k - P_{k-1}$ formulation, however, we observed a totally different behavior. For $k = 6$, we observed perturbation energy blowup at $T \approx 0.3T_0$. As shown in Fig. 3.5b, this unphysical behavior was characterized by an orders-of-magnitude increase in the energy growth rate in a very short period of time. When the resolution was increased to $k = 8$, the same blowup occurred, but at a later time $T/T_0 \approx 1$, making the diagnosis of this instability more difficult.

The source of this instability is similar to that of the instability occurring in the $Q_k - Q_{k-2}$ spectral element solution scheme for the Navier-Stokes equations reported

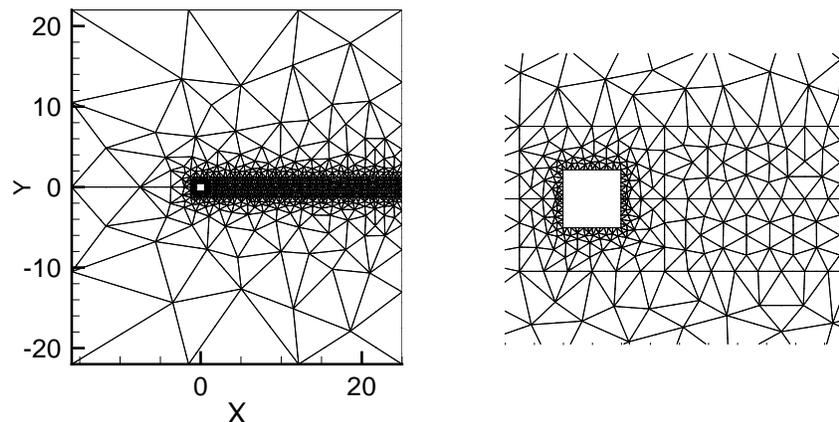


Figure 3.6: The mesh for flow past a square cylinder generated using Gmsh software [46]. Entire domain with 1706 triangles (left), and zoomed-in view near the cylinder (right).

in [121]. For the same Orr-Sommerfeld problem, Wilhelm and Kleiser [121] observed unphysical perturbation energy growth when the divergence form of the nonlinear term was discretized. Through an eigenvalue analysis of the full discretized linear system, they found eigenvalues with positive real parts (see Fig. 8 in [121]). Furthermore, they showed that the divergence of the velocity field grew exponentially at those points for which the divergence-free constraint was not enforced (see Figs. 4, 5 in [121]). For our mixed-order DG formulation the situation is similar. Since the velocity and pressure nodes are different, the divergence of the velocity field (in the weak sense) evaluated at the velocity nodes may grow and thus there is a chance of unphysical instability. For the equal-order method, however, velocity and pressure nodes are identical; thus, the divergence of the velocity field (in the weak sense) vanishes and the formulation is stable. The rigorous analysis of the stability of the method will be addressed in a forthcoming paper.

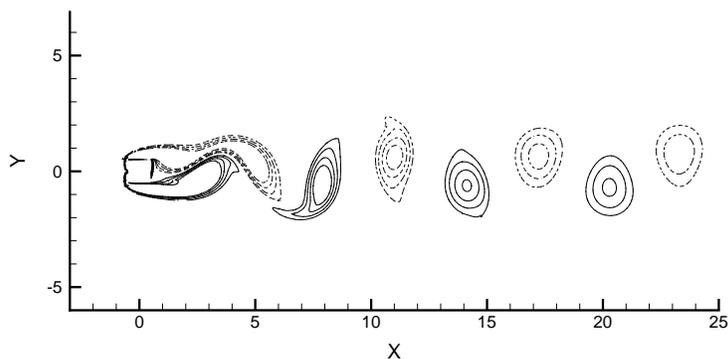


Figure 3.7: Snapshot of vorticity contours at $Re=100$. Solid and dashed lines denote positive and negative vorticity levels in $[-1, 1]$ with an increment of 0.25, respectively.

3.4.4 Flow Past a Square Cylinder

As a final test, we examine the spatial accuracy of our method by simulating vortex shedding in flow past a square cylinder at $Re = 100$, based on the unit inflow velocity and the square edge length. The geometry consisted of a unit square located in a rectangular domain with vertices $(-16, -22)$, $(25, -22)$, $(-16, 22)$, and $(25, 22)$, where the origin was placed in the center of the cylinder (see Fig. 3.6, left). This yields a blockage ratio $B = 2.3\%$. This geometry is identical to the geometry used in [31] and is chosen because it leads to geometry-independent results as shown in [10]. With flow in the positive x direction, we imposed the following boundary conditions: zero Dirichlet BCs on the square, $\mathbf{u} = (1, 0)$ on the inlet and side walls and outflow BCs at the outlet. The mesh consisted of 1706 triangles concentrated on the cylinder to resolve the large gradient of vorticity associated with the sharp corners (Fig. 3.6). We used the $P_k - P_k$ method with $k = 4$ and $\Delta t = 10^{-3}$. For this and the following test, we used absolute tolerances of 10^{-10} and 10^{-8} for the preconditioned conjugate gradient iterations of the velocity and pressure calculations, respectively.

An instantaneous view of the calculated vorticity contours is shown in Fig. 3.7, demonstrating the von-Karman vortex street downstream of the cylinder. For comparison

with available data in the literature, we calculated the Strouhal number $St = fD/u$, where f is the frequency of the vortex shedding, $u = 1$ is the inflow velocity and $D = 1$ is the edge length of the cylinder. The frequency was obtained from spectral analysis of the lift coefficient sampled over the time span of $t = 20$ dimensionless times, D/u . Our results, along with experimental data of Okajima [89] and the computational result of Darekar and Sherwin [31], are listed in Table 3.1. The latter was obtained using a spectral element Navier-Stokes solver with 1502 triangular elements of order $k = 6$. The total number of unknowns of this simulation was almost identical to that used in our DG simulation (1706 triangles of order $k = 4$). As is clear from the table, we obtained excellent agreement with both experimental and computational data.

Approach	St
Okajima (experimental) (1982), $B = 0\%$	0.141 – 0.145
Darekar and Sherwin (2001), $B = 2.3\%$	0.145
Present, $B = 2.3\%$	0.145

Table 3.1: Comparison of Strouhal number for vortex shedding in flow over a square cylinder at $Re = 100$. B is the blockage ratio.

3.5 Flow through a Two-dimensional Mechanical Heart Valve

Having verified the accuracy and stability of our developed methodology on several two-dimensional benchmarking problems with known solutions, in this section we consider flow through a simplified two-dimensional bileaflet mechanical heart valve geometry for which no detailed study, either experimental or computational, has been conducted previously. This two-dimensional study is the first step toward a three-dimensional mechanical

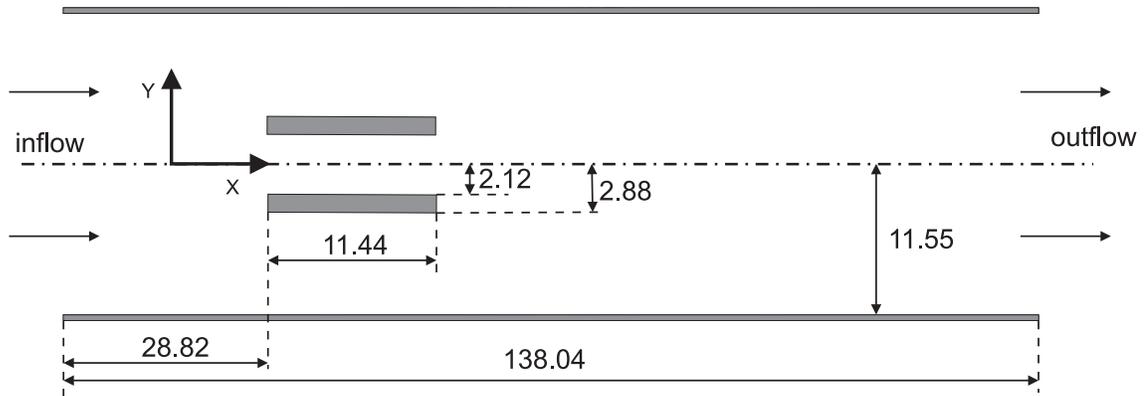


Figure 3.8: Geometry of the two-dimensional mechanical heart valve flow problem. The valve has two leaflets located symmetrically about, and parallel to, the centerline of the channel (the dashed-dotted line). The dimensions (in millimeters) are based on the Carbomedic aortic bileaflet valve with nominal diameter 23mm.

heart valve flow simulation and will reveal important features of the flow including symmetry breaking and vortex shedding, which can cause two-dimensional flows to transition to a three-dimensional flow.

The geometry consisted of a two-dimensional bileaflet mechanical heart valve located inside a straight channel (Fig. 3.8). The size and shape of the leaflets corresponded to the Carbomedic aortic bileaflet valve with nominal diameter 23mm [63]. The channel width was based on the three-dimensional model used for experimental investigation of the steady flow downstream of a bileaflet mechanical heart valve [18]. The outlet length was chosen to be approximately eight leaflet lengths to minimize the effect of the truncated domain.

The boundary conditions were specified as follows: homogeneous Dirichlet conditions for all walls and the leaflets, outflow conditions on the outlet (eq. 3.1e), and inflow at

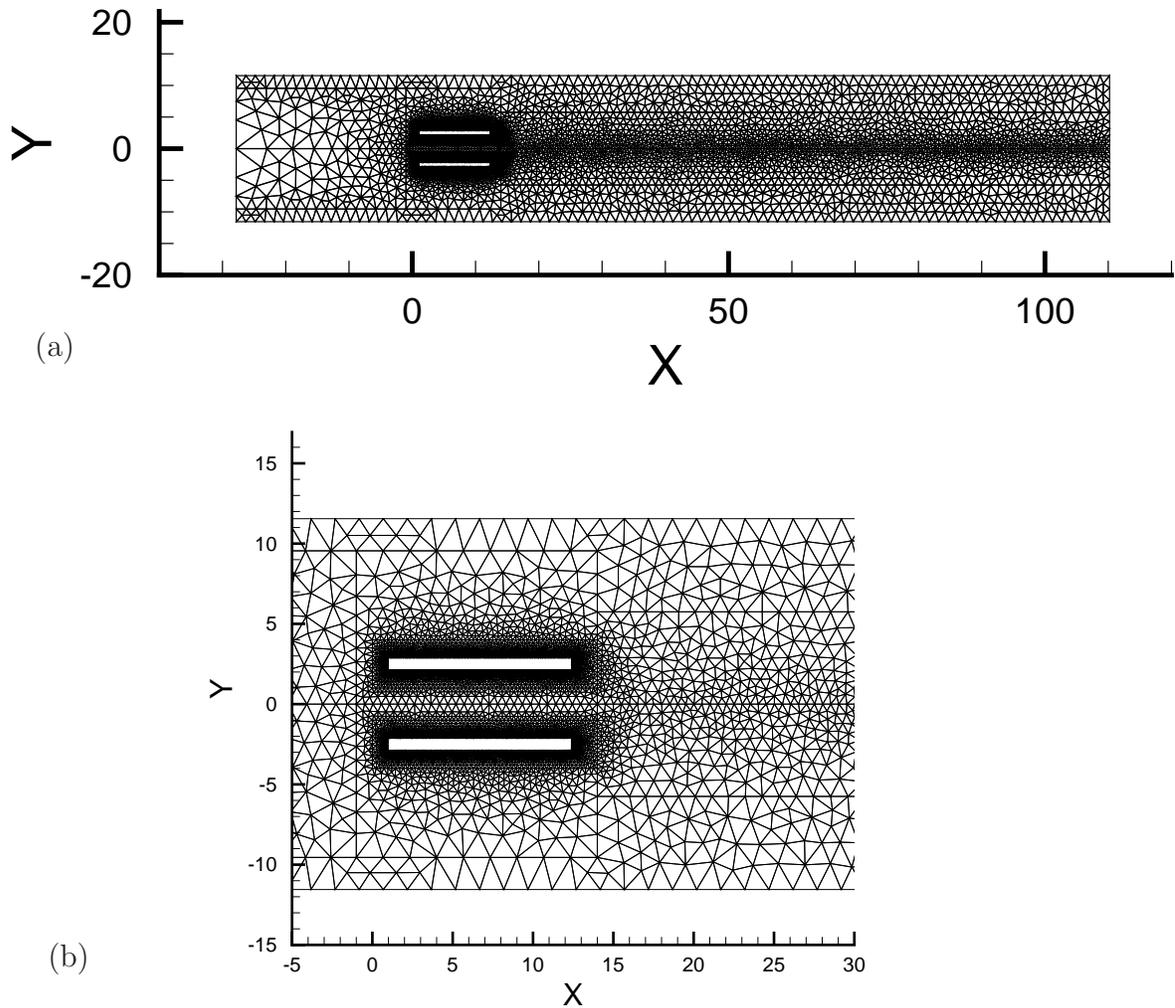


Figure 3.9: Triangular mesh (generated using Gmesh software [46]) used for simulating flow through a two-dimensional mechanical heart valve geometry; (a) the entire domain with 15902 triangles; (b) the zoomed-in view near the leaflets.

the inlet:

$$u = u_I \left(1 - \frac{y^2}{11.55^2}\right), \quad (3.36a)$$

$$v = 0, \quad (3.36b)$$

where, u_I the maximum inlet velocity. The Reynolds number was defined based on the maximum inlet velocity and the leaflet width (0.76mm). We here only studied the flow at $Re = 76$, leaving higher Reynolds number simulations for the future. Note that the corresponding Reynolds number based on the channel height and the mean inlet velocity

is 1540, which is close to the time-averaged Reynolds number for blood flow in the aorta.

The mesh consisted of 15902 triangles with very small elements concentrated around the leaflets to resolve the large gradient of velocity associated with the sharp corners (Fig. 3.9). Note that no effort was made to optimize the mesh. For instance, it would be possible to construct a more optimized mesh with smaller number of elements if highly anisotropic triangles with large sizes in the streamwise direction and small sizes in the spanwise direction were employed close to the channel walls.

A simulation was carried out using the equal-order scheme, with approximation order $k = 4$ and $\Delta t = 4 \times 10^{-3}$. Starting with the initial condition corresponding to the solution at $Re = 1$, the simulation ran for approximately two flow-through times until the L^2 norm of the difference in the consecutive velocity solutions dropped below 3×10^{-6} .

After the initial artificial transients, the flow reached a steady state condition, and no vortex shedding was observed. The calculated streamwise velocity contours are shown in Fig. 3.10, from which several points are notable. First, the flow remains symmetric with respect to the center line. Moreover, around each leaflet, two negative streamwise velocity regions are formed: one in the wake of the leaflet and the other on the outer edge of the leaflet near the leading edge. These retrograde flow regions indicate the existence of two recirculation zones. Finally, as is clear from Fig. 3.10c, the leaflet wake recirculation is asymmetric with respect to the leaflet. To validate these results, we carried out simulation with higher resolution of $k = 6$. For this simulation all parameters remained the same, except that the initial conditions were the solution obtained with $k = 4$. We plotted the velocity contours and found that they were indistinguishable from those at $k = 4$, confirming the resolution-independence of our results.

For solutions obtained using both $k = 4$ and 6, we also plotted the streamwise velocity along three observation lines and the spanwise velocity along two observation lines in Fig. 3.11a and b, respectively. As can be seen from the figures, we obtained indistinguishable results for both approximation orders. Also notable from Fig. 3.11a and b are very large

velocity gradients close to the leading leaflet's corner (along the line $y = -2.98$). This highlights the importance of using very fine elements close to the sharp corners.

This two-dimensional study suggests several important considerations for simulating flow through the three-dimensional valve geometry. First, since the ratio of the leaflet thickness to the vessel wall diameter is small (approximately 0.03) very fine elements are required close to the the leaflet to capture the geometry of the leaflet. This requirement of using very fine elements close to the leaflet is also critical for resolving the large gradient of field variables (such as velocity, vorticity) close to the sharp corners. Second, using optimized meshes with highly anisotropic elements along the vessel wall, stretched in the streamwise direction, is essential in reducing the high cost of simulating the three-dimensional flow.

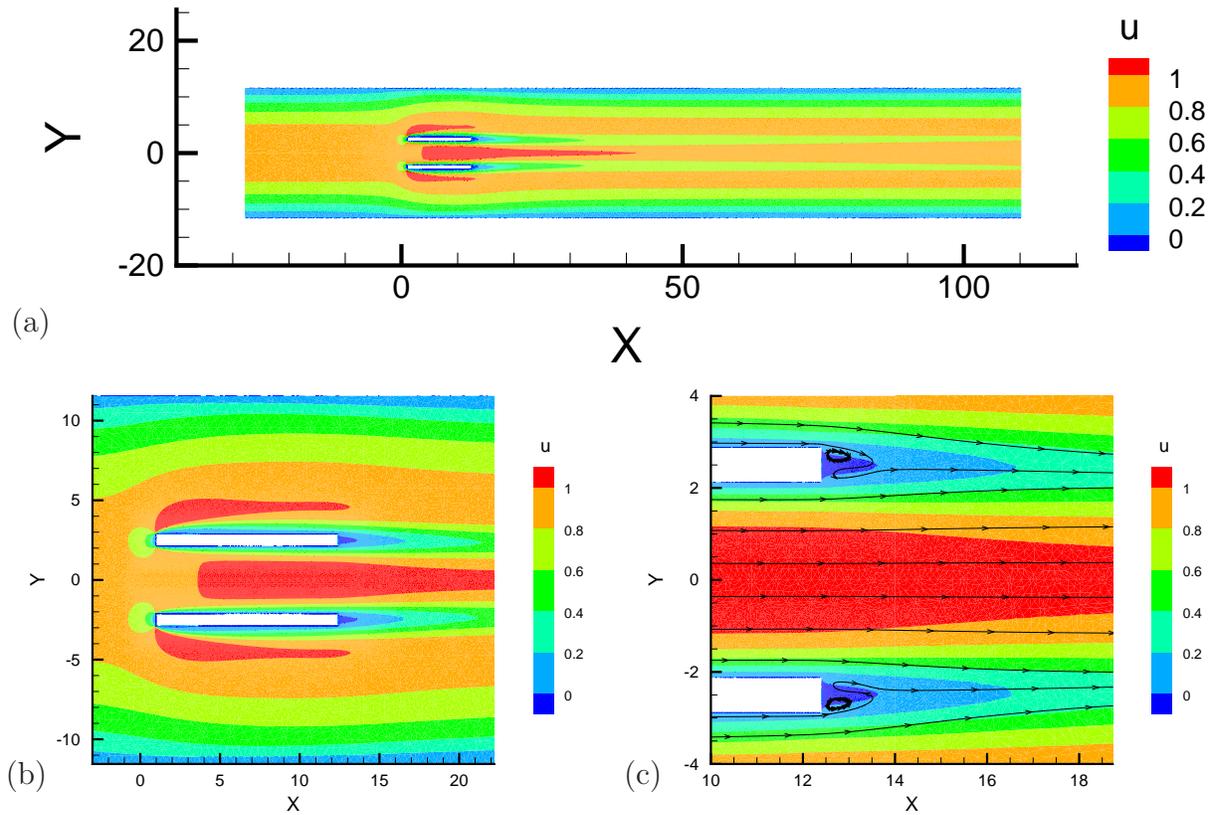


Figure 3.10: Streamwise velocity contours for flow through the two-dimensional mechanical heart valve at $Re = 76$ simulated using the DG method with $k = 4$; (a) the entire domain; (b) a zoomed-in view near the leaflets; (c) streamlines in the leaflet wakes. For each leaflet, two recirculation zones, one in the wake of the leaflet and the other at the outer edge close to the leading edge, are observed. (Computations were partially carried out by L. Vaisman.)

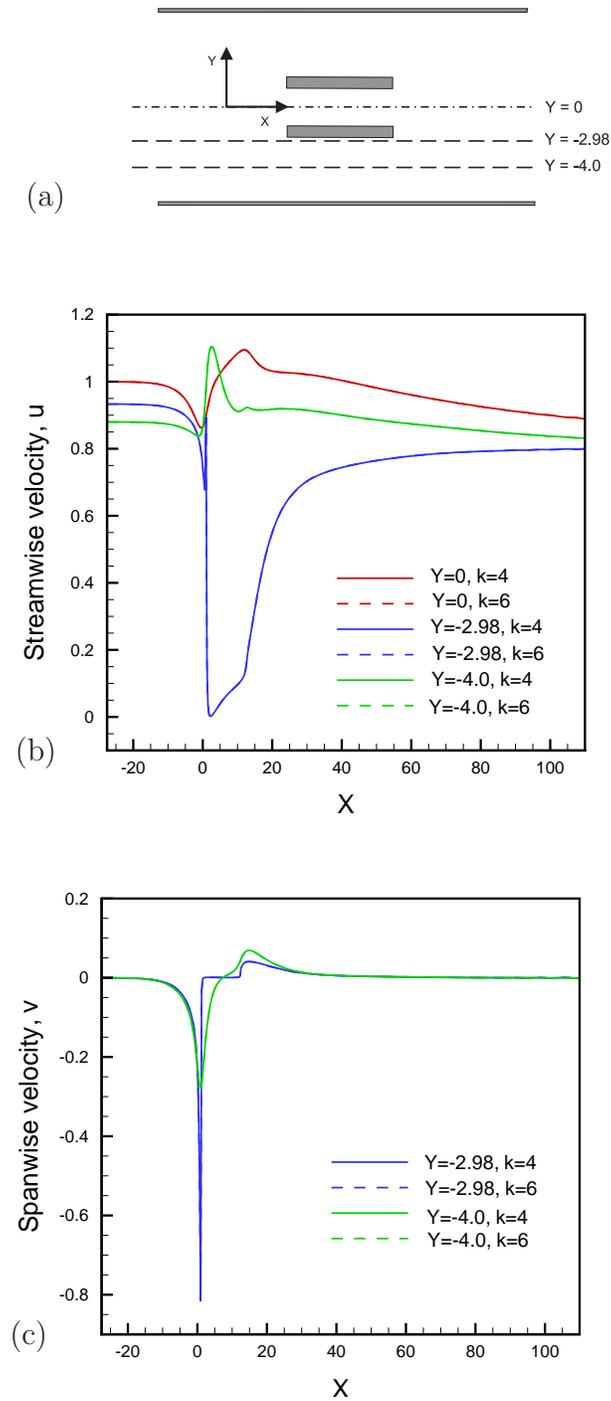


Figure 3.11: Velocity components along different lines for flow through the two-dimensional mechanical heart valve at $Re = 76$ calculated using the DG method with $k = 4$ and 6; (a) location of observation lines for panels (b) and (c); (b) streamwise velocity; (c) spanwise velocity. $X \approx 0$ corresponds to the leading edge of the valve.

Chapter 4

Verification on Three-dimensional Problems

In the previous chapter, we verified the performance of our method by solving two-dimensional benchmarking problems. It is also essential to examine the accuracy of the method by solving three-dimensional (3D) problems. To this end, we solve three problems in the this chapter, namely a generalized Taylor vortex flow, Orr-Sommerfeld stability problem and a backward-facing step flow.

4.1 Three-dimensional Taylor Vortex

Using the methodology developed in the previous chapters, we solved the unsteady Navier-Stokes equations on the cubic domain $[0, 2\pi]^3$ with $Re = 1$, and Dirichlet BCs

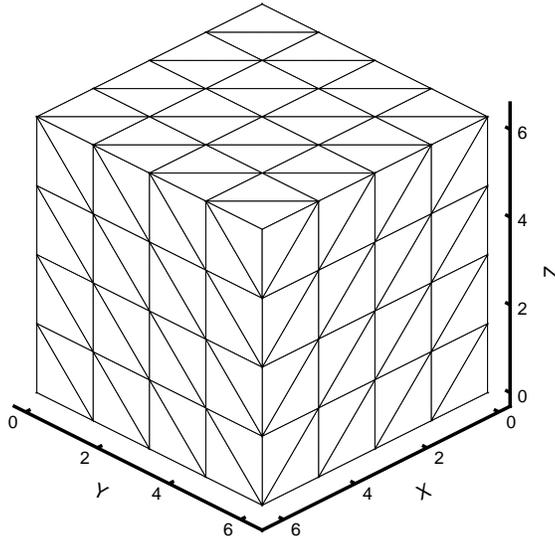


Figure 4.1: A structured mesh consisting of 384 tetrahedra ($M = 4$) used for the 3D Taylor vortex problem.

and initial conditions based on the exact solutions $\mathbf{u} = (u, v, w)$ and p :

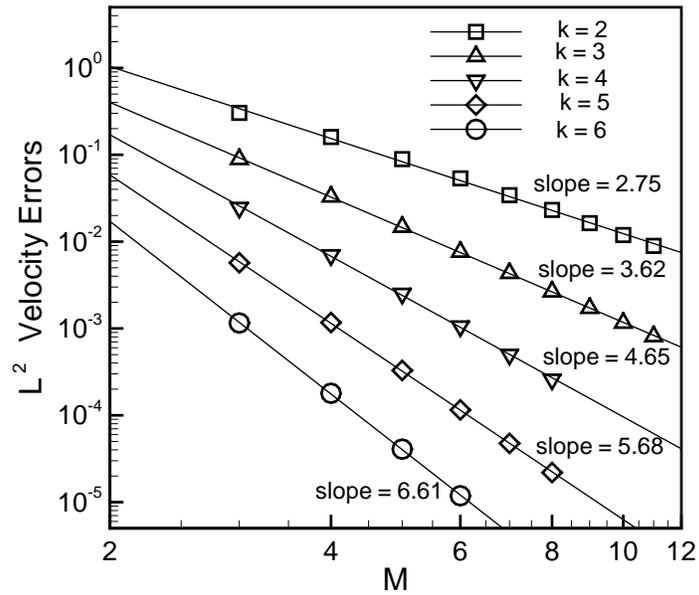
$$u = -\frac{1}{2}(\sqrt{3} \cos(x) \sin(y) \sin(z) + \sin(x) \cos(y) \cos(z)) \exp\left(\frac{-3t}{Re}\right), \quad (4.1a)$$

$$v = \frac{1}{2}(\sqrt{3} \sin(x) \cos(y) \sin(z) - \cos(x) \sin(y) \cos(z)) \exp\left(\frac{-3t}{Re}\right), \quad (4.1b)$$

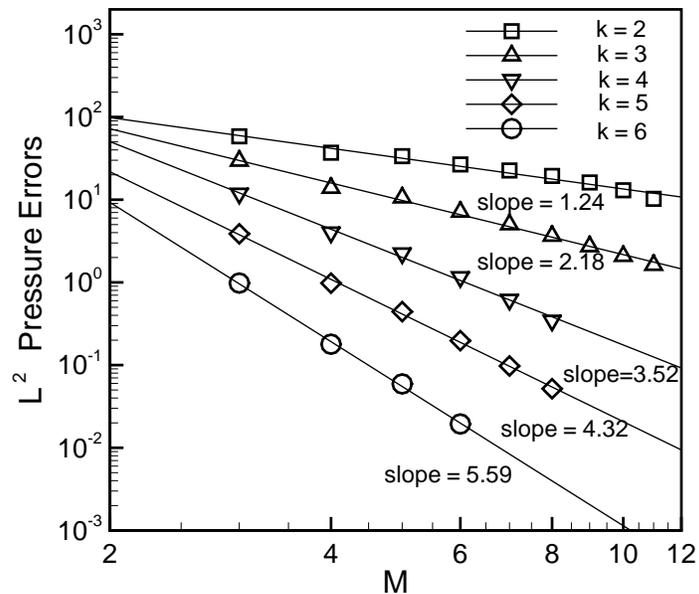
$$w = \cos(x) \cos(y) \sin(z) \exp\left(\frac{-3t}{Re}\right), \quad (4.1c)$$

$$p = -\frac{u^2 + v^2 + w^2}{2}. \quad (4.1d)$$

This solution of the Navier-Stokes equations first derived in [106] is a 3D generalization of the Taylor vortex problem studied in the previous chapter. We carried out convergence studies using a mesh series and for a range of approximation orders $k = 2, \dots, 6$. We used a structured mesh obtained from partitioning the domain into $6M^3$ uniform tetrahedra for $M = 3, \dots, 10$ (Fig. 4.1). The time step size $\Delta t = 10^{-4}$ was chosen to satisfy the CFL condition and to ensure that the dominant error was the spatial error. (This was verified by choosing a larger time step $\Delta t = 2 \times 10^{-4}$, which gave virtually identical results). We used absolute tolerances of 10^{-11} for the preconditioned conjugate gradient iterations of



(a)



(b)

Figure 4.2: (a) L^2 velocity errors vs. M (the number of partitions in each coordinate direction) for interpolation orders $k = 2, \dots, 6$, in solving the 3D Taylor vortex problem using $P_k - P_k$ equal-order method; (b) the same test showing the pressure error. The best linear fit and its slope are also shown for each data set.

Velocity								
k	$M = 3$	$M = 4$	$M = 5$	$M = 6$	$M = 7$	$M = 8$	$M = 9$	$M = 10$
2	2.23	2.63	2.80	2.87	2.93	3.00	2.99	3.01
3	3.46	3.58	3.66	3.65	3.65	3.70	3.74	3.76
4	4.37	4.57	4.75	4.88	4.94	*	*	*
5	5.51	5.69	5.75	5.74	5.80	*	*	*
6	6.48	6.64	6.80	*	*	*	*	*

Pressure								
k	$M = 3$	$M = 4$	$M = 5$	$M = 6$	$M = 7$	$M = 8$	$M = 9$	$M = 10$
2	1.59	0.43	1.26	1.08	1.14	1.53	2.08	2.54
3	2.61	1.21	2.17	2.27	2.40	2.50	2.55	2.61
4	3.78	2.59	2.97	4.07	4.36	*	*	*
5	4.77	3.55	4.44	4.58	4.74	*	*	*
6	5.91	5.19	6.12	*	*	*	*	*

Table 4.1: Convergence rates for solving the 3D Taylor vortex problem, as defined by eq. 4.2. “*” indicates that no data are available.

the velocity and pressure calculations. The relative L^2 errors were measured based on formulae similar to eq. 3.32. These errors in the calculated velocity and pressure for the $P_k - P_k$ equal-order method at $T = 0.4$ (corresponding to an approximately three-fold decay of the initial solution) are depicted in Figs. 4.2a and b, respectively. From Figs. 4.2a and b, it is clear that increasing the approximation order by one yields roughly an order of magnitude reduction in the velocity and pressure errors, consistent with a spectral rate of convergence. We further realize that the slope of the best linear fits for the velocity and pressure data are slightly smaller than the expected theoretical convergence rates (i.e., $k + 1$ for velocity and k for pressure). This is due to fact that some of the meshes

Order	ω	<i>error</i>
2	-0.0278808	$1.24e - 2$
3	-0.0280851	$4.84e - 3$
4	-0.0282074	$4.71e - 4$
5	-0.0282286	$6.83e - 5$
6	-0.0282314	$3.30e - 5$

Table 4.2: The averaged observed growth rates and the corresponding errors in the growth rate at different approximation orders in solving 3D Orr-Sommerfeld stability problem.

used (those with small M) were not sufficiently fine to yield the theoretical convergence rates. However, as shown in Table 4.1, pointwise convergence rates calculated based on the formula

$$\text{convergence rate} = \frac{\log\left(\frac{\text{error}(M)}{\text{error}(M+1)}\right)}{\log\left(\frac{M+1}{M}\right)} \quad (4.2)$$

grew with the refinement level M and approached the theoretical rates.

4.2 Three-dimensional Orr-Sommerfeld Stability

Our second test problem is the 3D version of the Orr-Sommerfeld stability problem considered in the previous chapter. The 3D Orr-Sommerfeld problem was previously studied by Krist and Zang [72]. The geometry was a 3D channel $[x = 0, x = 2\pi] \times [y = 0, y = 2\pi] \times [z = -1, z = 1]$. Homogeneous Dirichlet BCs were imposed at $z = -1$ and $z = 1$ and periodic BCs were applied in the streamwise direction (at $x = 0$ and $x = 2\pi$) and spanwise direction (at $y = 0$ and $y = 2\pi$). A parabolic mean flow was maintained in the x -direction by using a constant body force. This mean flow was disturbed with a

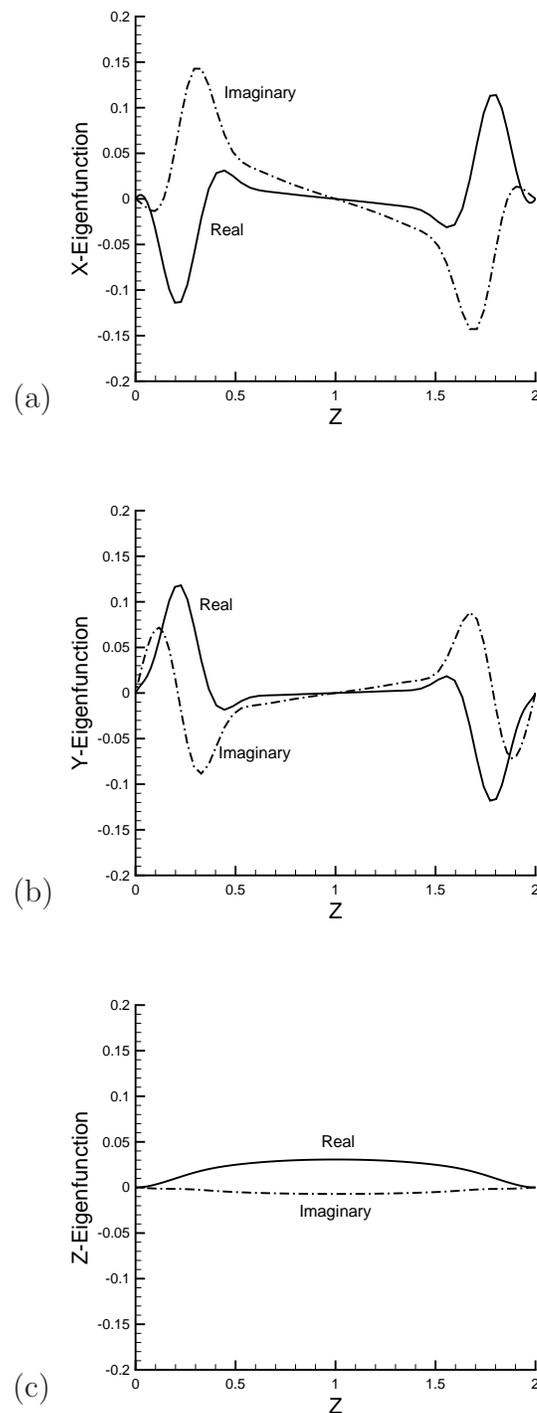


Figure 4.3: Eigenfunctions of the 3D Orr-Sommerfeld problem for $Re = 1500$ and wave numbers unity as described by eq. 4.3c; (a) x -direction eigenfunction; (b) y -direction eigenfunction; (c) z -direction eigenfunction.

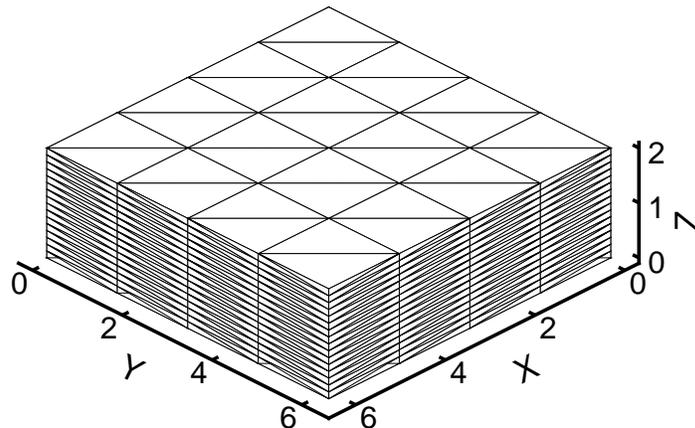


Figure 4.4: A structured mesh consisting of 1536 tetrahedra used for the 3D Orr-Sommerfeld problem. The number of partitions in z -direction is four-fold higher than that in each of x - and y -directions.

small amplitude 3D wave. Thus, the initial condition had the form

$$u = 1 - z^2 + \epsilon \hat{u}, \quad (4.3a)$$

$$v = \epsilon \hat{v}, \quad (4.3b)$$

$$w = \epsilon \hat{w}, \quad (4.3c)$$

where $(\hat{u}, \hat{v}, \hat{w})$ are the least damped eigensolution of the Orr-Sommerfeld equation with wave numbers unity at $Re = 1500$. These eigensolutions are depicted in Fig. 4.3. Since $(\hat{u}, \hat{v}, \hat{w})$ are normalized, ϵ represents the magnitude of the disturbances and we set ϵ to 10^{-4} .

According to linear stability theory, the perturbation energy

$$E(t) = \int_0^{2\pi} \int_0^{2\pi} \int_{-1}^1 [(1 - z^2 - u)^2 + v^2 + w^2] dz dx dy \quad (4.4)$$

should grow as $e^{2\omega_i t}$, where $\omega_i = -0.0282305$ is the growth rate. Note that the growth rate is negative and disturbances will damp.

We used $P_k - P_k$ formulation with a range of approximating orders $k = 2, \dots, 6$ and for a semi-structured mesh consisting of 1536 tetrahedra (Fig. 4.4), and $\Delta t = 10^{-3}$ (smaller

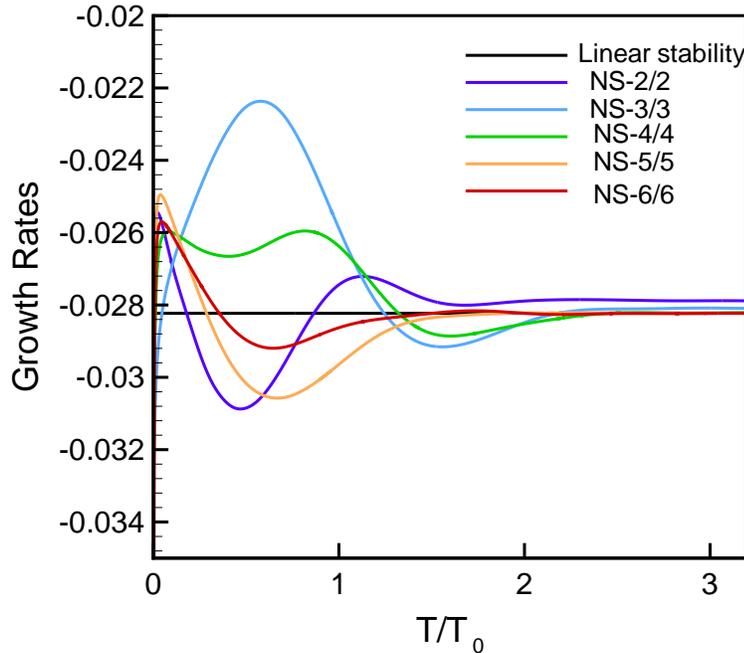


Figure 4.5: The computed perturbation growth rates vs. normalized time T/T_0 in solving the 3D Orr-Sommerfeld stability problem at $Re = 1500$. $T_0 = 15.657$ is the time taken for the perturbation wave to travel the channel length, 2π . A symbol of the form $NS - k/k$ signifies a Navier-Stokes solutions computed using the equal-order formulation of order k .

than Δt arising from the CFL condition). Absolute tolerances of 10^{-10} were used as convergence criteria for the velocity and pressure iterations. We plot the computed perturbation growth rates versus the normalized time (T/T_0) in Fig. 4.5. From the figure, two points are notable. First, at all orders k , we observe a transient period with unphysical growth rates. These unphysical growth rates are due to the initial solution, which does not satisfy the discretized system exactly. Second, after passing the transient period, the computed growth rates converge to the growth rate predicted by linear stability theory. To quantify the error in the calculated growth rates, we have calculated the averaged growth rates over a normalized time $T/T_0 = 1$ as well as the corresponding normalized errors. The results are listed in Table 4.2. Spectral convergence is clearly observed at all orders except at $k = 6$, where the spatial error size become comparable to the temporal

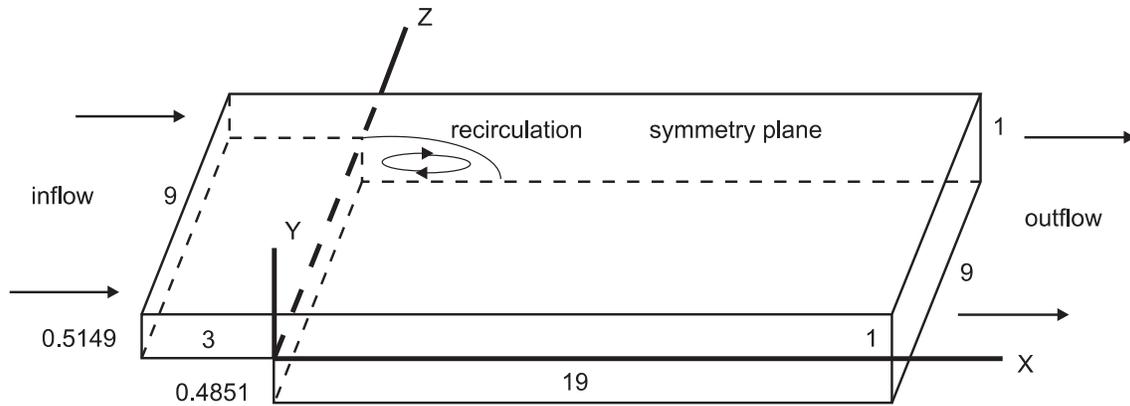


Figure 4.6: Geometry of the backward-facing step flow problem with a expansion ratio 1:1.94. The inflow velocity profile is given as a tensor product of a parabola and a Blasius boundary layer defined in the z direction. For the Re values considered here, only one recirculation area is formed whose location is indicated in the symmetry plane.

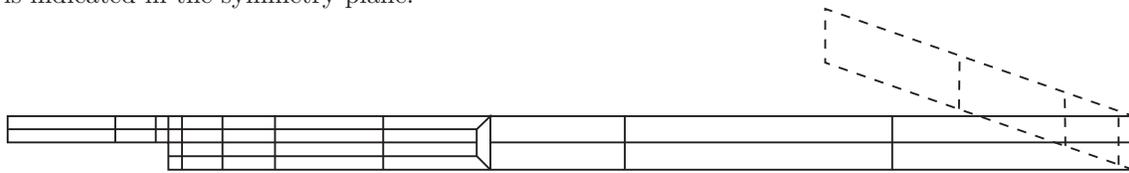


Figure 4.7: Mesh (consisting of 128 hexahedra) for backward-facing step problem (Fig. 4.6) used in a spectral element simulation with $k = 9$ and 11 by Couzy [29]. True aspect ratio is shown.

error size.

4.3 Backward-facing step flow

As a final test problem, we simulate flow over a 3D backward-facing step (BFS). This flow is characterized by three recirculation zones whose occurrences, sizes and locations depend on the Reynolds number of the flow and the expansion ratio of the step [3]. At an expansion ratio of 1 : 1.94 the first (primary) recirculation zone occurs on the bottom wall, directly at the base of the step. The second recirculation zone is formed on the upper wall downstream of the expansion for $400 \leq Re \leq 6600$. The third recirculation

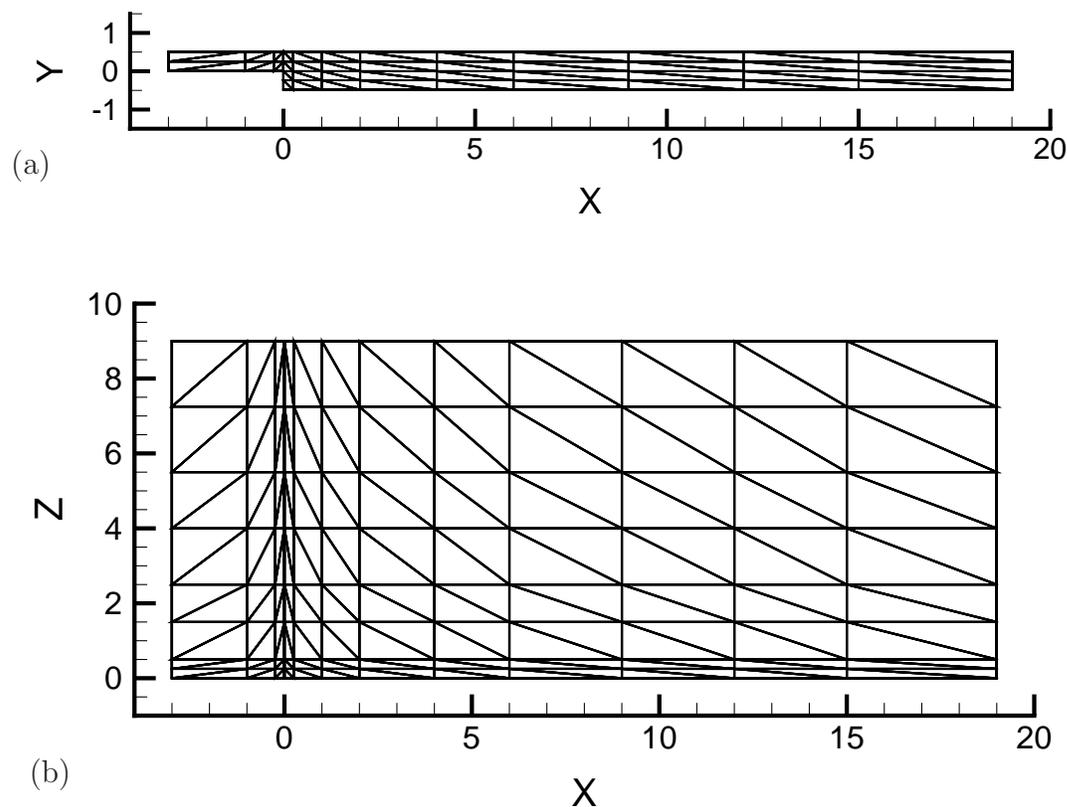


Figure 4.8: Tetrahedral Mesh consisting of 2016 elements used for simulating flow over a backward-facing step using the DG method with $k = 4$; (a) $x - y$ plane view; (b) $x - z$ plane view. The mesh is generated based on the spectral element mesh (Fig. 4.7) used by Couzy [29], but with two-fold higher partitioning in the z direction.

zone occurs at the bottom wall, just downstream of the first one for $1200 \leq Re \leq 2300$. Here, we focus on values of $Re < 400$, for which there exists only the primary recirculation zone.

The geometry consisted of a 3D step with the expansion ratio $1 : 1.94$ (Fig. 4.6). This geometry was previously used in the spectral element (SE) simulations by Couzy [29] and is based on the experimental setup of Armaly et al. [3]. The two-dimensional (2D) version of the geometry was also used by Biswas et al. [14]. The length of channel downstream of the step was chosen long enough to guarantee a fully developed flow at the outlet [29].

The Reynolds number is defined based on the bulk inlet velocity (i.e., two-thirds of

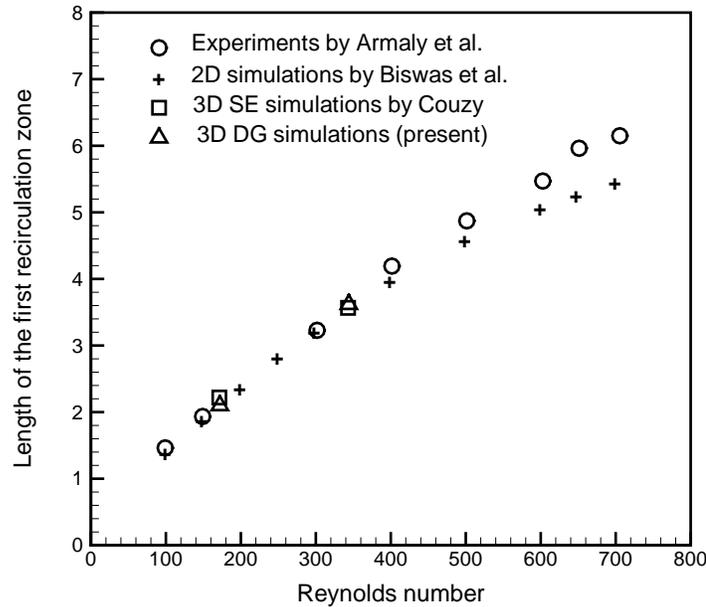


Figure 4.9: Length of the first recirculation zone in the backward-facing step problem. The results of the present 3D discontinuous Galerkin (DG) simulations are shown along with results of the experiments of Armaly et al. [3], the 2D simulations of Biswas et al. [14] and the 3D spectral element (SE) simulations of Couzy [29]. (In the original figure reporting the spectral element results [29], the value for higher Reynolds number of 344 was shown erroneously. It is corrected in this plot.)

the maximum inlet velocity, equal to unity) and a characteristic length which is chosen as twice the height of the inlet channel. We carried out simulations for two Reynolds numbers, $Re = 172$ and 344 , for which detailed experimental and numerical data are available in the literature for comparison purposes.

The boundary conditions were specified as follows: homogeneous Dirichlet conditions for all walls, symmetry conditions (i.e., zero spanwise velocity component and zero normal derivatives of streamwise and vertical velocity components) on the plane of symmetry, outflow conditions (zero normal stress and pressure, eq. 3.1e) on the outlet. The inflow

boundary conditions at $x = -3$ are

$$u = [15.08739(0.5149 - y)y]b(z), \quad (4.5a)$$

$$v = 0, \quad (4.5b)$$

$$w = 0, \quad (4.5c)$$

where the streamwise velocity u is a product of a parabola and a Blasius boundary layer profile $b(z)$. The Blasius boundary layer profile, $b(z)$, is characterized by the boundary layer thickness $\delta_{.99}(x)$ which represents the z -value at which u attains 99% of its maximum (e.g., [120]). Following [29], we set $\delta_{.99}(-3) = 0.5$. Hence, 4.5a yields a maximum inlet velocity of unity at $z > 0.5$ and $y = 0.5149/2$.

The initial conditions for $Re = 172$ was the solution at $Re = 1$ and for $Re = 344$ was the solution at $Re = 172$.

Based on the hexahedral mesh used in spectral element simulations by Couzy [29] (Fig. 4.7), we generated an anisotropic semi-structured mesh consisting of 2016 tetrahedra (Fig. 4.8). The mesh consisted of 42 partitions in the xy -plane with smaller elements concentrated close to the step to resolve the large gradients at the sharp corner, and larger elements located toward the outlet. In the z -direction, there were eight partitions with smaller elements located in the side wall boundary layer and larger elements laid close to the symmetry plane. This yielded highly stretched elements with aspect ratios as small as 0.08.

We used $\Delta t = 2 \times 10^{-3}$, and absolute tolerances of 10^{-8} and 10^{-7} for the preconditioned conjugate gradient iterations of the velocity and pressure calculations, respectively. Starting with initial conditions corresponding to a lower Reynolds number, the simulation typically ran for four to five flow-through times until the L^2 norm of the difference between consecutive velocity solutions and between consecutive pressure solutions both dropped below 5×10^{-8} .

We identified the length of the primary recirculation zone x_1 on the symmetry plane

based on the Prandtl criteria [94]:

$$\frac{\partial u}{\partial y}\Big|_{x_1} = 0. \quad (4.6)$$

An alternative approach to measure the recirculation length is to draw the surface shear stress (limiting) streamlines and identify the saddle points. However, as shown by Biswas et al. [14], these two approaches yield almost identical results so long the measurement is performed on a plane far from the side wall (such as the symmetry plane). The measured recirculation length along with results of three other studies, namely the experiments of Armaly et al. [3], two-dimensional (2D) simulations of Biswas et al. [14] and the 3D spectral element simulations of Couzy [29], are plotted in Fig. 4.9. At both Reynolds numbers, our results are in very good agreement with the other data. Specifically, for $Re = 172$ and 344, our calculations predict lengths of $x_0 = 2.10$ and 3.62 respectively, yielding approximately 1% errors when compared with the corresponding lengths determined experimentally.

In the following subsections, we give more details of the simulated flow at $Re = 172$ and 344 and compare our results with those of spectral element simulations of Couzy [29]. Couzy used a mesh consisting of 128 hexahedrons (Fig. 4.7). He reported resolution independent results for order $k = 9$, corresponding to a total number of grid points of 128000. This is approximately twice the total grid points (70560) used in our simulations.

4.3.1 $Re = 172$

To understand the flow patterns near the recirculation zone, we present contour plots of the streamwise velocity on the symmetry plane $z = 9$ (Fig. 4.10a) and of the streamwise and spanwise velocity components at the plane $x = 1.0$ (Fig. 4.10b and c, respectively). The negative velocity region in Fig. 4.10a shows the retrograde flow zone, which is an indication of the recirculation region. Non-negligible spanwise velocities close to the side wall boundary layers are observed in Fig. 4.10c, confirming the three-dimensionality of

the flow. These contour plots are very similar to those reported in [29] (Fig. 4.11). Both simulations predicted negative velocity regions near $z = 0$ as in Fig. 4.10b and Fig. 4.10c. Note that in Fig. 4.10c, the negative velocity contours (with absolute values as large as 10^{-3}) close to the symmetry plane are due to the weak imposition of the symmetry boundary condition.

We also plotted the streamwise velocity downstream of the step along four observation lines in the lower and upper parts of the geometry (Fig. 4.12) as well as the spanwise velocity along two observation lines (Fig. 4.13). The corresponding streamwise velocity profiles from the spectral element simulations of Couzy are also shown in Fig. 4.12. Couzy's results in 4.12 are based on digitization of data presented in his thesis. Overall, there is very good agreement between our results and those of Couzy, although slight differences along two observation lines (at $x = 5.75$) are noticeable. These differences may be due to the fact that we used less resolution than Couzy did. Specifically, the fact that better agreement was obtained along the lines close to the step compared to those further downstream of the step might suggest that while the resolution used close to the step is sufficient, higher resolution is required further downstream of the step. As is clear from Fig. 4.12, the velocity profiles have smaller maximum values on planes at higher values of x , suggesting the rapid development of the flow downstream of the recirculation zone. Fig. 4.13 specifically shows non-negligible values of the spanwise velocity even outside the side wall boundary layer.

4.3.2 $Re = 344$

We present contour plots of the streamwise velocity on the symmetry plane $z = 9$ (Fig. 4.14a) and of the streamwise and spanwise velocity components on the plane $x = 2.0$ (Fig. 4.14b and c, respectively). Similar observations as those in the $Re = 172$ case can be made. The negative flow region in Fig. 4.14a indicates a recirculation zone. Non-negligible spanwise velocity close to the side wall boundary layers is observed in Fig. 4.14c, con-

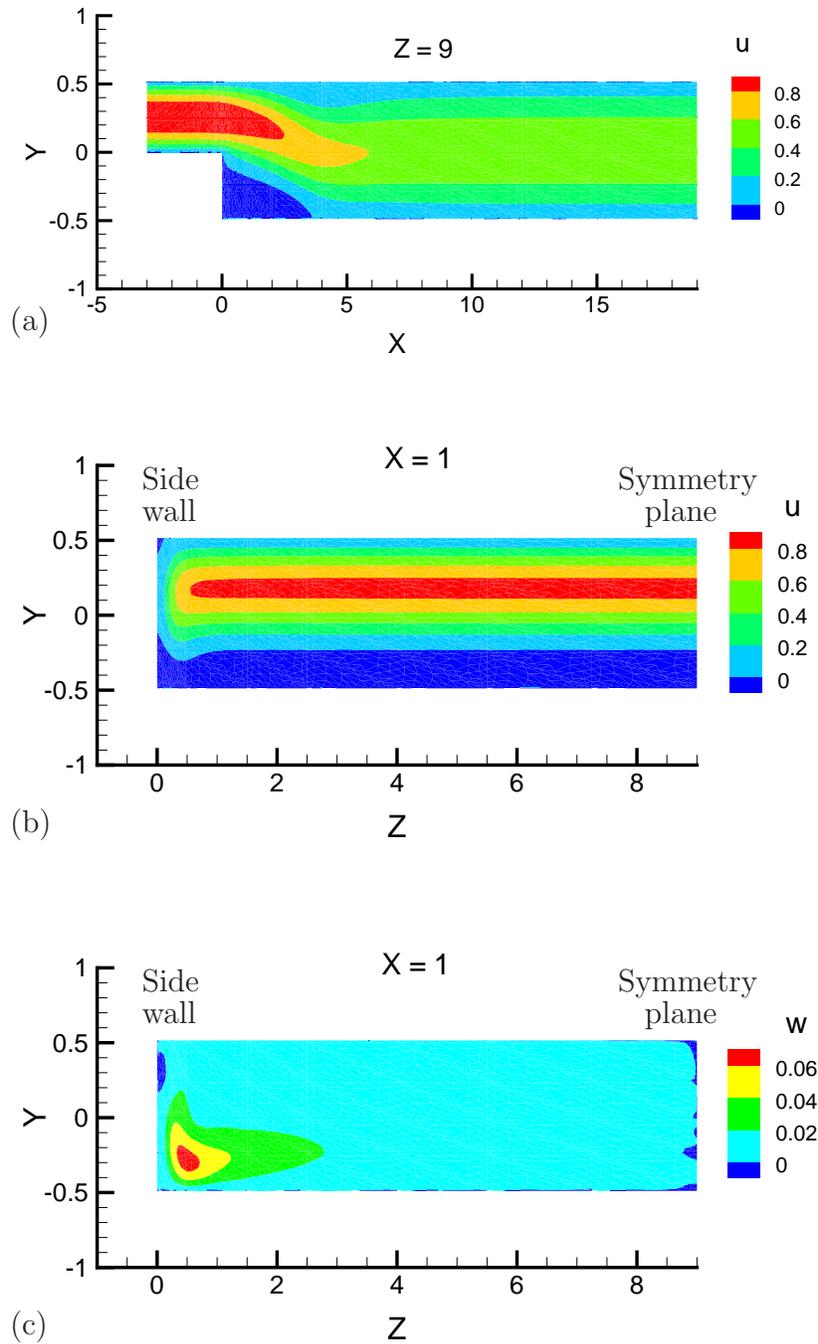


Figure 4.10: Velocity contours for the backward-facing step flow at $Re = 172$, calculated using the DG method; (a) streamwise velocity component on the symmetry plane, with the dark blue region indicating the retrograde flow zone; (b) streamwise velocity component at $x = 1.0$, which is approximately half-way through the recirculation zone; (c) spanwise velocity component at $x = 1.0$. In figure (a), the y -axis was enlarged six-fold, and in figures (b) and (c) it was enlarged two-and-half-fold.

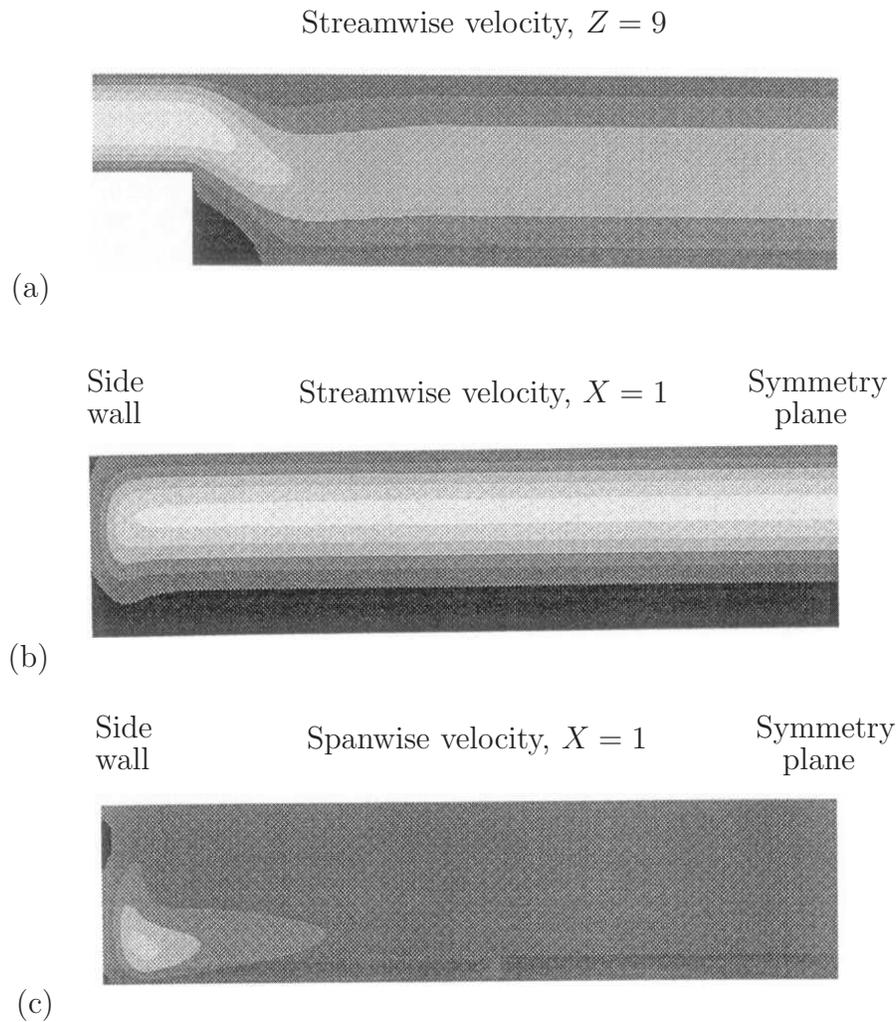


Figure 4.11: Velocity contours for the backward-facing step flow at $Re = 172$, calculated using the spectral element method as reported by Couzy [29]; (a) streamwise velocity component on the symmetry plane, with the black region indicating the retrograde flow zone; (b) streamwise velocity component at $x = 1.0$, which is approximately half-way through the recirculation zone; (c) spanwise velocity component at $x = 1.0$. The consecutive shades of gray indicate a velocity increase of 0.2 each with the black zones representing negative velocity values. The resolution of the figures are limited by the resolution of the source.

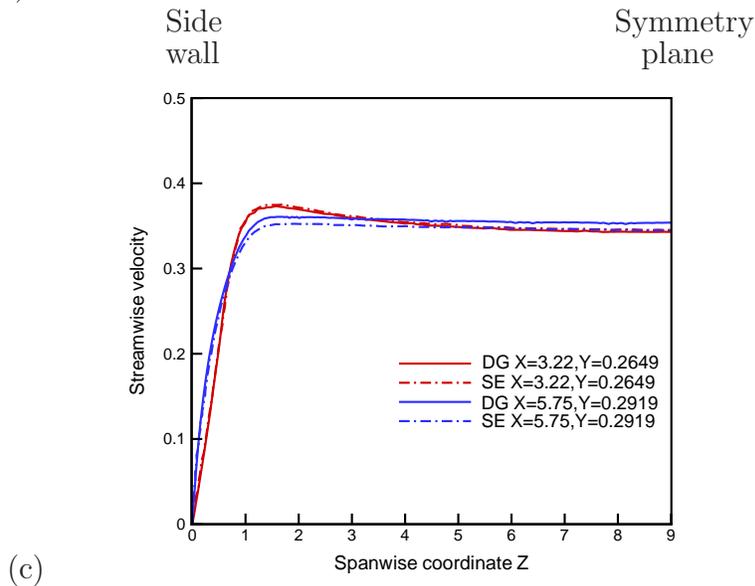
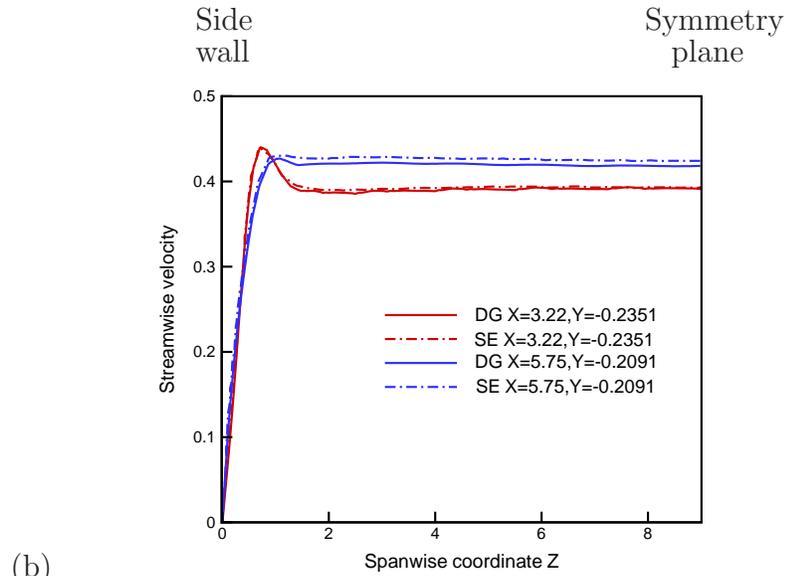
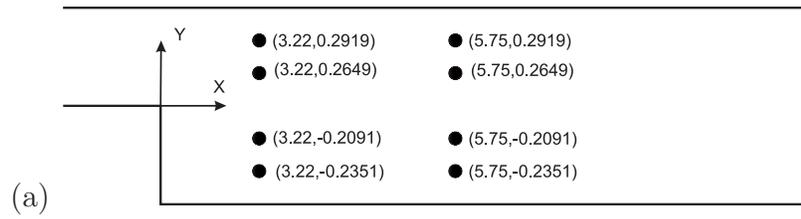


Figure 4.12: Streamwise velocity components at $Re = 172$. (a) Location of observation lines for panels (b) and (c); (b) velocities along two observation lines in the lower part of the geometry of the backward-facing step problem; (c) velocities along two observation lines in the upper part of the geometry. The DG results are given along with the spectral element (SE) results of Couzy [29].

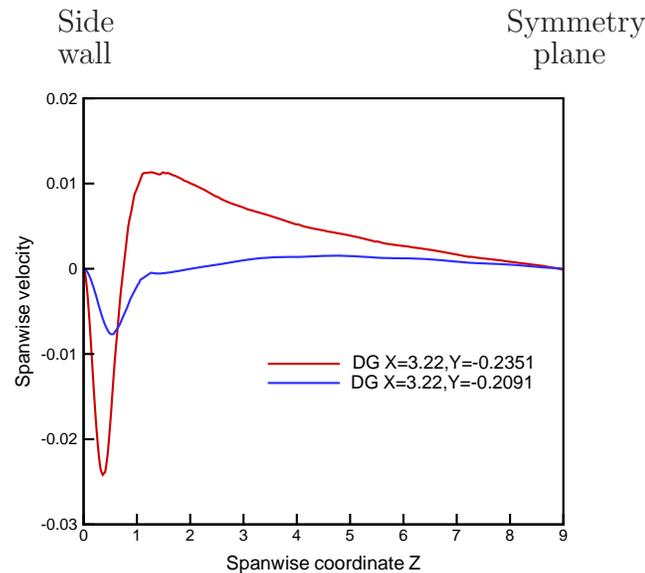


Figure 4.13: The spanwise velocity components along two observation lines in the lower part of the geometry of the backward-facing step problem at $Re = 172$.

firming the three-dimensionality of the flow. These contour plots match well with those reported in [29] (Fig. 4.15), even in some small details such as the negative velocity regions in Fig. 4.14b and that in Fig. 4.14c near $z = 0$. Some negative velocity contours (with absolute values as large as 10^{-3}) close to the symmetry plane in Fig. 4.14c are observed.

We also plotted the streamwise velocity downstream of the step along six observation lines in the lower and upper parts of geometry (Fig. 4.16) as well as the spanwise velocity along three observation lines (Fig. 4.17). The corresponding streamwise velocity profiles from the spectral element simulations of Couzy are also plotted in Figs. 4.16 and 4.17. The overall behaviors are again very similar, but slight differences along some observation lines are noticeable. We again believe the lower resolution in our calculations contributes to these variations. Analogous to the former case, from Figs. 4.16 and 4.17, we learn that flow develops rapidly downstream of the recirculation zone. Also, it is clear from Fig. 4.17, that the three-dimensionality of flow is not limited to the boundary layer, and it is more pronounced than that at the lower $Re = 172$.

4.4 Summary

Overall the developed solver performed very well in solving the three-dimensional problems. In the first problem test, the theoretical convergence rates were obtained in solving the three-dimensional Taylor vortex problem. In solving the Orr-Sommerfeld stability problem at $Re = 1500$, the perturbation energy growth rate was predicted highly accurately, up to four significant digits using a mesh consisting of 1536 tetrahedra of order $k = 6$. Finally, in simulating backward-facing step flow, excellent agreement with other computational and experimental data was obtained in calculating the length of the primary recirculation zone. On the same test, in comparison with a spectral element Navier-Stokes solver [29], similar accuracy, at lower cost, was obtained in capturing even fine features of flow such as small retrograde flow regions inside the side wall boundary layer. This last test implies that our solver based on the DG scheme is competitive with the spectral element solver of Couzy [29] in both accuracy and efficiency.

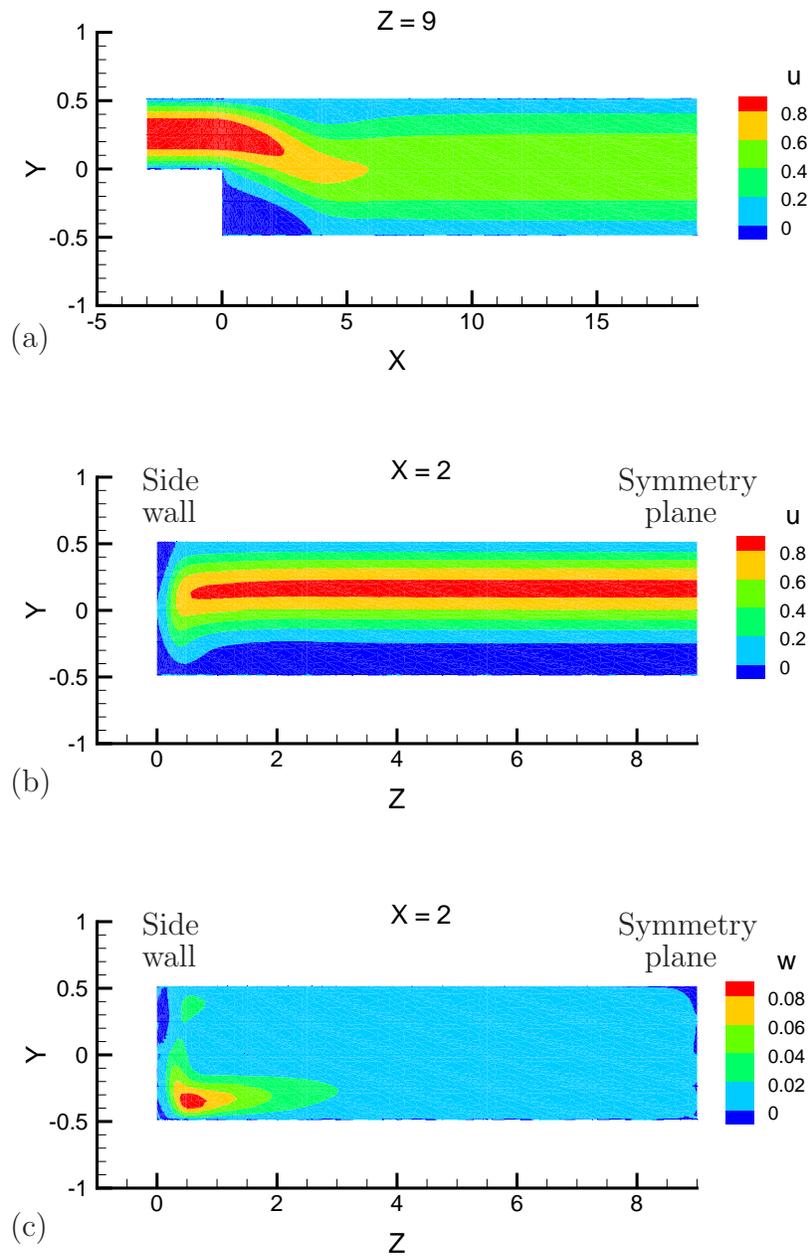


Figure 4.14: Velocity contours for the backward-facing step flow at $Re = 344$; (a) streamwise velocity component at the symmetry plane with dark blue region indicating the recirculation zone; (b) streamwise velocity component at $x = 2.0$ approximately half-way the recirculation zone; (c) spanwise velocity component at $x = 2.0$. In figure (a), the y -axis was enlarged six-fold, and in figures (b) and (c) it was enlarged two-and-half-fold.

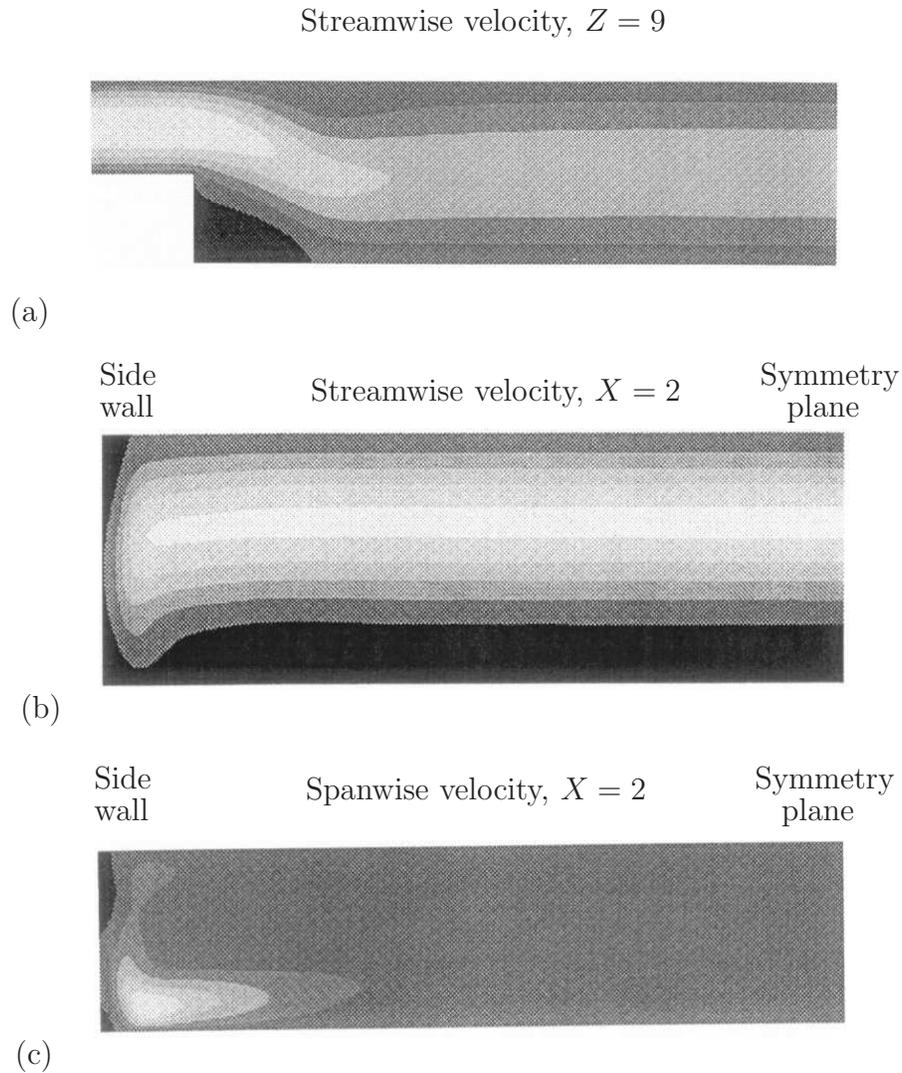
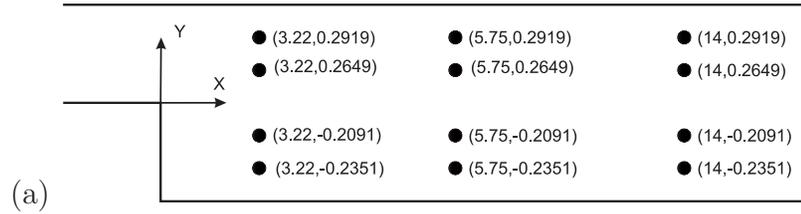
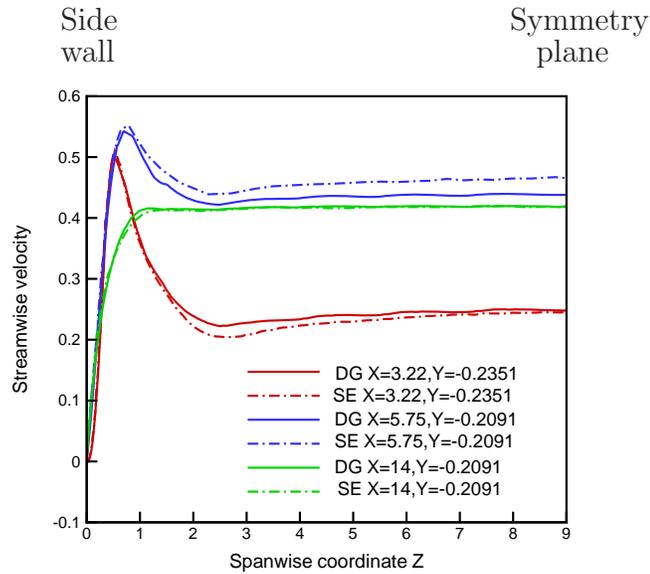


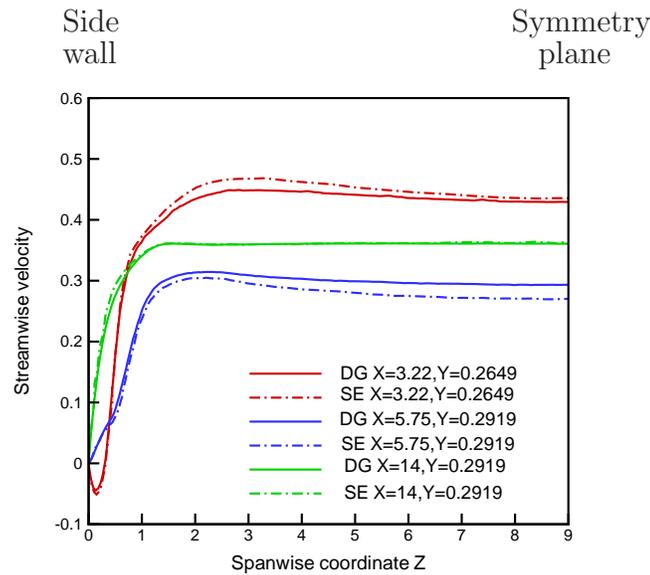
Figure 4.15: Velocity contours for the backward-facing step flow at $Re = 344$, calculated using the spectral element method as reported by Couzy [29]; (a) streamwise velocity component on the symmetry plane, with the black region indicating the recirculation zone; (b) streamwise velocity component at $x = 2.0$, which is approximately half-way through the recirculation zone; (c) spanwise velocity component at $x = 2.0$. The consecutive shades of gray indicate a velocity increase of 0.2 each with the black zones representing negative velocity values. The resolution of the figures are limited to the resolution of the source.



(a)



(b)



(c)

Figure 4.16: Streamwise velocity components at $Re = 344$. (a) Location of observation lines for panels (b) and (c); (b) velocities along three observation lines in the lower part of the geometry of the backward-facing step problem; (c) velocities along three observation lines in the upper part of the geometry. The DG results are given along with the spectral element (SE) results of Couzy [29].

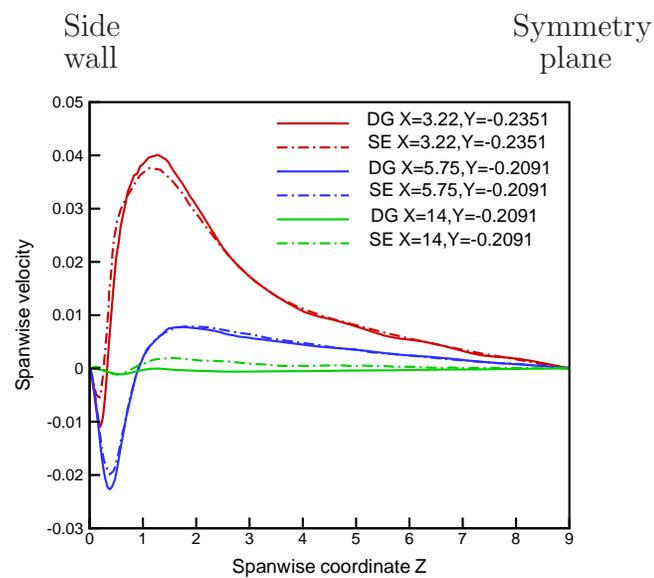


Figure 4.17: Spanwise velocity components along three observation lines in the lower part of the geometry of the backward-facing step problem at $Re = 344$. The DG results are given along with the spectral element (SE) results of Couzy [29].

Chapter 5

An Explicit Expression for the Penalty Parameter

In this chapter, we derive an explicit expression for the penalty parameter of the interior penalty method for the elliptic problems. The expression yields a coercive bilinear form and is valid for general meshes consisting of (geometrically nonconforming) simplicial elements.

5.1 Introduction

The interior penalty (IP) method devised in the late 1970s [4] is a type of discontinuous Galerkin method for the spatial discretization of elliptic partial differential equations. The IP method, like other discontinuous Galerkin methods, has advantages over the classical continuous Galerkin method in facilitating hp-adaptivity and yielding block diagonal mass matrices important in time-dependent problems. Moreover, the IP method gives a symmetric, locally conservative, and small-stencil discretization. This last property, in which the degrees of freedom of each element couple only with those of its immediate neighbors, is critical in reducing memory requirements and achieving efficient parallelization in large scale computations.

Despite its early introduction and its advantages, the IP method has not been popular. One drawback to this scheme is that it requires the user to specify a mesh-dependent parameter, known as a penalty parameter. If the value of this parameter is not sufficiently large, the approximate solution is unstable¹. On the other hand, an arbitrarily large value of the penalty parameter degrades the performance of the iterative solver of the linear system arising from the IP discretization, as shown in Section 3.2. In real applications, where highly anisotropic and heterogeneous mesh geometries are used and in adaptive algorithms, where varying approximation orders are also used, it is difficult to know *a priori* the minimum acceptable value of the penalty parameter. Therefore, the objective of this paper is to derive an explicit expression for the value of the penalty parameter guaranteed to give a stable solution. We consider a domain partitioned into triangular or tetrahedral elements in two or three space dimensions.

Before deriving this expression, we briefly describe the IP method and show the effect of the penalty parameter on the overall efficiency of the scheme.

5.2 Interior Penalty Method

We seek the IP formulation of the Poisson equation with Dirichlet boundary conditions:

$$-\Delta u = f \quad \text{in } \Omega, \quad (5.1a)$$

$$u = g \quad \text{on } \partial\Omega, \quad (5.1b)$$

where Ω is a polygonal domain of dimension $d = 1, 2$, or 3 .

We first introduce some notation. Let K^+ and K^- be two adjacent elements in \mathcal{T}_h , a triangulation of Ω ; let \mathbf{x} be an arbitrary point of the interior set $e = \partial K^- \cap \partial K^+$, which

¹Here instability is a direct consequence of non-coercivity of the associated bilinear form. Its characteristic is that small variations in the penalty parameter yield large variations in the field variable, as will be seen below.

is assumed to have non-zero dimension $(d - 1)$ and is referred to as a face; and let \mathbf{n}^- and \mathbf{n}^+ be the corresponding normal vectors at that point. Let u be a smooth function inside each element K^\pm and let us denote by u^\pm the trace of u on e from the interior of K^\pm . Then we define the mean $\{\cdot\}$ and the jump $[[\cdot]]$ at $\mathbf{x} \in e$ as

$$\{u\} := (u^+ + u^-)/2, \quad [[u]] := u^+ \mathbf{n}^+ + u^- \mathbf{n}^-.$$

For a point \mathbf{x} on the boundary set $\partial K \cap \partial\Omega$ with normal vector \mathbf{n} , we define the trace operators as

$$\{u\} := u, \quad [[u]] := u\mathbf{n}.$$

The mean of a vector-valued function is defined in a similar way. We also denote by Γ_I the union of all interior sets e , and we set $\Gamma = \Gamma_I \cup \partial\Omega$.

We take the discontinuous approximation to the exact solution u , u_h , in the finite element space V_h , where

$$V_h := \{v \in L^2(\Omega) | v|_K \in Q_{k_K}(K), \forall K \in \mathcal{T}_h\}.$$

Here $Q_{k_K}(K)$ is the set of polynomials of degree at most k_K on K , $k \geq 0$. The approximate solution is then defined by requiring that

$$a(u_h, v_h) = F(v_h) \quad \forall u_h, v_h \in V_h.$$

where

$$a(u, v) = \sum_K \int_K \nabla u \cdot \nabla v d\mathbf{x} - \int_\Gamma ([[v]] \cdot \{\nabla u\} + [[u]] \cdot \{\nabla v\}) ds + \int_\Gamma \mu [[u]] \cdot [[v]] ds, \quad (5.2a)$$

$$F(v) = \int_\Omega f v d\mathbf{x} - \int_{\partial\Omega} g \nabla v \cdot \mathbf{n} ds + \int_{\partial\Omega} \mu g v ds. \quad (5.2b)$$

The last term on the r.h.s of equation 5.2a is defined for interior and boundary faces, Γ . This is the penalty term, which is added to enforce the coercivity of the bilinear form

$a(u, v)$. We must specify a value for the penalty parameter μ that ensures the coercivity of the bilinear form and, thus, the stability of the approximate solution. In previous work, μ has only been defined to within a multiplicative constant. For example, Arnold [4] defined $\mu = \gamma_0/l_e$, where $l_e = \text{diam}(e)$, and γ_0 is a large unknown positive constant. In the context of the mixed hp-discontinuous Galerkin finite element method, Schötzau et al. [102] defined $\mu = \eta h^{-1}k^2$, where $h = \min(h_{K^+}, h_{K^-})$, $k = \max(k_{K^+}, k_{K^-})$, and $h_K = \text{diam}(K)$, again leaving η as a large unknown positive constant.

Unfortunately, the above expressions for μ are less than optimal in practice, because a large value of μ has a detrimental effect on the conditioning of the matrix that represents the bilinear form $a(u, v)$. As proved by Castillo [20] for all approximating polynomial degrees, the spectral condition number of this matrix in L^2 norm grows linearly with μ (see Theorem 3.4 in [20]). It is therefore expected that the magnitude of μ will affect the overall efficiency of the iterative solver of the system arising from the IP discretization.

To further investigate this, we conducted the following experiment. Using the nodal high order IP method, we discretized eq. 5.1 with $g = 0$ and $f = 2\pi^2 \sin(\pi x) \sin(\pi y)$, on the square domain $[-1, 1] \times [-1, 1]$. The corresponding exact solution is $u = \sin(\pi x) \sin(\pi y)$. Our nodal basis was the Lagrange polynomials calculated based on the nodal set of Hesthaven [57] defined on the standard triangle. The domain was partitioned once into 72 structured triangles and once into 72 unstructured (heterogeneous) triangles as shown in Figs. 5.1a and b, respectively. To solve the resulting linear system, we used the preconditioned conjugate gradient method. The preconditioner was a two-level element-based Schwarz preconditioner. Its local part corresponded to the IP discretization on each element, similar to that of Feng and Karakashian [39], and its global coarse part corresponded to the IP discretization on the same mesh but with the lower approximation order $k = 1$. (A similar preconditioner will be discussed in chapter 7.) Our implementation was based on the Algorithm Oriented Mesh Database (AOMD) [98] and Portable, Extensible Toolkit for Scientific Computing (PETSc) [8, 7, 9]. We carried out simulations

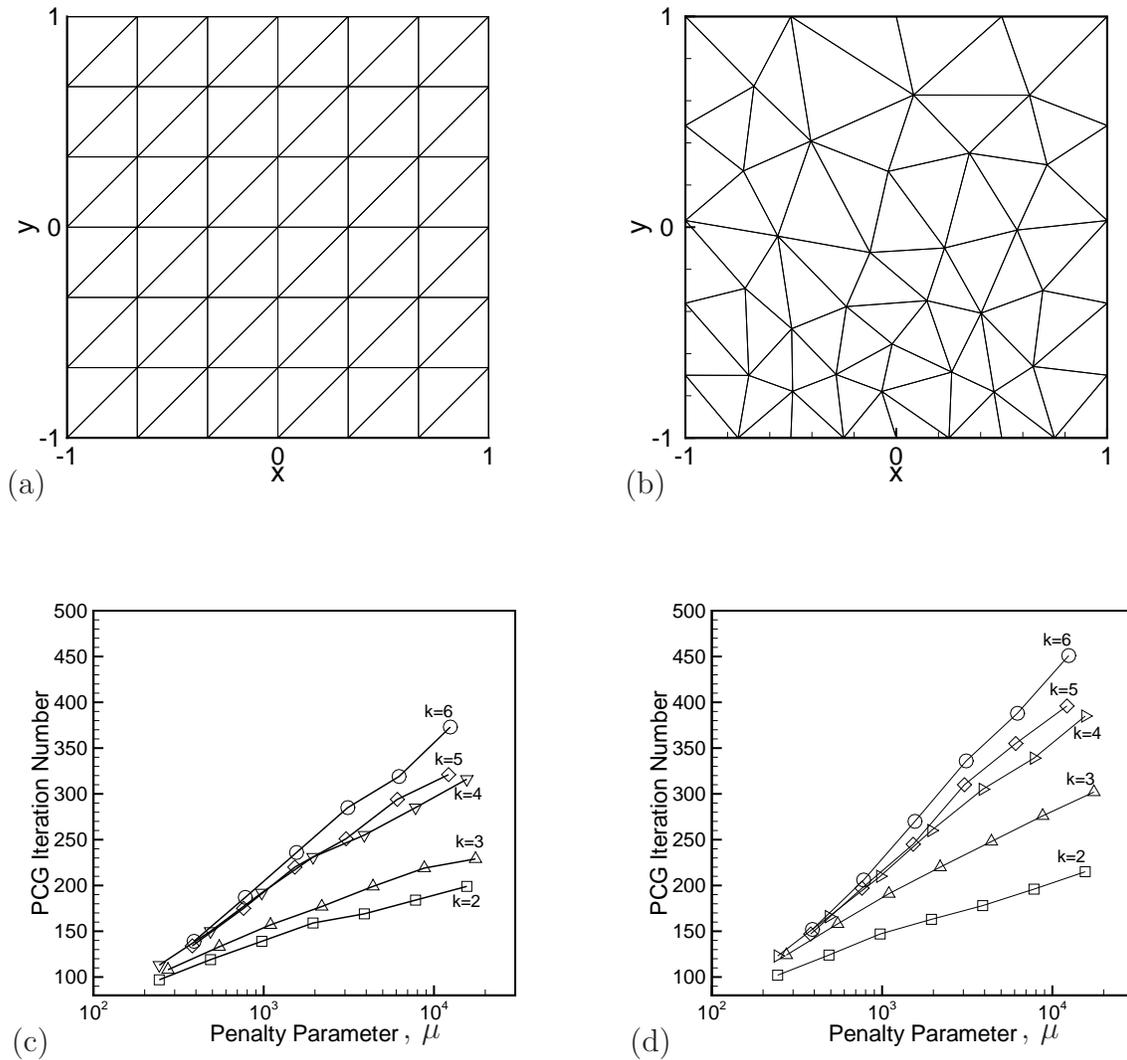


Figure 5.1: (a) The computational domain used for tests partitioned into 72 structured triangular elements; (b) the same domain partitioned into 72 unstructured (heterogeneous) triangular elements generated using Gmsh software [46]; (c) and (d) the number of preconditioned conjugate gradient (PCG) iterations needed to solve the Poisson problem vs. the penalty parameter μ using meshes in (a) and (b), respectively. The nodal high order IP method was used, with the range of approximation orders ($k = 2, \dots, 6$).

using different values of μ and with the range of approximation orders ($k = 2, \dots, 6$). The initial guess for the conjugate gradient iterations was a vector with random entries confined to the interval $[0, 1]$ and the stopping criterion was a relative residual smaller than 10^{-11} . The results are shown in Figs. 5.1c and d for the structured and unstructured meshes, respectively. Although the iteration counts are higher for the unstructured mesh, it is clear that the iteration counts in both cases grow almost logarithmically with μ , implying that arbitrarily large values of μ yield unacceptably large iteration counts. It is therefore clear that an explicit expression for the penalty parameter would be useful, so as to guarantee coercivity while minimizing computational expense.

5.3 Explicit Expression for the Penalty Parameter

Here we derive an explicit expression for the penalty parameter μ for a d -dimensional simplex. Our derivation is based on the results of Warburton and Hesthaven [118] on trace inverse inequalities. Using orthogonal polynomials, they proved the following inequality for a simplicial element K and $\forall v \in Q_k(K)$:

$$\int_e v^2 ds \leq \frac{(k+1)(k+d)}{d} \frac{\mathcal{A}(e)}{\mathcal{V}(K)} \int_K v^2 d\mathbf{x}, \quad (5.3)$$

where for $d = 3$, \mathcal{A} and \mathcal{V} denote area and volume, respectively, and for $d = 2$, they denote length and area, respectively. For $d = 1$, $\mathcal{A}(\cdot) = 1$ and \mathcal{V} denotes length. (See theorems 2, 3, 4 in [118]).

We must find μ such that the bilinear form $a(u, v)$ is coercive, i.e. so that there exists a positive constant c_s such that

$$a(v, v) \geq c_s \|v\|_h^2 \quad \forall v \in V_h, \quad (5.4)$$

where

$$a(v, v) = \sum_K \int_K (\nabla v)^2 d\mathbf{x} - 2 \int_{\Gamma} \llbracket v \rrbracket \cdot \{\nabla v\} ds + \int_{\Gamma} \mu \llbracket v \rrbracket^2 ds, \quad (5.5)$$

and

$$\|v\|_h^2 = \sum_K |v|_{1,K}^2 + \int_{\Gamma} \llbracket v \rrbracket^2 ds,$$

with the seminorm $|\cdot|_{1,K}$ defined over $H^1(K)$ by

$$|v|_{1,K}^2 = \int_K (\nabla v)^2 d\mathbf{x}.$$

We first find a bound on the negative term on the r.h.s. of equation 5.5. Using the arithmetic-geometric mean inequality $ab \leq (\epsilon_e/2)a^2 + (1/2\epsilon_e)b^2$ with $\epsilon_e > 0$ yields

$$\int_e \llbracket v \rrbracket \cdot \{\nabla v\} ds \leq \frac{\epsilon_e}{2} \int_e \llbracket v \rrbracket^2 ds + \frac{1}{2\epsilon_e} \int_e \{\nabla v\}^2 ds \quad \forall e \in \Gamma.$$

Adding the above inequality over all e , noting that on Γ_I , $\{\nabla v\}^2 = (\nabla v^+)^2/4 + (\nabla v^-)^2/4 + (\nabla v^+ \cdot \nabla v^-)/2$ and on $\partial\Omega$, $\{\nabla v\}^2 = (\nabla v)^2$, and using the inequality $a^2 + b^2 + 2ab \leq 2a^2 + 2b^2$ yields

$$\begin{aligned} \int_{\Gamma} \llbracket v \rrbracket \cdot \{\nabla v\} ds &\leq \sum_{e \in \Gamma} \frac{\epsilon_e}{2} \int_e \llbracket v \rrbracket^2 ds + \sum_{e \in \Gamma_I} \frac{1}{2\epsilon_e} \int_e \left[\frac{1}{2} (\nabla v^+)^2 + \frac{1}{2} (\nabla v^-)^2 \right] ds \\ &\quad + \sum_{e \in \partial\Omega} \frac{1}{2\epsilon_e} \int_e (\nabla v)^2 ds. \end{aligned}$$

Substituting the above inequality in equation 5.5, and then using equation 5.3 yields

$$a(v, v) \geq \sum_K \sum_{i_e=1}^n \left(\frac{c_{e,K}}{c_K} - \frac{c_{e,K}}{\epsilon_e} \right) \int_K (\nabla v)^2 d\mathbf{x} + \sum_{e \in \Gamma} \int_e (\mu - \epsilon_e) \llbracket v \rrbracket^2 ds, \quad (5.6)$$

where

$$c_{e,K} = \begin{cases} \frac{(k_K+1)(k_K+d)}{d} \frac{\mathcal{A}(e)}{\mathcal{V}(K)} & e \in \partial\Omega, \\ \frac{(k_K+1)(k_K+d)}{d} \frac{\mathcal{A}(e)/2}{\mathcal{V}(K)} & e \in \Gamma_I, \end{cases}$$

$$c_K = \frac{(k_K+1)(k_K+d)}{d} \frac{[\mathcal{A}(\partial K \setminus \partial\Omega)/2 + \mathcal{A}(\partial K \cap \partial\Omega)]}{\mathcal{V}(K)} \quad K \in \mathcal{T}_h. \quad (5.7)$$

In equation 5.6, i_e denotes the local index of face e restricted to the element K , and n denotes the total number of faces of each element K . To yield a positive r.h.s in equation 5.6, it is sufficient to choose $\epsilon_e \geq c_K$ for $e \in \partial\Omega$, $\epsilon_e \geq \max(c_{K^+}, c_{K^-})$ for $e \in \Gamma_I$, and

$\mu \geq \epsilon_e$. Thus, we choose the local penalty parameter μ_e as

$$\mu_e = c_K \quad \forall e \in \partial\Omega, \quad (5.8a)$$

$$\mu_e = \max(c_{K^+}, c_{K^-}) \quad \forall e \in \Gamma_I. \quad (5.8b)$$

Now, if we set

$$c_1 = \min_{K,e} \left(\frac{c_{e,K}}{c_K} - \frac{c_{e,K}}{\epsilon_e} \right) \quad \forall e \in \Gamma, K \in \mathcal{T}_h,$$

$$c_2 = \min_e (\mu - \epsilon_e) \quad \forall e \in \Gamma,$$

we obtain

$$a(v, v) \geq c_1 \sum_K \int_K (\nabla v)^2 d\mathbf{x} + c_2 \int_\Gamma \llbracket v \rrbracket^2 ds,$$

Finally, if we choose $c_s = \min(c_1, c_2)$, the coercivity of the bilinear form, 5.4, results.

Remark 1. The penalty expression in equation 5.8 is defined for each face $e \in \Gamma$ and depends on the geometries and approximating polynomial orders within the elements sharing e . Obviously, a global bound for the penalty parameter can be derived as

$$\mu = \max_e (\mu_e). \quad (5.9)$$

Remark 2. Our estimation in equation 5.9 is sharp, which we demonstrate by the following numerical experiment. By once again solving the Poisson equation 5.1 using the methodology of Section 3.2 and the structured triangular mesh in Fig. 5.1a, we computed the maximum nodal error versus μ for polynomial approximation orders $k = 1, \dots, 8$ (Fig. 5.2). It can be seen that the solution is unstable for $\mu < \mu^*$ (i.e., a small variation in the penalty parameter yields a large variation in the field variable u) and stable for $\mu > \mu^*$ (i.e., variations in μ yields almost no variations in u). The critical values of penalty parameter μ^* are approximately 10, 50, 50, 90, 90, 150, 200, and 200 for $k = 1, \dots, 8$, respectively. On the same figure, we also show the value of the penalty parameter computed using equation 5.9. We observe that the estimation based on 5.9 yields a stable solution and it is roughly three times larger than μ^* at each k . Based on the results of Fig. 5.1c,

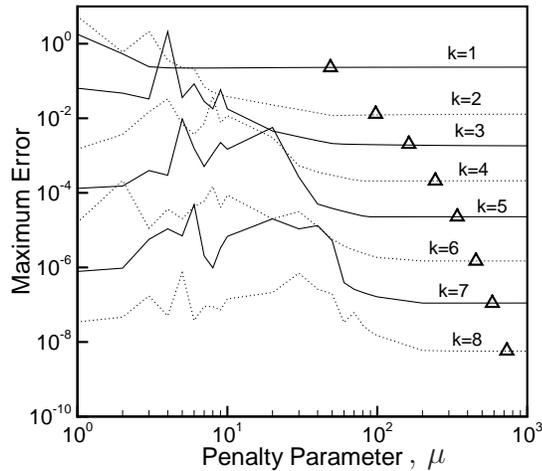


Figure 5.2: Maximum nodal error vs. penalty parameter μ for different orders of approximation $k = 1, \dots, 8$; the triangles represent the values of the penalty parameter calculated using equation 5.9.

this means that selecting μ according to equation 5.9 guarantees a stable solution at a computational cost within a factor of roughly 1.2 of that of the (unknown) optimal penalty parameter, μ^* , for the higher approximation orders ($k = 6, 7$, and 8), at least for the mesh used in this study.

Remark 3. For the case of a general mesh, when the elements are not face-to-face including those with hanging nodes, an explicit expression for the penalty parameter for a face e_f shared by a collection of adjacent elements $\{K_i | i = 1, \dots, N\}$ can be defined as

$$\mu_{e_f} = \max(c_{K_i}), \quad i = 1, \dots, N, \quad (5.10)$$

where c_{K_i} is computed using equation 5.7. Following the above procedure, it is proved that choosing the penalty parameter based on equation 5.10 guarantees a coercive bilinear form for general meshes.

Remark 4. Through a similar procedure, an explicit expression for the penalty parameter of the discontinuous Galerkin method of Baker [6] can be derived. The same

formulas 5.8 and 5.10 are valid in this method, but with a slightly different c_K

$$c_K = \frac{(k_K + 1)(k_K + d)}{d} \frac{\mathcal{A}(\partial K)}{\mathcal{V}(K)} \quad K \in \mathcal{T}_h.$$

Chapter 6

Implementation and Performance

Study

In chapter 2, we described the numerical implementation of integrals and derivatives arising from the discontinuous Galerkin scheme. We also presented quadrature schemes for evaluating the nonlinear term. We now present the strategy for solving the linear systems arising from the discontinuous Galerkin discretization of the velocity and pressure equations, further implementation and coding details and profiling studies demonstrating the performance of the developed solver.

6.1 Linear System Solves

For the velocity and pressure we must solve a system of the form

$$A\underline{x}^n = \underline{b}^n \quad n = 1, 2, \dots, \quad (6.1)$$

where A is the coefficient matrix, \underline{b}^n the right-hand-side vector, \underline{x}^n is the approximate solution vector and n is the time step number. Since A is typically large and very sparse, iterative methods are suitable for solving the above system and since A is positive definite the conjugate gradient method is a natural choice. To accelerate the convergence of the

conjugate gradient method, instead of solving the system 6.1 directly, the equivalent system

$$B^{-1}A\underline{x}^n = B^{-1}\underline{b}^n \quad n = 1, 2, \dots \quad (6.2)$$

is solved. 6.2 is called the preconditioned system corresponding to 6.1 and the matrix B^{-1} is referred to as a preconditioner matrix. B^{-1} should closely approximate the inverse of matrix A and should be easily computable. For the velocity systems, we use a preconditioner corresponding to the inverse of the block diagonal mass matrix. For the pressure equation, an incomplete LU (ILU) factorization preconditioner is a convenient choice. The ILU factorization process consists of forming a sparse lower triangular matrix L and a sparse upper triangular matrix U such that the residual matrix $R = LU - A$ satisfies a certain constraint. For instance, one constraint is that the matrix R has zero entries in locations where A has nonzero entries, yielding an ILU factorization with no fill-in (ILU(0)). We use the serial version of ILU(0) factorization corresponding to the factorization of nonoverlapping blocks of the matrix A owned by each processor.

As shown by numerical examples in the following section, a mass matrix preconditioner for the velocity equations yields very small iteration numbers. However, the pressure solve with the ILU(0) preconditioner requires a large number of iterations, implying the suboptimality of this preconditioner. Thus, to reduce iteration numbers, we need to either use a better preconditioner or find a suitable initial guess for the conjugate gradient iterations. Although an improved preconditioner may be obtained by adding a coarse grid correction, we here adopt a simple yet effective approach, namely the projection method of Fischer [42]. In the projection method, an optimal initial guess for the conjugate gradient iterations is constructed using the previous solution vectors. Although the detail of the projection method can be found in [42], for the purpose of completeness, we briefly describe the method below.

6.1.1 Projection Technique for the Pressure System

Consider system 6.1 for the pressure. Having stored a set of previous solution vectors $\mathcal{P}_l = \{\underline{x}_i | i = 1, \dots, l\}$, we seek an approximation vector to \underline{x}^n , $\bar{\underline{x}}$, given by

$$\bar{\underline{x}} = \sum_{i=1}^l \alpha_i \underline{x}_i, \quad (6.3)$$

where α_i is defined such that the error in the A -norm,

$$\|\bar{\underline{x}} - \underline{x}^n\|_A = \left((\underline{x}^n)^T A \underline{x}^n - 2 \sum_{i=1}^l \alpha_i (\underline{x}_i)^T A \underline{x}^n + \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j (\underline{x}_i)^T A \underline{x}_j \right)^{1/2}, \quad (6.4)$$

is minimized. The minimization procedure is simplified if we insist on having an orthonormal basis, i.e.,

$$(\underline{x}_i)^T A \underline{x}_j = \delta_{ij} \quad \forall i, j = 1, \dots, l, \quad (6.5)$$

where δ_{ij} is the Kronecker delta. Applying this orthonormality relation to 6.4, then requiring that the first derivative of 6.4 with respect to α_i vanish, yields

$$\alpha_i = (\underline{x}_i)^T A \underline{x}^n = (\underline{x}_i)^T \underline{b}^n. \quad (6.6)$$

Thus, using 6.3 and 6.6, we can calculate the initial guess for the iterative solution of 6.1. After solving the system, we update the set \mathcal{P}_l for the next system solve. The update is performed through a standard Gram-Schmidt orthogonalization [49]. Specifically, if $l < L$ with L being the maximum number of vectors to be stored,

$$\beta_i = (\underline{x}_i)^T A \tilde{\underline{x}} \quad i = 1, \dots, l, \quad (6.7a)$$

$$\underline{x}_{l+1} = \left(\tilde{\underline{x}} - \sum_{i=1}^l \beta_i \underline{x}_i \right) / \left\| \tilde{\underline{x}} - \sum_{i=1}^l \beta_i \underline{x}_i \right\|_A \quad (6.7b)$$

where

$$\tilde{\underline{x}} = \underline{x}^n - \bar{\underline{x}}. \quad (6.8)$$

If $l = L$, we simply re-initialize the approximation set with $l = 1$ and $\mathcal{P}_l = \{\underline{x}_1\}$, where

$$\underline{x}_1 = \underline{x}^n / \|\underline{x}^n\|_A. \quad (6.9)$$

Using the orthonormality relation 6.5 and 6.7a, one can easily show that

$$\left\| \tilde{\underline{x}} - \sum_{i=1}^l \beta_i \underline{x}_i \right\|_A = \left(\tilde{\underline{x}}^T A \tilde{\underline{x}} - \sum_{i=1}^l \alpha_i^2 \right)^{1/2}. \quad (6.10)$$

Thus, the above update procedure requires only one matrix-vector product. The communication intensive parts of the above projection algorithm are calculation of the vector inner-products in 6.7a and 6.6. Suppose the vectors are distributed among N_P processors. Then, the vector inner-products in 6.7a can be carried out in a single $\mathcal{O}(\log_2 N_P)$ data exchange by first calculating $A\tilde{\underline{x}}$ and then passing the resultant vector along with l vectors, \underline{x}_i , to a routine that performs *multiple* vector inner-products. The vector inner-products in 6.6 are implemented in a similar fashion.

6.2 Some Coding Details

We have implemented two-dimensional and three-dimensional versions of the developed scheme in parallel using the C++ programming language, the Algorithm Oriented Mesh Database (AOMD) [98] and the Portable Extensible Toolkit for Scientific Computation (PETSc) [8, 7, 9].

We have exploited the object-oriented and data-encapsulation features of C++. In particular, for our nodal high-order discontinuous Galerkin approximation, we have designed and programmed the namespace “nodalFamily” and the class “mixedDGNodalElement”. Putting all variables and routines in the namespace nodalFamily reduces the chance of name conflict among libraries. The class mixedDGNodalElement allows the instantiation of an object with mixedDGNodalElement type by passing its dimension and the velocity and pressure approximation orders. Via an instantiation, the Vandermonde matrix V , V^{-1} , the derivative matrices and other discontinuous Galerkin operators defined on the standard elements are automatically constructed and can be accessed through the accessors provided in the class.

Each processor must read its own part of the mesh in the adopted parallel strategy, and so the mesh needs be partitioned into submeshes where each submesh is assigned to a processor. The partitioning is handled through calling the “split” program provided by the AOMD library. The “split” program internally uses the METIS library [69] to perform the partitioning. The AOMD library also provides grid adjacencies based on the algorithm needs, a (generic) class “mAttachableData” for attaching data to a mesh entity and another (generic) class “AMOD_DataExhanger” for exchanging data related to the partition boundaries.

The communication required to construct the global velocity and pressure matrices are performed through deriving a class “DGDataExhanger” from the “AMOD_DataExhanger”. All elemental data such as Jacobians, derivative metrics, and the iD of a given element having a face (in three dimensions) or an edge (in two dimensions) on the partition boundary are first packed together and then sent to a destination processor through a single message. This is performed by calling functions “AP_alloc_and_fill_buffer” and “receiveData” responsible for message packing, sending and receiving. These functions are programmed based on AUTOPACK [81], which is a library that provides several useful features such as automatic message packing, and management of send and receive requests for programs using the Message Passing Interface (MPI) [52].

Other required data exchanges are those associated with the evaluation of surface integrals in the discretization of the nonlinear term. We implemented this communication through a simple procedure. The d components of the velocity corresponding to each face (edge) on the partition boundary are first packed together and then the resulting data is sent to the destination processor. This strategy is programmed in the class “dataExhangerForNSNonlinearity” derived also from “AMOD_DataExhanger”. While the communication for constructing the global matrices is performed only once in the pre-processing stage, the communication call for the nonlinear term needs to be repeated at

each time step. Therefore, it is important to verify the efficiency of this implementation, and we will address this issue in the following section.

The global right-hand-side vectors, solution vectors and any other global auxiliary vectors are programmed based on the parallel vectors of PETSc. The global pressure and velocity matrices were based on the parallel sparse matrix type of PETSc. The conjugate gradient method and ILU(0) preconditioner were accessed through the Krylov subspace class of PETSc. The *shell preconditioner* feature of PETSc was exploited for implementing our block diagonal mass matrix preconditioner. The shell preconditioning context allows an easy interface for programming a user-defined preconditioner. Note that the required communication for matrix-vector products, vector inner-products and norm evaluations are automatically handled by PETSc.

6.3 Performance Study

Below, we first study the performance enhancement achieved by using the projection method. We then show the parallel and computational performance of the solver.

6.3.1 Performance Enhancement due to the Projection Method

We compare the performance of the code in solving the three-dimensional Orr-Sommerfeld stability problem studied in the previous chapter with and without the projection method for the pressure solve. We first ran the code without the projection method, i.e. with a zero vector as an initial guess for the pressure iterations. The runs were in parallel using 16 processors (1500 MHz Itanium 2 with 6 MBytes L3 cache) arranged in four nodes of four processors per node. We used absolute tolerances of 10^{-10} for the convergence criteria for the velocity and pressure iterations. Fig. 6.1a shows the pressure iteration numbers and summation of three velocity iteration numbers for 500 time steps. From the figure, it is clear that the number of velocity iterations are very small, verifying the

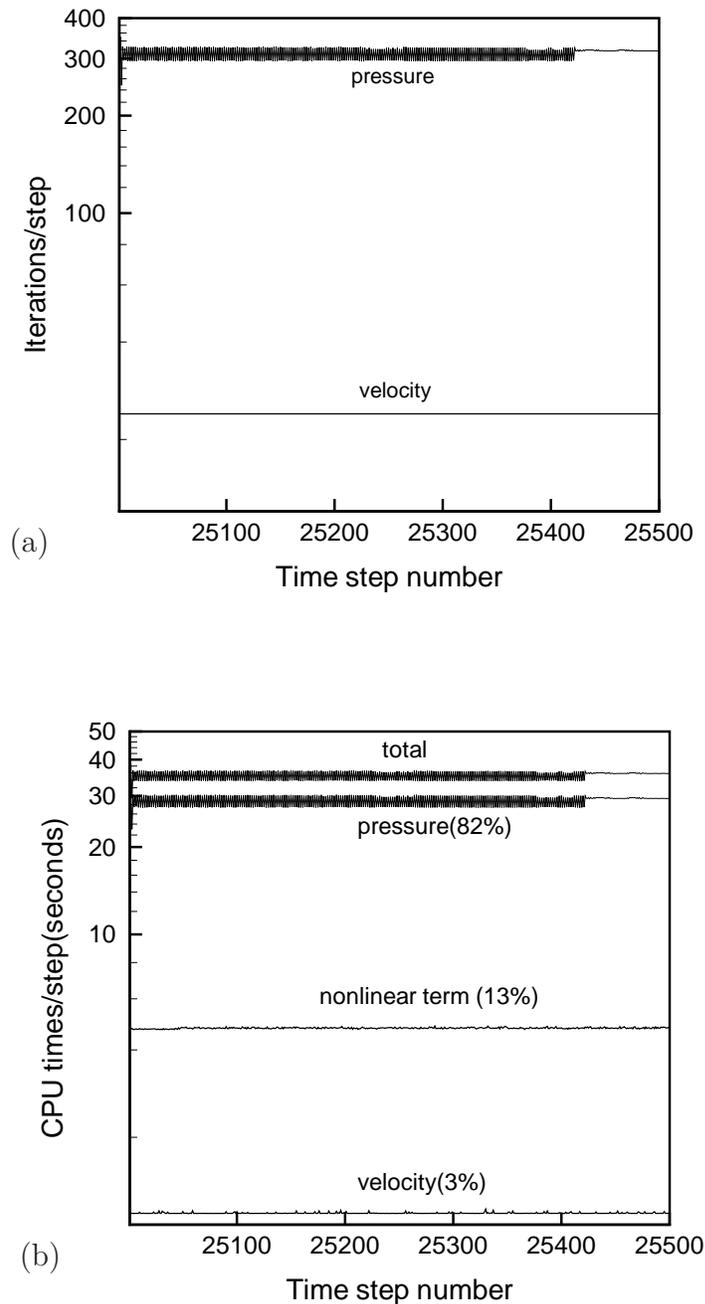


Figure 6.1: Performance of the code in solving the three-dimensional Orr-Sommerfeld stability problem *without* the projection method for the pressure equation. (a) Velocity (all three components) and pressure iteration numbers per time step vs. time step number; (b) CPU times per time step for all three velocity solves, the pressure solve, nonlinear term evaluation, and the total operations vs. the time step number. In (b), the percentage time of each individual stage is also shown. All timings were obtained using the “PetscGetTime” function.

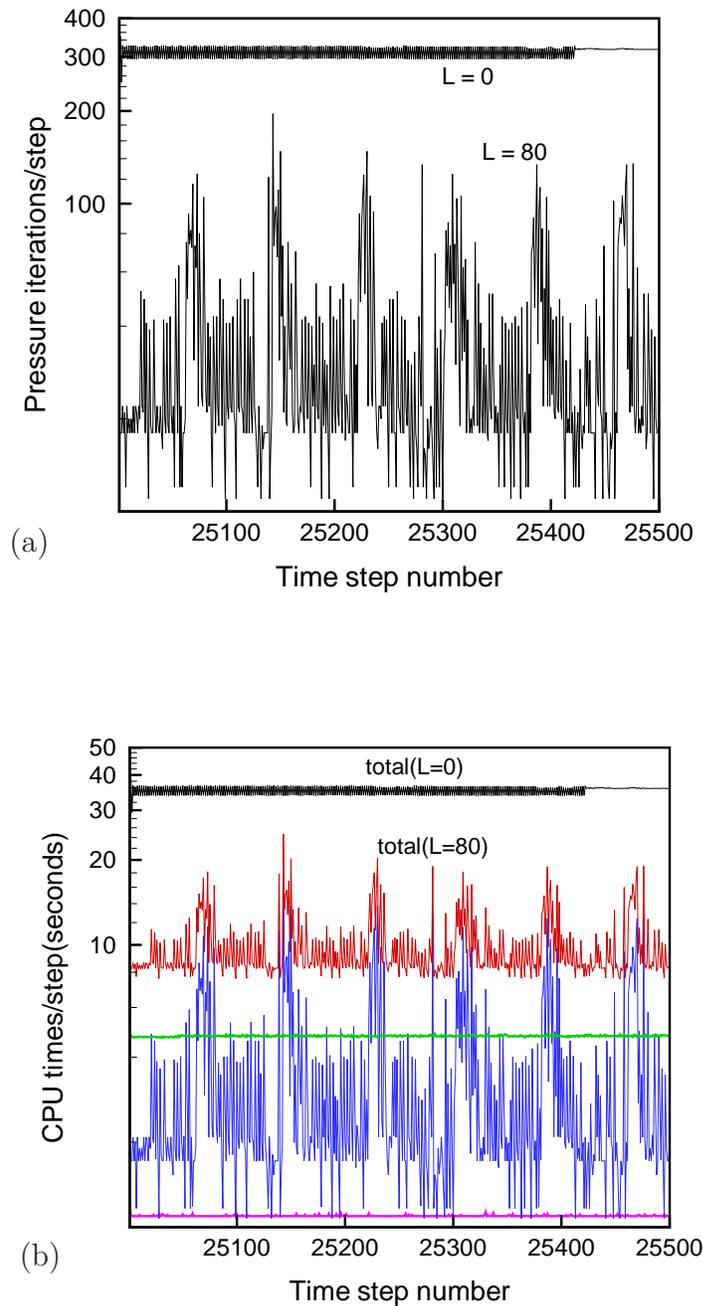


Figure 6.2: Performance of the code in solving the three-dimensional Orr-Sommerfeld stability problem *with* the projection method for the pressure equation. (a) Pressure iteration numbers per time step for $L = 0$ (without the projection method) and $L = 80$ (up to 80 previous solution vectors used in the projection method) vs. time step number; (b) CPU times per time step for all three velocity solves (pink line), the pressure solve (blue line), nonlinear term evaluation (green line), and the total operations (red line) vs. the time step number. In (b), the CPU time for the total operations with $L = 0$ is also shown. All timings were obtained using the “PetcGetTime” function.

effectiveness of the block diagonal mass matrix preconditioner. On the other hand, the number of pressure iterations is at least one order of magnitude larger than the total number of velocity iterations. We plot in Fig. 6.1b the total CPU time per step along with the CPU time breakdowns for the pressure solve, velocity solves and the nonlinear term evaluation. It is clear that the pressure solve is the dominant computational cost.

We next ran the code with the projection method ($L = 80$), i.e. initial guesses for the pressure iterations were constructed using up to 80 previous solution vectors. Fig. 6.2a shows the pressure iteration numbers in comparison with those of the former case. A significant reduction (approximately eleven-fold on average) in the pressure iterations is observed. This reduction readily translates into a four-fold reduction in the total CPU time as shown in Fig. 6.2b. Another important point notable from Fig. 6.2b is that now the pressure solve and the nonlinear term constitute comparable portions (on average 35% and 55%, respectively) of the total CPU time. This balanced role of the pressure solve and nonlinear term evaluation is typical of our solver and was also observed in other simulations. For example, in simulating flow over a backward-facing step at $Re = 344$, for a time interval of approximately three flow-through times and with $L = 60$, the pressure solve and the nonlinear term evaluation accounted for 59% and 29% of the total CPU time, respectively.

6.3.2 Parallel Performance

We examined the parallel performance of the developed code in simulating the backward-facing step flow at $Re = 172$ with the initial condition being the solution at $Re = 1$. We carried out runs for five cases: $(N_P, \text{DOF}) = (4, 35280), (8, 35280), (16, 70560), (32, 70560)$ and $(64, 141120)$ with N_P and DOF denoting the number of processors and the number of degrees of freedom, respectively. The three DOFs correspond to a fixed approximation order $k = 4$ and three meshes obtained from (anisotropic) refinement of a reference mesh.

Case	N_P	DOF	CPU time(seconds)/step	Parallel efficiency
1	4	35280	23.06	—
2	8	35280	12.64	91%
3	16	70560	17.24	73%
4	32	70560	8.09	94%
5	64	141120	8.35	97%

Table 6.1: Parallel efficiency in solving three-dimensional backward-facing step problem at $Re = 172$, with initial condition being the solution at $Re = 1$. N_P and DOF denote the number of processors and the number of degrees of freedom, respectively. The CPU time(seconds)/step is the total time per step averaged over the first 400 steps.

The parallel efficiency of the solver, η , was measured based on

$$\eta = \left(\frac{T_i}{T_{i+1}}\right)\left(\frac{N_P|_i}{N_P|_{i+1}}\right)\left(\frac{DOF|_{i+1}}{DOF|_i}\right) \times 100 \quad i = 1, \dots, 4, \quad (6.11)$$

where T_i denotes the CPU time per time step for case i . We chose this definition because, unfortunately, the parallel code has evolved independently of the sequential version, making the standard definition of the parallel efficiency impossible to calculate. (The standard definition of the parallel efficiency is

$$\eta = \left(\frac{T_S}{T_i} \frac{1}{N_P|_i}\right) \times 100 \quad i = 1, \dots, 5, \quad (6.12)$$

with subscript S referring to the fastest serial solver [44].) The definition given in 6.11 is a reasonable compromise in this situation. The characteristics of the processors were as mentioned in the previous section. The CPU time per step averaged over the first 400 time steps for each case along with the corresponding parallel efficiencies are listed in Table 6.1. Parallel efficiencies of above 90% were obtained in all cases except for case 3. The reduced efficiency in case 3 was due to the higher pressure iteration numbers in case 3 than that in the case 2. To verify this, the number of pressure iterations (averaged over 400 time steps) are shown in Table 6.2 for different cases. Clearly, a jump of almost 30%

Case	Pressure solve			Nonlinear term evaluation			3 velocity solves		
	Iter	T_P (%T)	η_P	T_C (%T)	T_N (%T)	η_N	Iter	T_V (%T)	η_V
1	481.0	20.54 (89%)	-	0.001 (0%)	1.8 (8%)	-	18	0.32 (1%)	-
2	557.3	11.34 (90%)	91%	0.001 (0%)	0.92 (7%)	98%	18	0.16 (1%)	100%
3	748.3	15.85 (92%)	72%	0.002 (0%)	0.93 (5%)	99%	24	0.23 (1%)	70%
4	821.8	7.37 (91%)	95%	0.003 (0%)	0.48 (6%)	97%	24	0.06 (1%)	192%
5	807.6	7.49 (90%)	98%	0.1 (1%)	0.56 (7%)	86%	25	0.07 (1%)	86%

Table 6.2: Profiling of the solver and parallel performance of the pressure solve, the nonlinear term evaluation and the velocity solves for the three-dimensional backward-facing step problem. Cases 1-5 correspond to those in Table 6.1 and “Iter” denotes the number of pressure (or summation of three velocity) iterations averaged over the first 400 time steps. T_P (T_C , T_N , or T_V) denote the CPU time in seconds per time step of the pressure (the nonlinear term communication, the nonlinear term evaluation, or the summation of three velocity solves) averaged over the first 400 steps. “%T” denotes the percentage of the total CPU time spent on a particular stage. η_P , η_N and η_V represent the parallel efficiency of the pressure solve, the nonlinear term evaluation and the velocity solves, respectively.

in the pressure iteration numbers in case 3 compared to case 2 is observed. A similar jump is also observed in the velocity iteration numbers (Table 6.2). We believe that these jumps in the iteration numbers are due to the increased anisotropy of the mesh in case 3 and the fact that the velocity and pressure preconditioners are not optimized for anisotropic meshes.

Besides the velocity and pressure iterations, we list in Table 6.2 both the CPU time breakdown and the percentage of the total CPU time for the pressure solve, the velocity solves, the nonlinear term evaluations, and the communication required for the nonlinear term evaluation (all averaged over the first 400 time steps). The parallel efficiency of the pressure solve (η_P), the velocity solves (η_V) and the nonlinear term evaluation (η_N) defined analogously to 6.11 are also given in Table 6.2. From Table 6.2, several points are

Case	Mat-Vec			PCApply		VecDot/VecNorm		PCGSolve
	VecScatter %T	%T	γ_s	%T	γ_s	%T	γ_s	γ_s
1	0%	29%	13.7%	54%	4.7%	6%	3.1%	8.9%
2	0%	31%	13.0%	50%	6.8%	7%	1.5%	9.1%
3	0%	31%	12.5%	49%	6.7%	10%	1.2%	8.8%
4	0%	33%	12.7%	40%	8.9%	16%	0.3%	10.2%
5	2%	31%	12.2%	37%	8.8%	20%	0.5%	9.9%

Table 6.3: Profiling of the preconditioned conjugate gradient solve (“PCGSolve”) in simulating three-dimensional flow over a backward-facing step. For cases 1-5 introduced in table 6.1, the percentage of the total CPU time (%T) spent on the matrix-vector products (“Mat-Vec”), the communication required in the matrix-vector products (“VecScatter”), the application of the preconditioned matrix (“PCApply”), and vector inner products/norm evaluations (“VecDot/VecNorm”) are listed. The computational rate efficiency γ_s , as defined in eq. 6.13, is also given for each operation and for the whole PCGSolve. In the second column, percentages smaller than 0.1% are shown as 0%.

notable. First, while the pressure solve requires 89 – 92% of the total time, the nonlinear term evaluation and the velocity solves take 6 – 8% and 1% of the total CPU time, respectively. Note that these figures are different from those presented in the previous section, which shows that pressure solve and the nonlinear term evaluation constitute comparable portions of the total CPU time. This is because at early simulation times, the number of pressure iterations is high due to the inexact initial condition, and as the simulation goes on the pressure iteration numbers drop, on average. Therefore, if instead of the first 400 time steps, the whole simulation time (two or three flow-through times) was considered, the average pressure iterations would be a few-fold lower than those presented in Table 6.2, and correspondingly the percentage of pressure solve time would drop to 50 – 60% of the total CPU time. This would then yield a more balanced role of the pressure solve and the nonlinear term evaluation.

Second, the communication for the nonlinear term evaluation was for all cases equal to or smaller than 1% of the total CPU time, verifying the compactness of the scheme and the effective implementation of the required communication.

Third, parallel efficiencies of higher than 86% for most cases and for the three tasks (velocity solves, pressure solve and the nonlinear term evaluation) are observed. The exception is again case 3, where the elevated iteration numbers degrade the parallel performance of the pressure and the velocity solves to 72% and 70%, respectively. Since the velocity solves only constitute a very small portion (1%) of the total time, this reduced performance does not materially affect overall performance. On the other hand, since the pressure solve constitutes the majority of the total CPU time, this reduced efficiency directly translates into a reduced overall efficiency of 73%, as seen in Table 6.1. However, if instead of only the first 400 time step, a realistic time interval of two-three flow-through times (40000 – 60000 time steps) was considered, the relatively low pressure efficiency would have less negative impact on the overall efficiency. This is due to the fact that during longer time periods, the pressure solve time is reduced to approximately 50% of the total CPU time as seen in the previous subsection, and the nonlinear term evaluation with very high parallel efficiency would contribute to higher overall efficiency.

We also list the percentage of the total CPU time required by different operations in the preconditioned conjugate gradient (PCG) iterations in Table 6.3. These operations are matrix vector products, the application of the preconditioner, vector inner-products and norm evaluations. Since all other operations including vector additions, scalar vector products and LU factorization contributed less than 0.5% of the total CPU time, they are not listed in the table. Also listed in Table 6.3 is the computational rate efficiency γ_s for each phase of operations and for the entire PCG solve. γ_s , also referred to as per processor (serial) efficiency, is defined as

$$\gamma_s = \frac{\text{MFlops}}{N_P S_P} \times 100, \quad (6.13)$$

where $S_P = 1500\text{MHz}$ is the nominal speed of the processors, and

$$\text{MFlops} = 10^{-6}(\text{sum of flops over all processors})/(\text{maximum time over all processors}). \quad (6.14)$$

The sum of flops over all processors was measured by calling the flag “log_summary” of the PETSc library. Here one flop is defined as one operation of any of the following types: addition, subtraction, multiplication and division. As in the nonlinear term evaluation, we observe a very small communication time (maximum 2% of the total CPU time) in the matrix-vector products. Since we used the serial ILU preconditioner for the pressure and a block diagonal mass matrix for the velocity equations, application of the preconditioner does not require any communication. On the other hand, the vector inner products and vector norm evaluations require global data exchanges. The very small computational rate efficiencies for these operations are the direct consequence of the required global communications. Despite negligible or zero communication required, small computational efficiencies of 12 – 13% and 5 – 8% are observed for matrix-vector products and the preconditioner applications, respectively. This is mainly due to memory loads and stores (out of cache operations) which are the real performance barriers. A total computational rate efficiency of approximately 10% is observed for the whole PCG solves.

We now compare this performance with two other high-order flow solvers. The first is the highly optimized spectral element Navier-Stokes code of Tufo and Fischer [115]. In a three-dimensional simulation with 27,799,110 DOF (8168 hexahedral elements of order $k = 15$) on 2048 processors of the Intel ASCI-Red machine with 333 MHz CPU at the US Sandia National Laboratory, Tufo and Fischer achieved approximately 46% computational efficiency. The second is the global discontinuous Galerkin atmospheric simulator of Dennis et al. [35]. In a two dimensional simulation with 393,216 DOF (6144 quadrilateral elements of order 7) on 16 – 2048 processors of the IBM Blue Gene machines (with 700 MHz speed and 2 Mbytes of L3 cache), they obtained approximately 8 – 11% computational efficiency. Compared with the latter case, our code performance

is considered acceptable, while compared with the former, our code requires further improvements. To this end, some implementation strategies will be explored in the following chapter.

Chapter 7

Conclusions and Future Directions

In this final chapter, we summarize the entire work and the contributions before exploring some future directions.

7.1 Summary and Conclusions

The objectives of this thesis were two-fold:

- to develop an efficient high-order discontinuous Galerkin scheme for solving the unsteady incompressible Navier-Stokes equations using triangular and tetrahedral elements in two and three space dimensions, respectively; and
- to implement the scheme in parallel and verify the accuracy and stability of the method by solving popular benchmarking problems in two and three space dimensions.

We here present the results and conclusions addressing each objective.

7.1.1 First Objective

We have developed an efficient and simple method for the numerical solution of the unsteady incompressible Navier-Stokes equations in convection-dominated flow regimes.

The scheme is based on a semi-explicit temporal discretization with explicit treatment of the nonlinear term and implicit treatment of the Stokes operator. The spatial discretization in the scheme is based on a high-order discontinuous Galerkin method on triangular and tetrahedral elements. The nonlinear term is discretized in divergence form by using the local Lax-Friedrichs fluxes. Spatial discretization of the Stokes operator employs both equal-order $(P_k - P_k)$ and mixed-order $(P_k - P_{k-1})$ velocity and pressure approximations. The interior penalty method is used for the discretization of the diffusion term. A second order approximate algebraic splitting is used to decouple the velocity and pressure calculations leading to an algebraic Helmholtz equation for each component of the velocity and a consistent Poisson equation for the pressure. The consistent Poisson operator was replaced by an equivalent (in stability and convergence) operator, namely that arising from the interior penalty discretization of the standard Poisson operator with appropriate boundary conditions. An important efficiency aspect of our scheme is compact stencil size discretization of the nonlinear term, and velocity and pressure equations.

For the penalty parameter of the interior penalty method, an explicit lower bound based on trace inverse inequalities has been derived that ensures the coercivity (stability) of the bilinear form for the second order Laplacian. The sharpness of the expression has been demonstrated by numerical examples. Knowing an explicit expression for the penalty parameter has a significant efficiency implication.

7.1.2 Second Objective

We have extensively verified the temporal and spatial accuracy of the method on several two- and three-dimensional benchmarking problems. We obtained second-order temporal convergence and the expected theoretical spatial convergence rates in solving test problems with known exact solutions. On the challenging Orr-Sommerfeld test problem at $Re = 7500$, the equal-order polynomial approximations of the velocity and pressure $(P_k - P_k)$ led to a stable and accurate solution, while the mixed-order method $(P_k - P_{k-1})$

yielded a non-physical instability. In simulating vortex shedding past a square cylinder at $Re = 100$ and in simulating a three-dimensional backward-facing step flow using the equal-order method, excellent agreement with other computational and experimental results was obtained.

Moreover, the developed solver was used to study the flow through a two-dimensional bileaflet mechanical heart valve geometry at $Re = 76$ (based on the leaflet thickness). Based on resolution independent results, a steady state solution was obtained. This solution was characterized by two recirculation zones associated with each leaflet: an asymmetric recirculation zone at the wake of the leaflet and another zone on the outer leading edge of the leaflet.

As regards implementation, the integrals involving the nonlinear term evaluations are carried out using quadrature schemes of sufficiently high order to ensure the stability of the discretization. The linear systems arising from the velocity and pressure discretizations are solved using the preconditioned conjugate gradient method. The velocity preconditioner is the matrix associated with the block diagonal mass matrix and the pressure preconditioner is based on the serial ILU(0) factorization. To accelerate the convergence of the pressure solve, an optimal initial guess based on the previous solution vectors through the projection method of Fischer [42] is used. The parallel implementation of the algorithm is based on the AOMD mesh library, the PETSc library and the C++ programming language. Basing the programming on these three elements greatly reduced the code development time and enhanced the readability, extendibility and the ease of maintenance of the code.

The performance of the solver was also studied. It appears that the nonlinear term evaluation and pressure solves constitute comparable portions of the total computational time and together account for the overwhelming majority (approximately 90%) of the total computational time. The parallel performance of the code was examined by simulating the backward-facing step flow problem on a series of anisotropically refined meshes

using 4 – 64 processors. Overall very good parallel performance was obtained. A total computational rate efficiency of approximately 10% was observed for the preconditioned conjugate gradient iterations.

7.2 Contributions

The main contributions of this work include the following:

- We solved a long-standing problem arising from a class of discontinuous Galerkin methods for elliptic problems, namely the lack of an explicit expression for the penalty parameter. Specifically, an explicit lower bound for the penalty parameter of the interior penalty method of Arnold [4] and the similar method of Baker [6] for the case of triangular and tetrahedral meshes and for both conforming and non-conforming discretizations was derived.
- A new strategy for the discretization of the non-linear term in the Navier-Stokes equations was introduced. This method is characterized by its simplicity and inherent local conservativity, properties that are absent in previously developed schemes for the nonlinear term discretization.
- A new method for solving the Stokes system based on an algebraic splitting method was also introduced. An important feature of the method is its compact stencil size for both velocity and pressure operators, enhancing the overall efficiency and simplicity of the scheme.

7.3 Future Directions

As emphasized in the introductory chapter, our ultimate goal is to use the developed code to perform direct numerical simulations of transitional and turbulent flows through mechanical heart valve geometries. Since extensive validations on a variety of two- and

three-dimensional problems have been conducted, we are confident that the developed numerical scheme for the Navier-Stokes equations is suitable for mechanical heart valve flow simulations. However, several immediate tasks related to implementation and performance must be accomplished before realistic mechanical heart valve flow simulations become possible. These steps include further efficiency evaluation and possible efficiency improvements, tuning the code for large numbers of processors, and programming curved elements. We describe these steps below.

7.3.1 Strategies for Efficiency Improvement

As seen in the previous chapter, the pressure solve and the nonlinear term evaluations typically constitute more than 90% of the total computational time. Moreover, the dominant operations involving the pressure solve are the Laplacian matrix-vector product and the application of the preconditioner. Therefore, any attempt to improve the efficiency must address the three operations: nonlinear term evaluation, Laplacian matrix-vector product and the preconditioning strategy for the pressure equations. Here we present alternatives for the implementation of these operations which may enhance code efficiency.

Nonlinear Term Evaluation

For simplicity, let's consider the nonlinear term of the form

$$C_i = \int_O u_h^2 \frac{\partial L_i}{\partial \xi} d\xi d\eta \quad i = 0, \dots, N. \quad (7.1)$$

Applying the quadrature formulae described in chapter 3 in evaluating this integral yields

$$C_i \approx \sum_{j=0}^M w_j u_h^2(\zeta_j) \frac{\partial L_i}{\partial \xi} \Big|_{\zeta_j} \quad i = 0, \dots, N, \quad (7.2)$$

where

$$u_h(\zeta_j) = \sum_{i=0}^N u_h(\xi_i) L_i(\zeta_j) \quad j = 0, \dots, M. \quad (7.3)$$

Here, $\{\xi_i\}$ denotes the set of nodal points with the total number of $N + 1$ corresponding to the approximation order k , and $\{\zeta_i\}$ represents a set of economic quadrature points ([108]) with total number of $M + 1$ corresponding to the quadrature order q . For stability reasons the nonlinear term must be evaluated exactly using quadrature order $q \approx 3k$; thus, $M \approx \gamma N$ where $\gamma > 1$. The cost associated with the operations in 7.2 and 7.3 are $(2M + 1)(N + 1) + 2(M + 1)$ and $(M + 1)(2N + 1)$ floating point operations (or memory references), respectively. Note that $w_j u_h^2(\zeta_j)$ is computed and the resulting vector is used in the evaluation of eq. 7.2. The total cost is then

$$\mathcal{C} = (2M + 1)(N + 1) + (M + 1)(2N + 1) + 2(M + 1). \quad (7.4)$$

Since in two space dimensions $N + 1 = \frac{(k+1)(k+2)}{2}$ (in three space dimensions $N + 1 = \frac{(k+1)(k+2)(k+3)}{6}$) the leading complexity order is $\mathcal{O}(4\gamma k^4)$ ($\mathcal{O}(4\gamma k^6)$).

To reduce this cost, we suggest to first expand u_h and $\frac{\partial L_i}{\partial \xi}$ with respect to a orthonormal tensor-product basis function, namely multivariate analogues of Jacobi polynomials [71, 38], as follows

$$u_h = \sum_{i=0}^N \alpha_i b_i(\xi), \quad (7.5a)$$

$$\frac{\partial L_i}{\partial \xi} = \sum_{i=0}^N \beta_{i,j} b_j(\xi), \quad (7.5b)$$

where

$$b_i(\xi) := b_{rs}(\xi) := \phi_r(\zeta) \phi_{rs}(\eta), \quad (7.6)$$

with $\zeta = \frac{2(1+\xi)}{1+\eta} - 1$, ($0 \leq r, s; r + s \leq i$) and

$$\phi_i(z) = P_i^{0,0}(z) \quad (7.7a)$$

$$\phi_{ij}(z) = \left(\frac{(2r+1)(r+s+1)}{2} \right)^{1/2} \left(\frac{1-z^2}{2} \right) P_j^{2i+1,0}(z). \quad (7.7b)$$

Here, $P_i^{a,b}(z)$ signifies Jacobi polynomials (e.g., [110]). Then the integral in 7.1 can be written as

$$C_i = \int_O u_h^2 \sum_{j=0}^N \beta_{i,j} b_j(\xi) d\xi d\eta = \sum_{j=0}^N \beta_{i,j} \int_O u_h^2 b_j(\xi) d\xi d\eta, \quad (7.8)$$

or in matrix form

$$\begin{bmatrix} C_0 \\ \cdot \\ \cdot \\ \cdot \\ C_N \end{bmatrix} = \begin{bmatrix} \beta_{0,0} & \cdot & \cdot & \cdot & \beta_{0,N} \\ \cdot & \cdot & & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot \\ \beta_{N,0} & \cdot & \cdot & \cdot & \beta_{N,N} \end{bmatrix} \begin{bmatrix} \int_O u_h^2 b_0(\boldsymbol{\xi}) d\xi d\eta \\ \cdot \\ \cdot \\ \cdot \\ \int_O u_h^2 b_N(\boldsymbol{\xi}) d\xi d\eta \end{bmatrix}, \quad (7.9)$$

To evaluate the integral, we follow the procedure described in [67]. We exploit the tensor-product form of $b_i(\boldsymbol{\xi})$, map the integral over a unit square, and use the composite one dimensional Gauss quadratures to carry out the integral. Specifically,

$$\int_O u_h^2 b_i(\boldsymbol{\xi}) d\xi d\eta = \int_{-1}^1 \int_{-1}^1 u_h^2(\zeta, \eta) b_i(\zeta, \eta) \frac{(1-\eta)}{2} d\zeta d\eta \quad (7.10a)$$

$$\approx \sum_{m=0}^Q \sum_{l=0}^Q u_h^2(\zeta_l, \eta_m) \phi_r(\zeta_l) \phi_{rs}(\eta_m) w_l w_m \quad (7.10b)$$

$$= \sum_{m=0}^Q \phi_{rs}(\eta_m) \sum_{l=0}^Q u_h^2(\zeta_l, \eta_m) \phi_r(\zeta_l) w_l w_m, \quad (7.10c)$$

(ζ_l, η_m) are quadrature points with total number of $(Q+1)$ in each direction. Exact integration requires $Q+1 \approx 3k/2$. w_l and w_m are quadrature weights which for definition we refer to [67]. The last equation can be evaluated in two steps as

$$\tilde{f}_r(\eta_m) = \sum_{l=0}^Q u_h^2(\zeta_l, \eta_m) \phi_r(\zeta_l) w_l w_m, \quad (7.11a)$$

$$\int_O u_h^2 b_i(\boldsymbol{\xi}) d\xi d\eta \approx \sum_{m=0}^Q \tilde{f}_r(\zeta_m) \phi_{rs}(\eta_m) \quad (7.11b)$$

Assuming $u_h^2(\zeta_l, \eta_m) w_l w_m$ can be computed before the evaluation of the summation in the first step, the first step requires $Q(2Q+1)(k+1) + 2(Q+1)^2$ floating point operations, while the second step requires $(2Q+1)(N+1)$ floating point operations. After evaluating this integral, the vector C_i in eq. 7.9 is recovered through the matrix-vector product with the cost of $(N+1)(2N+1)$ floating point operations.

We now step back and find the complexity of representing u_h based on the modal basis, eq. 7.5a, as well as the complexity of projecting u_h to the quadrature points

(ζ_l, η_m) required in eq. 7.11a. Note that since $\frac{\partial L_i}{\partial \xi}$ is constant for all elements, the expansion coefficients in eq. 7.5b can be computed and stored in a preprocessing stage. To compute the expansion coefficient of u_h , we start with the equivalence of nodal and modal representation of u_h , namely

$$u_h = \sum_{i=0}^N u_h(\boldsymbol{\xi}_i) L_i(\boldsymbol{\xi}) = \sum_{i=0}^N \alpha_i b_i(\boldsymbol{\xi}). \quad (7.12)$$

Taking the elemental inner-product of both sides of the second equality with respect to the basis function $b_j(\boldsymbol{\xi})$ yields

$$\sum_{i=0}^N \alpha_i \int_O b_i(\boldsymbol{\xi}) b_j(\boldsymbol{\xi}) d\xi d\eta = \sum_{i=0}^N u_h(\boldsymbol{\xi}_i) \int_O L_i(\boldsymbol{\xi}) b_j(\boldsymbol{\xi}) d\xi d\eta, \quad j = 0, \dots, N, \quad (7.13)$$

which in matrix form reads

$$\begin{bmatrix} \int_O b_0 b_0 & \cdot & \cdot & \cdot & \int_O b_0 b_N \\ \cdot & \cdot & & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot \\ \int_O b_N b_0 & \cdot & \cdot & \cdot & \int_O b_N b_N \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_N \end{bmatrix} = \begin{bmatrix} \int_O L_0 b_0 & \cdot & \cdot & \cdot & \int_O L_N b_0 \\ \cdot & \cdot & & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot \\ \int_O L_0 b_N & \cdot & \cdot & \cdot & \int_O L_N b_N \end{bmatrix} \begin{bmatrix} u_h(\boldsymbol{\xi}_0) \\ \cdot \\ \cdot \\ \cdot \\ u_h(\boldsymbol{\xi}_N) \end{bmatrix}, \quad (7.14)$$

In this system, the matrix on the left is identity due to the fact that the modal basis functions are orthonormal. The matrix on the right is constant for all elements and can be computed and stored in a preprocessing stage. Therefore, the expansion coefficients α_i can be calculated through the matrix-vector product in $(N+1)(2N+1)$ floating point operations.

Finally, to obtain $u_h(\zeta_l, \eta_m)$ we once again exploit the tensor-product structure of the modal basis. We have

$$u_h(\zeta_l, \eta_m) = \sum_{rs}^N \alpha_{rs} b_{rs}(\zeta_l, \eta_m) = \sum_r \sum_s \alpha_{rs} \phi_r(\zeta_l) \phi_{rs}(\eta_m), \quad (7.15)$$

which can be evaluated in two steps as

$$\tilde{f}_r(\eta_m) = \sum_s \alpha_{rs} \phi_{rs}(\eta_m) \quad (7.16a)$$

$$u_h(\zeta_l, \eta_m) = \sum_r \tilde{f}_r(\eta_m) \phi_r(\zeta_l). \quad (7.16b)$$

The first and the second steps can be performed in $(Q + 1)(2(N + 1) - (k + 1))$ and $(Q + 1)^2(2k + 1)$ floating point operations, respectively.

Below we summarize each significant step and its complexity in evaluating the nonlinear term (eq. 7.1).

- Evaluation of the right-hand-side vector in eq. 7.9 with the cost of $Q(2Q + 1)(k + 1) + 2(Q + 1)^2 + (2Q + 1)(N + 1)$ floating point operations.
- Matrix-vector product in eq. 7.9 requiring $(N + 1)(2N + 1)$ floating point operations.
- Matrix-vector product on the right-hand-side of eq. 7.14 requiring $(N + 1)(2N + 1)$ floating point operations.
- Projection in eq. 7.15 with the cost of $(Q + 1)(2(N + 1) - (k + 1)) + (Q + 1)^2(2k + 1)$ floating point operations.

Therefore, the total number of floating point operations in evaluating the nonlinear term (eq. 7.1) using the new procedure is

$$\begin{aligned} \mathcal{C}_{2D} = & 2(N + 1)(2N + 1) + Q(2Q + 1)(k + 1) + (2Q + 1)(N + 1) + 2(Q + 1)^2 \\ & + (Q + 1)(2(N + 1) - (k + 1)) + (Q + 1)^2(2k + 1) \end{aligned} \quad (7.17)$$

The first term is $\mathcal{O}(k^4)$ operations while the remaining terms are $\mathcal{O}(k^3)$ operations. Similar procedure can be followed for the evaluation of the integral in three space dimensions leading to the total cost of

$$\begin{aligned} \mathcal{C}_{3D} = & 2(N + 1)(2N + 1) + Q^2(2Q + 1)(k + 1) + Q(2Q - 1)(k + 1)(k + 2)/2 \\ & + (2Q + 1)(N + 1) + 2(Q + 1)^3 + Q(2(N + 1) - (k + 1))(k + 2)/2 \\ & + Q^2((k + 1)(k + 2) - (k + 1)) + Q^3(2k + 1) \end{aligned} \quad (7.18)$$

In the above equation the first term is proportional to $\mathcal{O}(k^6)$ while the remaining terms are proportional to $\mathcal{O}(k^4)$.

Order	# of floating point operations			
	Two dimensions		Three dimensions	
	\mathcal{C}	\mathcal{C}_{2D}	\mathcal{C}	\mathcal{C}_{3D}
1	49	60	81	116
2	294	297	974	967
3	769	859	4273	4008
4	1998	1911	29575	11791
5	4059	3648	74194	28337
6	7882	6283	240871	59571
7	15951	10059	481361	113838
8	26019	15237	1142043	202509
9	40223	22108	2167371	340661
10	59559	30981	3864089	547831

Table 7.1: The number of floating point operations in evaluating the nonlinear term (eq. 7.1) using two different approaches: current implementation using the quadrature formulae eq. 7.2; and new approach based on the tensor-product modal basis functions. \mathcal{C} (eq. 7.4) is the number of operations for the current implementation and \mathcal{C}_{2D} (eq. 7.17) and \mathcal{C}_{3D} (eq. 7.18) denote the corresponding operation counts for the new approach. Note that since economic quadratures on standard triangle and tetrahedron have been reported only up to quadrature orders $q = 20$ and $q = 21$ [108], the complexity estimate \mathcal{C} for $k \geq 6$ in two dimensions (and $k \geq 7$ in three dimensions) are based on composite one-dimensional quadrature formulae.

Table 7.1 shows the total number of floating point operations with respect to the approximation orders $k = 1, \dots, 10$ for the current implementation (eq. 7.4) and for the new approach (eqs. 7.17 and 7.18) for both two and three space dimensions. As is

clear from the table, the new approach does not offer any advantage over the current implementation for low orders $k \leq 3$ in both two and three space dimensions. On the other hand, for higher orders the new approach appears more efficient. While in two dimensions the advantage of the new approach is only notable at high orders $k > 8$, in three dimensions the advantage of the new approach is significant even for $k = 4$ and increases as k increases. For example, for $k = 4$, the evaluation of the nonlinear term using the new approach requires less than half of the number of floating point operations required using the current implementation.

Matrix-vector product

Since in the current implementation, the matrix associated with the global Laplacian is explicitly constructed, a Laplacian matrix-vector product requires $\mathcal{O}(Ek^{2d})$ floating point operations and $\mathcal{O}(Ek^{2d})$ memory references with E denoting the number of elements. This complexity is higher than the complexity $\mathcal{O}(Ek^d)$ used (in chapter 1, section 1.3.1) to demonstrate the efficiency of the high-order methods over the low-order methods. Therefore, a more efficient implementation scheme needs to be adopted. As mentioned in the previous chapter, the real performance barriers are out of cache operations. This is due to the fact that in current computer architectures, the regular memory speed is much lower than the microprocessor speed and a single noncached memory reference can cost 10 – 100 clock cycles [36]. Thus, an effective strategy for reducing cost is to reduce the memory reference rate. We here present a matrix-free approach for the global matrix-vector products with a reduced memory reference rate.

For simplicity, let us consider the evaluation of the term

$$\mathcal{S} \equiv \sum_{K=1}^E \int_K \left(\frac{\partial v}{\partial x} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \frac{\partial u}{\partial y} \right) d\mathbf{x}, \quad (7.19)$$

appearing in the interior penalty discretization of the Laplacian in two space dimensions ($d = 2$) on a mesh consisting of straight-sided triangles. After applying the nodal basis

described in chapter 3, the above integral can be evaluated as

$$\underline{\mathcal{S}} \equiv \begin{bmatrix} S_1 & & & & \\ & \cdot & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & S_E \end{bmatrix} \begin{bmatrix} \underline{u}_1 \\ \cdot \\ \cdot \\ \cdot \\ \underline{u}_E \end{bmatrix}, \quad (7.20)$$

where S_K is a $(N + 1) \times (N + 1)$ matrix defined as

$$S_K = J_K (D_K^x{}^T (V^{-1})^T V^{-1} D_K^x + D_K^y{}^T (V^{-1})^T V^{-1} D_K^y), \quad (7.21)$$

with

$$D_K^x = D^\xi \frac{\partial \xi}{\partial x} \Big|_K + D^\eta \frac{\partial \eta}{\partial x} \Big|_K, \quad (7.22a)$$

$$D_K^y = D^\xi \frac{\partial \xi}{\partial y} \Big|_K + D^\eta \frac{\partial \eta}{\partial y} \Big|_K \quad (7.22b)$$

and $\underline{u}_K \equiv (u_0, \dots, u_N)$ is the vector containing the unknown values of element K . The matrices V , D^ξ and D^η were defined in chapter 3. If matrix S_K is constructed for each element and then the matrix-vector multiplication is carried out (similar to the current explicit construction of the global matrix), the cost scales as $\mathcal{O}(Ek^{2d})$ floating point operations and $\mathcal{O}(Ek^{2d})$ memory references.

Alternatively, for a matrix-free approach, we first write the matrix S_K as

$$S_K = a_K D^{\xi\xi} + b_K D^{\eta\eta} + c_K D^{\xi\eta}, \quad (7.23)$$

where

$$D^{\xi\xi} = D^{\xi T} (V^{-1})^T V^{-1} D^\xi, \quad (7.24a)$$

$$D^{\eta\eta} = D^{\eta T} (V^{-1})^T V^{-1} D^\eta, \quad (7.24b)$$

$$D^{\xi\eta} = D^{\xi T} (V^{-1})^T V^{-1} D^\eta + D^{\eta T} (V^{-1})^T V^{-1} D^\xi, \quad (7.24c)$$

dimensions.

Pressure Preconditioner

If the global pressure matrix is not to be explicitly constructed, the ILU(0) preconditioning strategy for the pressure equation as employed in the current implementation is no longer possible and we need to adopt a different preconditioning approach. We suggest a two level element-based additive Schwarz method with a global coarse part and a local fine part. Specifically, the preconditioner matrix B has the form

$$B = JA_C^{-1}J^T + \sum_{K=1}^{K=E} R_K^T J_K A_f^{-1} R_K, \quad (7.27)$$

where J is the interpolation matrix from coarse to fine space, consistent with the fine space, and R_K is a boolean operator that maps global indices to the local ones restricted to the element K . With approximation order $k > 1$, A_C is the matrix associated with the IP discretization on the original mesh but with lower order approximation $k_C = 1$. J_K is the Jacobian of each element and A_f corresponds to the discretization of the operator

$$\int_O \nabla u_h \cdot \nabla v d\mathbf{x} - \int_{\partial O} \left(\frac{1}{2} u_h \nabla v \cdot \mathbf{n} + \frac{1}{2} v \nabla u_h \cdot \mathbf{n} \right) ds + \int_{\partial O} \mu v u_h ds, \quad (7.28)$$

where O is the standard element. This preconditioner is not optimal with respect to the approximation order and mesh isotropy; that is, the iteration numbers grow as the approximation order increases or as the mesh becomes more anisotropic. Despite this, it is potentially an effective preconditioner. There exists a fast parallel direct solver for the solution of the global coarse preconditioner system [116]. This solver is easily accessible through the PETSc library. Moreover, the local part of the preconditioner involves only a single matrix inversion of order $\mathcal{O}(k^d)$ and the application of the preconditioner requires only $\max(\mathcal{O}(Ek^d), \mathcal{O}(k^{2d}))$ memory references. The performance of this preconditioner on large problems needs to be investigated.

7.3.2 Performance Tuning on Large Parallel Computers

As seen in the previous chapter, the parallel performance of the code has been studied for up to 64 processors. Since mechanical heart valve flow simulations require larger number of processors (order several hundreds to several thousands), the code must be tested on and tuned for such large machines. During the testing process, possible improvements regarding communication strategies may become necessary. In particular, in the current implementation, the communication pattern for nonlinear term evaluations requires n data exchanges per processor where n is the number of elements on the partition boundaries. To reduce this, the data for all elements destined to a given processor must be first packed together and then sent to the destination through a single message. This can be programmed using the general *scatter* class provided by the PETSc library.

7.3.3 Curved Elements

Physiological flow simulations involve geometries with curved boundaries. As a result, the corresponding computational meshes contain curved elements along the geometry boundaries. While a numerical scheme for operator evaluations over curved elements was presented in chapter 3, the scheme has not yet been implemented. This is the subject of future work. The AOMD library will facilitate this task by providing functions for constructing Bezier and Lagrange curves. Although some efficiency strategies such as those for inner product evaluation and system preconditioning described in the previous sections may not directly apply to the curved elements, the use of curved elements does not have a significant impact on the efficiency of the solver. This is because the use of curved elements is limited to boundaries of the domain and the majority of the elements still remain straight-sided (or planar) to fill in the interior of the domain.

Bibliography

- [1] Heart and Vascular Institute, 2006. <http://www.clevelandclinic.org/heartcenter/>.

- [2] Minnesota Biomedical and Bioscience Network, 2006. <http://mbbnet.umn.edu/firsts/>.

- [3] B.F. Armaly, F. Durst, J.C.F. Pereira, and B. Schönung. Experimental and theoretical investigation of backward-facing step flow. *Journal of Fluid Mechanics*, 127:473–496, 1983.

- [4] D.N. Arnold. An interior penalty finite-element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19(4):742–760, 1982.

- [5] D.N. Arnold, F. Brezzi, B. Cockburn, and L.D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.

- [6] G.A. Baker. Finite Element Methods for Elliptic Equations Using Nonconforming Elements. *Mathematics of Computation*, 31(137):45–59, 1977.

- [7] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.

- [8] S. Balay, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, and H. Zhang. PETSc Web page, 2001. <http://www.mcs.anl.gov/petsc>.
- [9] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.
- [10] D. Barkley and R.D. Henderson. Three-dimensional Floquet stability analysis of the wake of a circular cylinder. *Journal of Fluid Mechanics*, 322:215–241, 2006.
- [11] T.J. Barth. Recent developments in high order k-Exact reconstruction on unstructured meshes. *AIAA paper 93-0668*, 1993.
- [12] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 131(2):267–279, 1997.
- [13] Y. Bazilevs and T.J.R. Hughes. Weak imposition of Dirichlet boundary conditions in fluid mechanics. *Computers and Fluids*, December 2005. In press.
- [14] G. Biswas, M. Breuer, and F. Durst. Backward-facing step flows for various expansion ratios at low and moderate Reynolds numbers. *Journal of Fluids Engineering*, 126:362–374, 2004.
- [15] M. M. Black, T. Cochrane, P. V. Lawford, H. Reul, and A. Yoganathan. Design and flow characteristics. In E. Bodnar and R. Frater, editors, *Replacement cardiac valves*, pages 1–20, FairElmsford, NY, 1991. Pergamon Press Inc.

- [16] B. Bluestein, Y.M. Li, and I. B. Krukenkamp. Free emboli formation in the wake of bileaflet mechanical heart valves and the effects of implantation techniques. *Journal of Biomechanics*, 35:1533–1540, 2002.
- [17] C. H. Brown, L. B. Leverett, C. W. Lewis, C. P. Alfrey, and J. D. Hellums. Morphological, biochemical, and functional changes in human platelets subjected to shear stress. *Journal of Laboratory and Clinical Medicine*, 86(3):462–471, 1975.
- [18] P. Browne, A. Ramuzat, R. Saxena, and A. P. Yoganathan. Experimental investigation of the steady flow downstream of the St. Jude bileaflet heart valve: a comparison between laser Doppler velocimetry and particle image velocimetry techniques. *Annals of Biomedical Engineering*, 28:39–47, 2000.
- [19] C. Canuto, M.Y. Hussaini, A. Quarteroni, and T.A. Zang. *Spectral Methods in Fluid Mechanics*. Springer-Verlag, Heidelberg.
- [20] P. Castillo. Performance of discontinuous Galerkin methods for elliptic PDEs. *SIAM Journal on Scientific Computing*, 24:524–547, 2002.
- [21] A.J. Chorin. On the convergence of discrete approximations to the Navier-Stokes equations. *Mathematics of Computation*, 23(106):341–353, 1969.
- [22] B. Cockburn, G. Kanschat, and D. Schötzau. The local discontinuous Galerkin method for the Oseen equations. *Mathematics of Computation*, 73:569–593, 2004.
- [23] B. Cockburn, G. Kanschat, and D. Schötzau. A locally conservative LDG method for the incompressible Navier-Stokes equations. *Mathematics of Computation*, 74(251):1067–1095, 2005.
- [24] B. Cockburn, G. Kanschat, D. Schötzau, and C. Schwab. The local discontinuous Galerkin method for the Stokes system. *SIAM Journal on Numerical Analysis*, 40:319–343, 2002.

- [25] B. Cockburn, G.E. Karniadakis, and C.W. Shu. Discontinuous Galerkin Methods: Theory, Computation and Applications. 2000.
- [26] B. Cockburn and C.W. Shu. The local discontinuous Galerkin finite element method for convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6):2440–2463, 1998.
- [27] B. Cockburn and C.W. Shu. Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, 2001.
- [28] S. Collis and S. Ramakrishnan. The local variational multiscale method. *Third MIT Conference on Computational Fluid and Solid Dynamics*, 2005.
- [29] W. Couzy. *Spectral Element Solution Discretization of the Unsteady Navier-Stokes Equations and Its Iterative Solution on Parallel Computers*. PhD thesis, Swiss Federal Institute of Technology, Lausanne, 1995.
- [30] M. Crouzeix and P.A. Raviart. Conforming and nonconforming finite element methods for solving the stationary Stokes equations. *Recherche Operationnelle Ser. Rouge*, 7:33–75, 1973.
- [31] R.M. Darekar and S.J. Sherwin. Flow past a square-section cylinder with a wavy stagnation face. *Journal of Fluid Mechanics*, 426:263–295, 2001.
- [32] P.J. Davis and P. Rabinowitz. *Methods of numerical integration*. Academic Press New York, 1975.
- [33] D. B. DeGraaff and J. K. Eaton. A high-resolution laser Doppler anemometer: design, qualification, and uncertainty. *Experiments in Fluids*, 30:522–530, 2001.
- [34] L. Demkowicz, W. Rachowicz, and P. Devloo. A fully automatic hp-adaptivity. *Journal of Scientific Computing*, 17(1):117–142, 2002.

- [35] J.M. Dennis, M. Levy, R.D. Nair, H.M. Tufo, and T. Voran. Towards an efficient and scalable discontinuous Galerkin atmospheric model. *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, page 257a, 2005.
- [36] M.O. Deville, P.F. Fischer, and E.H. Mund. *High-order methods for incompressible fluid flow*. Cambridge University Press, 2002.
- [37] W.R. Dimitri and B.T. Williams. Fracture of the duromedics mitral valve housing with leaflet escape. *Journal of Cardiovascular Surgery*, 31:41–46, 1990.
- [38] M. Dubiner. Spectral methods on triangles and other domains. *Journal of Scientific Computing*, 6(4):345–390, 1991.
- [39] X. Feng and O. A. Karakashian. Two-level additive Schwarz methods for a discontinuous Galerkin approximation of second order elliptic problems. *SIAM Journal on Numerical Analysis.*, 30:1343–1365, 2001.
- [40] P. Fischer, F. Loth, S.E Lee, S.W. Lee, D. Smith, and H. Bassiouny. Simulation of high Reynolds number vascular flows. 2006. submitted, <http://www-unix.mcs.anl.gov/~fischer/pubhtml/>.
- [41] P. Fischer, F. Loth, S.W. Lee, D. Smith, H. Tufo, and H. Bassiouny. Parallel simulation of high Reynolds number vascular flows. *Proceeding of Parallel Computation Fluid Dynamics*, 2005. <http://www-unix.mcs.anl.gov/~fischer/pubhtml/>.
- [42] P.F. Fischer. Projection techniques for iterative solution of $Ax = b$ with successive right-hand sides. *Computer Methods in Applied Mechanics and Engineering*, 163(1):193–204, 1998.

- [43] P.F. Fischer and J.W. Lottes. Hybrid Schwarz-multigrid methods for the spectral element method: extensions to Navier-Stokes. *Domain Decomposition Methods in Science and Engineering*, pages 35–49, 2004.
- [44] G.C. Fox, M.A. Johnson, G.A. Lyzenga, S.W. Otto, J.K. Salmon, and D.W. Walker. *Solving problems on concurrent processors. Vol. 1: General techniques and regular problems*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1988.
- [45] L.P. Franca, S.L. Frey, and T.J.R. Hughes. Stabilized finite element methods. I: Application to the advective-diffusive model. *Computer Methods in Applied Mechanics and Engineering*, 95(2):253–276, 1992.
- [46] C. Geuzaine and J. F. Remacle. Gmsh reference manual, edition 1.12, 2003. <http://www.geuz.org.org/gmsh/>.
- [47] F.X. Giraldo. High-order triangle-based discontinuous Galerkin methods for hyperbolic equations on a rotating sphere. *Journal of Computational Physics*, 214(2):447–465, 2006.
- [48] V. Girault, B. Riviere, and M.F. Wheeler. A discontinuous Galerkin method with nonoverlapping domain decomposition for the Stokes and Navier-Stokes problems. *Mathematics of Computation*, 74:53–84, 2005.
- [49] G.H. Golub and C.F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 1983.
- [50] D. Gottlieb and S.A. Orszag. *Numerical analysis of spectral methods*. SIAM-CBMS, Philadelphia, 1977.
- [51] M. Grigioni, C. Daniele, G. D’Avenio, and V. Barbaro. The influence of the leaflets’ curvature on the flow field in two bileaflet heart valves. *Journal of Biomechanics*, 34:613–621, 2001.

- [52] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI: portable parallel programming with the message-passing interface*. MIT Press Cambridge, MA, USA, 1994.
- [53] B. Gustafsson, H. Kreiss, and J. Oliger. *Time dependent problems and difference methods*. Wiley, New York, 1995.
- [54] P. Hansbo and M. G. Larson. Discontinuous Galerkin methods for incompressible and nearly incompressible elasticity by Nitsche’s method. *Computer Methods in Applied Mechanics and Engineering*, 191(17):1895–1908, 2002.
- [55] J.J. Hathcock. Flow Effects on Coagulation and Thrombosis. *Arteriosclerosis, Thrombosis, and Vascular Biology*, 26(8):1729, 2006.
- [56] M.O. Henriksen and J. Holmen. Algebraic Splitting for Incompressible Navier–Stokes Equations. *Journal of Computational Physics*, 175(2):438–453, 2002.
- [57] J.S. Hesthaven. From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex. *SIAM Journal on Numerical Analysis*, 35(2):655–676, 1998.
- [58] J.S. Hesthaven and C. H. Teng. Stable spectral methods on tetrahedral elements. *SIAM J. Sci. Comput.*, 21:2352–2380, 2000.
- [59] J.S. Hesthaven and T. Warburton. Nodal high-order methods on unstructured grids. *Journal of Computational Physics*, 181(1):186–221, 2002.
- [60] J.G. Heywood, R. Rannacher, and S. Turek. Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 22(5):325–352, 1996.
- [61] J.O. Hinze. *Turbulence*. McGraw Hill, New York, 1975.

- [62] T.J.R. Hughes, G. Engel, L. Mazzei, and M. G. Larson. The continuous Galerkin method is locally conservative. *Journal of Computational Physics*, 163:467–488, 2000.
- [63] Carbomedics Inc. Prosthetic heart valve information for use. www.carbomedics.com/pdfs/CPHV.pdf.
- [64] K.E. Jansen. A stabilized finite element method for computing turbulence. *Computer Methods in Applied Mechanics and Engineering*, 174(3):299–317, 1999.
- [65] A.K. Joshi, R.L. Leask, J.G. Myers, M. Ojha, J. Butany, and C.R. Ethier. Intimal thickness is not associated with wall shear stress patterns in the human right coronary artery. *Arteriosclerosis, Thrombosis, and Vascular Biology*, 24:2408–2414, 2004.
- [66] A.S. Kantak, B.K. Gale, Y. Lvov, and S.A. Jones. Platelet function analyzer: shear activation of platelets in microchannels. *Biomedical Microdevices*, 5(3):207–215, 2003.
- [67] G. Karniadakis and S.J. Sherwin. *Spectral/hp element methods for CFD*. Oxford University Press New York, 2005.
- [68] G.E. Karniadakis, S.A. Orszag, and M. Israeli. High-order splitting methods for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 97:414–443, 1991.
- [69] G. Karypis and V. Kumar. METIS: A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices, 1998. <http://glaros.dtc.umn.edu/gkhome/views/metis>.

- [70] M.J. King, T. David, and J. Fischer. Three-dimensional study of the effect of two leaflet opening angles on the time-dependent flow through a bileaflet mechanical heart valve. *Medical Engineering and Physics*, 3:235–241, 1997.
- [71] T. Koornwinder. Two-variable analogues of the classical orthogonal polynomials. In R.A. Askey, editor, *Theory and Application of Special Functions*. Academic Press, New York, 1975.
- [72] S. Krist and T. Zang. Numerical simulation of channel flow transition, resolution requirements and structure of the hairpin vortex. Technical Report 2667, NASA, LaRC Hampton, VA, 1987.
- [73] J.F. Ladisa, I. Guler, L.E. Olson, D.A. Hettrick, J.R. Kersten, D.C. Warltier, and P.S. Pagel. Three-dimensional computational fluid dynamics modeling of alterations in coronary wall shear stress produced by stent implantation. *Annals of Biomedical Engineering*, 31(8):972–980, 2003.
- [74] Y. G. Lai, K.B. Chandran, and J. Lemmon. A numerical simulation of mechanical heart valve closure fluid dynamics. *Journal of Biomechanics*, 35:881–892, 2002.
- [75] H. L. Leo, Z. He, J.T. Ellis, and A.P. Yoganathan. Microflow fields in the hinge region of the CarboMedics bileaflet mechanical heart valve design. *Journal of Thoracic and Cardiovascular Surgery*, 124:561–574, 2002.
- [76] V. Leytin, D.J. Allen, S. Mykhaylov, L. Mis, E.V. Lyubimov, B. Garvey, and J. Freedman. Pathologic high shear stress induces apoptosis events in human platelets. *Biochemical and Biophysical Research Communications*, 320(2):303–10, 2004.
- [77] S. Lin and R.B. Rood. An explicit flux-form semi-Lagrangian shallow-water model on the sphere. *Quarterly Journal of Royal Meteorological Society*, 123:2531–2533, 1997.

- [78] J. S. Liu, P. C. Lu, and S. H. Chu. Turbulence characteristics downstream of bileaflet aortic valve prostheses. *Journal of Biomechanical Engineering- Transactions of the ASME*, 122:118–124, 2000.
- [79] F. Loth, N. Arslan, P.F. Fischer, C.D. Bertram, S.E. Lee, T.J. Royston, R.H. Song, W.E. Shaalan, and H.S. Bassiouny. Transitional flow at the venous anastomosis of an arteriovenous graft: potential relationship with activation of the ERK1/2 mechanotransduction pathway. *Journal of Biomechanical Engineering*, 125, 2003.
- [80] J.W. Lottes and P.F. Fischer. Hybrid multigrid/Schwarz algorithms for the spectral element method. *Journal of Scientific Computing*, 24(1):45–78, 2005.
- [81] R. Loy. Autopack user manual, 2000. <http://www-unix.mcs.anl.gov/autopack/>.
- [82] Y. Maday, A.T. Patera, and E.M. Rønquist. An operator-integration-factor splitting method for time-dependent problems: Application to incompressible fluid flow. *Journal of Scientific Computing*, 5(4):263–292, 1990.
- [83] D. M. McQueen and C. S. Peskin. Computer-assisted design of pivoting disc prosthetic mitral valves. *Journal of Thoracic and Cardiovascular Surgery*, 86:126–135, 1983.
- [84] M. Menegatti, G. Cristalli, L. Gallo, P.M. Mannucci, and F.I. Pareti. Effect of adenosine derivatives on in vitro thrombus formation induced by shear stress. *Haematologica*, 84(8):721–5, 1999.
- [85] J. Mohara, K. Kawahito, Y. Misawa, and K. Fuse. Evaluation of platelet damage in two different centrifugal pumps based on measurements of alpha-Granule packing proteins. *Artificial Organs*, 22(5):371–374, 1998.
- [86] P. Moin and K. Mahesh. Direct numerical simulation: A tool in turbulence research. *Annual Review of Fluid Mechanics*, 30:539–578, 1998.

- [87] Y. Morinishi, T. S. Lund, O. V. Vasilyev, and P. Moin. Fully conservative higher order finite difference schemes for incompressible flow. *Journal of Computational Physics*, 143:90–124, 1998.
- [88] R. D. Moser and P. Moin. The effect of curvature in wall-bounded turbulent flows. *Journal of Fluid Mechanics*, 175:479–510, 1987.
- [89] A. Okajima. Strouhal numbers of rectangular cylinders. *Journal of Fluid Mechanics*, 123:379–398, 1982.
- [90] S.A. Orszag and G.S. Patterson. Numerical simulation of three-dimensional homogeneous isotropic turbulence. *Physical Review Letters*, 28:76–79, 1972.
- [91] R. Paul, J. Apel, S. Klaus, F. Schugner, P. Schwindke, and H. Reul. Shear stress related blood damage in laminar couette flow. *Artificial Organs*, 27:517–529, 2003.
- [92] K. Perktold, M. Resch, and R.O. Peter. Three-dimensional numerical analysis of pulsatile flow and wall shear stress in the carotid artery bifurcation. *Journal of Biomechanics*, 24(6):409–20, 1991.
- [93] J.B. Perot. An analysis of the fractional step method. *Journal of Computational Physics*, 108(1):51–58, 1993.
- [94] L. Prandtl. On the motion of a fluid with very small viscosity. *Third International Congress of Mathematics, Heidelberg*, 1904.
- [95] S. Ramakrishnan and S. Collis. Multiscale modeling for turbulence simulation in complex geometries. *AIAA paper 2004-0241*, 2004.
- [96] J. M. Ramstack, L. Zuckerman, and L. F. Mockros. Shear-induced activation of platelets. *Journal of Biomechanics*, 2:113–125, 1979.
- [97] W.H. Reed and T.R. Hill. Triangular mesh methods for the neutron transport equation. *Los Alamos Scientific Laboratory Report LA-UR-73-479*, 1973.

- [98] J. Remacle. AOMD Web page, 2002. <http://www.scorec.rpi.edu/AOMD>.
- [99] J.F. Remacle, O. Klaas, J.E. Flaherty, and M.S. Shephard. Parallel algorithm oriented mesh database. *Engineering with Computers*, 18(3):274–284, 2002. <http://www.scorec.rpi.edu/AOMD/>.
- [100] A. M. Sallam and N. H. C. Hwang. Human red blood cell hemolysis in a turbulent shear flow: contribution of Reynolds shear stress. *Biorheology*, 21:783–797, 1984.
- [101] J. Schöberl, J.M. Melenk, C.G.A. Pechstein, and S.C. Zaglmayr. Additive Schwarz preconditioning for p-version triangular and tetrahedral finite elements. Technical Report 2005-10, Johann Radon Institute for Computational and Applied Mathematics (RICAM), 2005.
- [102] D. Schötzau, C. Schwab, and A. Toselli. Mixed hp-DGFEM for incompressible flow. *SIAM Journal on Numerical Analysis*, 40:2171–2194, 2003.
- [103] D. Schötzau, C. Schwab, and A. Toselli. Mixed hp-DGFEM for incompressible flows II: Geometric edge meshes. *IMA Journal of Numerical Analysis*, 24(2):273–308, 2004.
- [104] K. Shahbazi. An explicit expression for the penalty parameter of the interior penalty method. *Journal of Computational Physics*, 205(2):401–407, 2005.
- [105] K. Shahbazi, P.F. Fischer, and C.R. Ethier. A high-order discontinuous Galerkin method for the unsteady incompressible Navier-Stokes equations. *Journal of Computational Physics*, 2006. in press, doi:10.1016/j.jcp.2006.07.029.
- [106] A. Shapiro. The use of an exact solution of the Navier-Stokes equations in a validation test of a three-dimensional nonhydrostatic numerical models. *Monthly Weather Review*, 121:2420–2425, 1993.

- [107] R. Smith, E. Blick, J. Coalson, and P. Stein. Thrombus production by turbulence. *Journal of Applied Physiology*, 32:261–264, 1972.
- [108] P. Solin, P. Segeth, and I. Zel. *High-Order Finite Element Methods*. Studies in Advanced Mathematics, Chapman and Hall, Florida, 2004.
- [109] P. D. Stein and H. N. Sabbah. Measured turbulence and its effect on thrombus formation. *Circulation Research*, 35:608–614, 1974.
- [110] G. Szegő. *Orthogonal Polynomials*. American Mathematical Society, Providence, 1939.
- [111] C.A. Taylor, M.T. Draney, J.P. Ku, and D. Parker. Predictive medicine: computational techniques in therapeutic decision-making. *Computer Aided Surgery*, 4:231–247, 1999.
- [112] C.A. Taylor, T.J.R. Hughes, and C.K. Zarins. Finite element modeling of three-dimensional pulsatile flow in the abdominal aorta: Relevance to atherosclerosis. *Annals of Biomedical Engineering*, 26(6):975, 1998.
- [113] R. Temam. Une méthode d’approximation de la solution des équations de Navier-Stokes. *Bull. Soc. Math. France*, 98(6):115–152, 1968.
- [114] H. Tennekes and J. L. Lumly. *A first course in turbulence*. MIT Press, Cambridge, MA, 1972.
- [115] H.M. Tufo and P.F. Fischer. Terascale spectral element algorithms and implementations. *Supercomputing, ACM/IEEE 1999 Conference*, page 68, 1999.
- [116] H.M. Tufo and P.F. Fischer. Fast parallel direct solvers for coarse grid problems. *Journal of Parallel and Distributed Computing*, 61(2):151–177, 2001.
- [117] H.K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Longman Scientific and Technical, 1995.

- [118] T. Warburton and J.S. Hesthaven. On the constants in hp-finite element trace inverse inequalities. *Computer Methods in Applied Mechanics and Engineering*, 192(25):2765–2773, 2003.
- [119] T. Warburton, L.F. Pavarino, and J.S. Hesthaven. A Pseudo-spectral scheme for the incompressible Navier-Stokes equations using unstructured nodal elements. *Journal of Computational Physics*, 164(1):1–21, 2000.
- [120] F.M. White. *Viscous Fluid Flow*. McGraw-Hill New York, 1991.
- [121] D. Wilhelm and L. Kleiser. Stability analysis for different formulations of the nonlinear term in $P_N - P_{N-2}$ spectral element discretizations of the Navier–Stokes equations. *Journal of Computational Physics*, 174(1):306–326, 2001.
- [122] D. Xiu and G.E. Karniadakis. A semi-Lagrangian high-order method for Navier-Stokes equations. *Journal of Computational Physics*, 172:658–684, 2001.
- [123] A.P. Yoganathan, Z. He, and S. Casey Jones. Fluid mechanics of heart valves. *Annual Review of Biomedical Engineering*, 6(1):331–362, 2004.
- [124] A.P. Yoganathan, T.E. Jeffrey, M. H. Timothy, and G.P. Chatzimavroudis. Fluid dynamic studies for the year 2000. *Journal of Heart Valve Disease*, 7:130–139, 1998.