INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®
RICE UNIVERSITY

IMMERSED BOUNDARY METHODS
WITH APPLICATIONS TO FLOW CONTROL

by

Steven Kellogg

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science

Houston, Texas
October, 2000
RICE UNIVERSITY

Immersed Boundary Methods
With Applications to Flow Control

by

Steven Kellogg

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Master of Science

Approved, Thesis Committee:

S. Scott Collis, Chair
Assistant Professor of Mechanical Engineering

Marek Behr
Assistant Professor of Mechanical Engineering

Steven Cox
Professor of Computational and Applied Mathematics

Houston, Texas

September, 2000
Abstract

Immersed Boundary Methods
With Applications to Flow Control

by

Steven Kellogg

While some engineers use computers as a first line of attack on design problems, others are persistently making computers and their software faster and more capable of solving realistic problems. The technology used to build the microscale electronic components that makes computers fast is also used to construct micron-scale electromechanical (MEMS) actuators ideal for use in control schemes to reduce drag in industrial flows, promising millions of dollars in cost savings. The Immersed Boundary Condition (IBC) developed here augments a common fractional step, pseudospectral method used with Large Eddy Simulation to inexpensively and more realistically simulate turbulent flow over MEMS-like actuators. This is done by augmenting the numerical method to simulate flow over unsteady, irregular boundaries with static, structured rectilinear grids. The method is validated and applied to evaluate actuator characteristics and simulate open and closed loop flow control with continuous and discrete, MEMS-like actuators.
Acknowledgments

Tremendous thanks to my advisor, Dr. Collis. His ideas have been invaluable, his support unflagging, and his willingness to spend as much time as it takes generous. Dr. Collis has played a more active role than I could ever have hoped for in formulating the boundary condition, thinking up actuators and control schemes, documenting results, and even in the painful task of debugging. He has been gracious as I put my foot in my mouth several times over the years, he has taught me to think on a new level, and he has been a friend.

Thanks to Dr. Behr and Dr. Cox for their willingness to serve on my committee and their intelligent comments and suggestions.

My research group also deserves thanks: Yong Chang for his programming expertise and inexhaustible patience in answering my questions, Alex Dobrinsky for his uncanny willingness to invest his time and intellect in solving my problems, R.D. Prabhu for his confidence and ebullience in the quest for knowledge, and Zach Smith for his aid in the final days of generating results and his desire to continue this vein of research. Thanks, too, to all of my other friends who have supported me throughout this period, enduring both my transient exuberance and my incessant moaning.

My parents and Laura deserve special appreciation for their confidence, emotional support, and willingness to occasionally take lower priority than my work.

Thanks also to Pamela Haley and Jamaludin Mohd-Yusof for their intellectual contributions. This work has been supported by NASA grant NGT-1-52215 with computer time provided by the NASA-IT program and the CAAM Department at Rice University via NSF SREMS grant DMS-9872009 and LACSI through LANL contract number 03891-99-23 as part of prime contract W-7405-ENG-36 between the Department of Energy and the Regents of the University of California.
Nomenclature

Roman Symbols

\( A_i \)  Explicit terms in the momentum equations

\( B_i \)  Implicit terms in the momentum equations

\( B_I \)  Discrete volume integral operator

\( C \)  Coefficient of the SGS Smagorinsky model

\( C_L \)  Centerline

\( D \)  Discrete divergence operator

\( f \)  Body force or force distribution in the momentum equations or circular frequency of temporal oscillation in open-loop control schemes

\( f_s \)  Lagrange multiplier

\( g_s \)  Approximate Lagrange multiplier

\( G \)  Discrete gradient operator

\( H \)  Constraint operator

\( J \)  Force distribution matrix

\( k \)  Substep index in temporal discretization

\( M \)  Implicit operator in temporal discretization

\( \tilde{M} \)  Implicit operator modified with constraints

\( L_x \)  Channel length in \( x \)
\( L_z \) Channel width in \( z \)

\( l_x \) Actuator length in \( x \)

\( l_z \) Actuator pole width in \( z \)

\( m \) Iteration index

\( n \) Timestep index in temporal discretization

\( N_x \) Number of Fourier modes in \( x \)

\( N_y \) Number of finite difference points in \( y \)

\( N_z \) Number of Fourier modes in \( z \)

\( p \) Pressure

\( Q_i \) Collected pressure, convective, and diffusive terms in the momentum equations

\( r \) Right hand side of the temporally discretized momentum equations

\( \bar{r} \) Right hand side of the momentum equations modified with constraints

\( Re \) Reynolds number

\( s \) Numerical source term

\( SC \) Stability Constraint

\( S_iS_j \) Strain-rate tensor

\( t \) Time

\( t^+ \) Viscous time, \( t^+ = tu_z^2/\nu \)
\((u, v, w)\) Velocity components in Cartesian coordinates

\(u_i\) Velocity components in Cartesian coordinates

\(u_s\) Inhomogeneous portion of a constraint equation

\(\bar{U}\) Bulk flow

\(\tilde{u}\) Velocity field that does not satisfy constraints

\(u^d, u_i^d\) Desired velocity field

\(u_r\) Friction velocity, \(u_r = (\tau_w/\rho)^{1/2}\)

\(u^+\) Velocity in viscous units, \(u^+ = u/u_r\)

\((x, y, z)\) Cartesian coordinates

\(x_i\) Cartesian coordinates

\(y\) Deformation in the wall-normal direction

\(\dot{y}\) Deformation rate in the wall-normal direction

**Greek Symbols**

\(\alpha\) Streamwise wavenumber in Tollmien-Schlichting wave problems

\(\beta_i\) Coefficients for time discretization of momentum equations

\(\delta\) Half-height or mean half-height of the channel

\(\Delta\) Finite difference operator

\(\overline{\Delta}\) LES filter width

\(\epsilon\) A small scaling factor
\( \eta_i \) Coefficients for terms in the temporal discretization

\( \gamma_i \) Coefficients for terms in the temporal discretization

\( \mu \) Dynamic viscosity coefficient

\( \nu \) Kinematic viscosity coefficient, \( \nu = \mu / \rho \)

\( \nu_T \) Non-dimensional eddy viscosity in SGS model

\( \Omega \) Domain of a volume integral

\( \omega \) Wave frequency of Tollmien-Schlichting wave problems

\( \phi \) Pseudopressure approximation to \( p \)

\( \rho \) Fluid density

\( \tau_{ij} \) Stress tensor

\( \tau_w \) Average viscous drag

\( \xi \) Grid stretching factor

**Accents, Superscripts, and Subscripts**

\( (\cdot)_{max} \) Maximum value

\( (\cdot)_{rms} \) Root-mean-square value

\( (\cdot)^+ \) Wall coordinates or viscous units

\( \bar{(\cdot)} \) Mean or grid-filtered variables

\( \tilde{(\cdot)} \) Modified with constraints or not satisfying constraints
\hat{} \quad \text{Complex Tollmien-Schlichting wave coefficients, Fourier coefficients, or non-solenoidal fields}

\prime \quad \text{Disturbance quantity}

**Abbreviations**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>Aspect ratio</td>
</tr>
<tr>
<td>c.c.</td>
<td>Complex conjugate</td>
</tr>
<tr>
<td>DNS</td>
<td>Direct Numerical Simulation</td>
</tr>
<tr>
<td>IBC</td>
<td>Immersed Boundary Condition</td>
</tr>
<tr>
<td>LES</td>
<td>Large Eddy Simulation</td>
</tr>
<tr>
<td>MEMS</td>
<td>Micro-electromechanical System</td>
</tr>
<tr>
<td>NC</td>
<td>No-control</td>
</tr>
<tr>
<td>POD</td>
<td>Proper Orthogonal Decomposition</td>
</tr>
<tr>
<td>RMS</td>
<td>Root-Mean-Square</td>
</tr>
<tr>
<td>RK</td>
<td>Runge-Kutta</td>
</tr>
<tr>
<td>TS</td>
<td>Tollmien-Schlichting</td>
</tr>
</tbody>
</table>
# Table of Contents

Abstract .................................................................................................................. iii

Acknowledgments ................................................................................................. iv

Nomenclature ........................................................................................................ v

1 Introduction ........................................................................................................ 1
   1.1 Motivation .................................................................................................... 1
   1.2 Introduction to Flow Control ...................................................................... 3
   1.3 Heuristic Approach to the IBC .................................................................. 8
   1.4 Review of IBC Literature .......................................................................... 14
   1.5 Objectives and Accomplishments ............................................................. 21

2 Numerical Method for Turbulent Flow .............................................................. 23
   2.1 Geometry and Non-Dimensionalization ................................................. 23
   2.2 The State Equation and LES ................................................................. 24
   2.3 A Numerical Method for Channel Flow .................................................. 27
       2.3.1 Temporal Discretization .................................................................. 27
       2.3.2 Treatment of the Non-Linearity ...................................................... 29
       2.3.3 Fractional Step Method .................................................................. 30
       2.3.4 Computation of Pressure to Second Order Accuracy .................. 33
       2.3.5 Spatial Discretization and Grid Generation .................................. 34

3 Immersed Boundary Condition ......................................................................... 37
   3.1 Explicitly Constrained State Equation ...................................................... 37
   3.2 Implicitly Constrained State Equation ...................................................... 44
3.3 Implementational Details ......................................................... 47
  3.3.1 Details of the Explicit IBC .................................................. 48
  3.3.2 Details of the Implicit IBC .................................................. 49
  3.3.3 Use of the IBC ................................................................. 52

4 Validation Studies ................................................................. 54
  4.1 Laminar Flow in a Channel .................................................... 54
  4.2 Tollmien-Schlichting Wave Growth ........................................... 61
  4.3 Laminar Flow with Wall-Normal Acceleration .............................. 69
  4.4 Laminar Flow in a Grooved Channel ......................................... 74
  4.5 Turbulent Flow Statistics ..................................................... 80

5 Results .................................................................................. 86
  5.1 Experimental Setup .............................................................. 87
  5.2 Opposition Control with a Continuous Deforming Wall ................. 90
  5.3 Sinusoidal Membrane Actuator Study ....................................... 96
  5.4 Open Loop Actuator Parameter Study ...................................... 101
    5.4.1 Varying Frequency ........................................................... 103
    5.4.2 Varying Phase ................................................................. 107
    5.4.3 Varying Aspect Ratio ....................................................... 109
    5.4.4 Limiting Case — Sinusoidal Riblets ................................... 111
  5.5 Closed Loop Control ............................................................ 113
    5.5.1 Discrete Heuristic Control ............................................... 113
    5.5.2 Discrete Control with Spanwise Shear Sensors ....................... 118
  5.6 Summary ........................................................................... 121

6 Future Work and Conclusions ................................................. 124
  6.1 Future Work ..................................................................... 124
6.2 Conclusions ..................................................... 126

Bibliography ....................................................... 128
List of Tables

5.1 Drag Reduction for Continuously Deforming Wall Opposition Control . . 92
5.2 Drag Reduction Varying Frequency . . . . . . . . . . . . . . . . . . . . . . . 104
5.3 Drag Reduction Varying Phase . . . . . . . . . . . . . . . . . . . . . . . . . . 109
5.4 Drag Reduction Varying Aspect Ratio . . . . . . . . . . . . . . . . . . . . . 110
5.5 Drag Reduction for Sinusoidal Riblets . . . . . . . . . . . . . . . . . . . . 112
5.6 Drag Reduction of Closed Loop Discrete Heuristic Control Schemes . . 116
5.7 Drag Reduction Varying Gain Factor for Spanwise Shear Control . . . 120
List of Figures

1.1 Golf Ball ................................................. 2
1.2 Micro-Electromagnetic Flap Actuator ..................... 6
1.3 Immersed Boundary Concept ............................. 12
1.4 Velocity Profiles Satisfying IBC ......................... 13

2.1 Simulation Geometry ..................................... 24
2.2 Stretched Grid ........................................... 35
2.3 Staggered Grid ........................................... 36

3.1 Explicit IBC Forcing Schemes ............................ 48
3.2 Implicit IBC Forcing Schemes ............................ 50

4.1 Laminar Profiles with Explicit IBC ....................... 57
4.2 Laminar Velocity Profiles with Implicit IBC .............. 59
4.3 Body Force Profiles .................................... 60
4.4 Initial Disturbances for TS Wave Problem ................ 63
4.5 TS Wave Profiles ...................................... 67
4.6 Spatial and Temporal Convergence of IBC ................ 68
4.7 Shaking Channel Schematic .............................. 69
4.8 Shaking Channel Profiles ................................ 73
4.9 Riblet Schematic ....................................... 75
4.10 Riblet Convergence .................................... 77
4.11 Riblet Contours ....................................... 78
4.12 Turbulent Flow Schematic ............................... 80
4.13 Turbulent Flow Statistics ............................... 82
4.14 Turbulent Flow Statistics ............................... 83
4.15 Turbulent Flow Statistics ........................................... 84
4.16 Turbulent Flow Statistics ........................................... 85

5.1 Continuously Deforming Wall Opposition Control ............... 91
5.2 Drag History for Continuously Deforming Wall Opposition Control .... 93
5.3 Statistics for Continuously Deforming Wall Opposition Control .... 95
5.4 Wall Image for Continuously Deforming Wall Opposition Control .... 96
5.5 Monopole and Dipole Membrane Actuators ......................... 97
5.6 Flowfield for Steady Actuator Study ................................ 100
5.7 Array of Dipole Membrane Actuators ................................ 102
5.8 Drag Histories Varying Frequency ................................... 104
5.9 Statistics for Open Loop Control Varying Temporal Oscillation Frequency 105
5.10 Out of Phase Array of Dipole Membrane Actuators ................ 108
5.11 Drag Histories Varying Phase ...................................... 109
5.12 Dipole Actuators with $AR = 6$ and $AR = \infty$ ................... 110
5.13 Drag Histories for Varying Actuator Aspect Ratio .................. 111
5.14 Drag Histories for Stationary Sinusoidal Riblets .................. 112
5.15 Array of Deformed Dipole Actuators ............................... 115
5.16 Drag Histories for Discrete Heuristic Control Cases ............... 116
5.17 Statistics for Opposition Control with Discrete Dipolar Actuators .... 117
5.18 Schematic of Discrete Control with Spanwise Shear Sensors ......... 119
5.19 Drag Histories for Discrete Control with Spanwise Shear Sensors .... 121
5.20 Statistics for Discrete Control with Spanwise Shear Sensors........ 122
Chapter 1

Introduction

1.1 Motivation

Computer modeling of physical phenomena has become incispensable to the engineering process since computers first became commercially available in 1950 [23]. Computer simulations are frequently the first line of attack on any engineering problem. Automotive and aerospace engineers, for example, use computers to determine the shapes of their planes and cars, the vibration characteristics of their structural members, and the space limitations for each part, sometimes years before a physical model is built. Meanwhile, other engineers are persistently advancing the state of the art in making computers and their software faster and more capable of solving realistic problems.

This research investigates the Immersed Boundary Condition (IBC), one advancing front in the push to make simulations faster and more realistic. Numerical simulations of the flow of fluids such as air and water over irregular surfaces have typically relied on boundary-fitted grids, thus allowing the use of standard boundary treatments to discretely satisfy analytical boundary conditions. For moving, irregular boundaries, the expense of adjusting unstructured or regenerating structured grids each time the boundary moves is often as expensive as solving the governing equations themselves [9]. The Immersed Boundary Condition, however, eliminates this expense by cleverly augmenting the numerical method to simulate flows over irregular boundaries on an unchanging, structured rectilinear grid.

The Immersed Boundary Condition will be used here as a tool to facilitate research into flow control. Flow control is a coordinated disturbance of the state of a fluid
Figure 1.1: (a) The dimples on a golf ball add energy to the boundary layer, delaying flow separation. As a result, the low pressure wake behind a golf ball has a smaller cross-sectional area and pressure drag on the golf ball is lower. Figure adapted from [91]. (b) Likewise, adding roughness elements to the front of a bowling ball dropped in water reduces the size of the low pressure wake. Figure taken by U.S. Navy, Ordnance Test Station, abstracted from [128].

to achieve a desired effect. For example, the dimples on a golf ball (Figure 1.1) intentionally introduce disturbances in the surrounding air during flight in order to reduce the size of the ball’s wake, thus reducing the amount of energy the ball forfeits to the flow around it and allowing it to fly farther [128]. Or, very small flaps might be used to counteract acoustic disturbances in the air [40]. By easing the computational expense of simulating flows over irregular and moving surfaces, the IBC facilitates investigation of such flow control problems.

The class of flow control problems addressed in this work have emerged only in the last decade with the advent of manufacturing techniques capable of building microscopic mechanical devices such as pumps and flaps that operate on the same spatial scale as the velocity disturbances in turbulent flows, the high-drag flows common in everyday engineering applications. These micro-electromechanical systems — or MEMS — are precisely the irregular and deforming surfaces we intend to use the
CHAPTER 1. INTRODUCTION

Immersed Boundary Condition to model.

In short, we develop the Immersed Boundary Condition as a tool to speed and simplify the simulation of turbulent flow over unsteady, irregular surfaces such as flat walls covered with moving MEMS devices. By making simulation of these complicated flows more practicable, in this work we are able to explore flow control using MEMS-like actuators with a realism never before accomplished.

This chapter presents a brief introduction to flow control first, followed by a conceptual overview of the Immersed Boundary Condition. Then, with the fundamental concepts of the IBC in hand, the work of others regarding the Immersed Boundary Condition is reviewed, highlighting the development of the method and the differences between the formulations of others and the one developed in detail in Chapter 3. The successes of this research are briefly outlined before embarking, in Chapter 2, on a description of the discrete mathematics used to solve the Navier-Stokes equations, a complete derivation of the IBC in Chapter 3, and a set of validation studies to establish the accuracy of the method in Chapter 4. New results are presented in Chapter 5, followed by a brief wrap-up suggesting future avenues of exploration in Chapter 6.

1.2 Introduction to Flow Control

To reiterate, flow control is a coordinated disturbance of the state of a fluid to achieve a specific goal. Possible goals include: the enhancement of air and fuel vapor mixing in an internal combustion engine to improve efficiency; the reduction of drag on a streamlined body; the suppression of noise generation in a fluid by mechanical components such as rotorcraft wings, airplane jet engines, and submarine propellers; and the augmentation or suppression of heat transfer between fluids and solids. The means of controlling a flow may be passive, such as the dimple patterns on golf balls, or active, such as the use of small flaps to remove disturbances from a turbulent flow.
CHAPTER 1. INTRODUCTION

At low speeds, in the laminar regime, fluids move in a predictable manner. The effects of the natural stickiness, or viscosity, of the slowly moving fluid easily overcome the inertial effects, or those of its momentum. The non-dimensional parameter called Reynolds number quantifies the relative effects of inertial and viscous forces; fluids with Reynolds numbers less than about $10^3$ are typically laminar. For less viscous or faster moving fluids, the flow enters a turbulent regime and the motion of the fluid becomes chaotic and unpredictable except in a statistical sense. Viscous effects no longer dominate the flow, and transient coherent structures emerge and decay irregularly. These coherent structures, eddies aligned with the flow called rolls and low momentum jets of fluid called streaks, constantly mix the flow. In turbulent flow over a streamlined body, these coherent structures move low momentum fluid from near the body out into the flow and high momentum fluid from the flow closer to the body, increasing drag and ultimately dissipating energy from the body’s motion into the flow [127]. Higher drag costs additional electricity to pump a fluid through a pipe and additional fuel to move a plane through the air. Abating the effects of the coherent structures in turbulence is a popular goal of flow control because turbulent flow is ubiquitous in engineering applications.

According to Gad-el-Hak [40], flow control predates its name by well over a millennium with the discovery that placing fins on the tails of arrows and spears produces a dramatic improvement in their flight stability. With Prandtl’s pioneering work in 1904 on boundary layers — the portions of a flow dramatically affected by the relative motion of a fluid and a solid body — the empirical age of flow control dawned. By the 1940’s, the military applications of flow control dominated the field, with researchers exploring means of keeping flow around objects in the low drag, laminar flow regime to improve the speed, maneuverability, and efficiency of aircraft, missiles, ships, submarines, and torpedoes. Beginning in the 1970’s, a new thrust for flow
control emerged, this time with the goal of improving energy efficiency by reducing drag in turbulent flows around airplanes, cars, and in industrial pipelines. Of late, with computer technology driving engineers to create ever smaller electronic components for computer chips, manufacturing techniques have emerged capable of building microscopic mechanical devices such as pumps and flaps that can have a localized impact on the coherent structures in turbulent flows. These micro-electromechanical systems, or MEMS, have made flow control practicable in a whole new range of problems [38, 56, 84, 89], encouraging engineers to develop active control strategies for turbulent flow — sensing the fluctuations in the velocity field and then using the actuators, components that modify the flow, in the right way to accomplish a given goal, such as moving or removing turbulent coherent structures [40].

Methods of flow control are generally classified into active and passive, open loop and closed loop schemes [89]. Passive control schemes have stationary actuators like the dimples on a golf ball; active schemes employ moving actuators, such as jets, flaps, and dynamic body forces. In open loop control, the actuators act independently of the flow. In closed loop control schemes, sensors gather information about the flow state and actuators react accordingly.

The relative simplicity of passive control methods has generated considerable research interest, for example the work of Bushnell and Hefner [16] and Pollard [103] on viscous drag reduction in boundary layers. Careful design of a body's geometry, for example to manipulate the pressure distribution in the flow over it and delay transition to turbulence [105], is also passive control. So is the placement of longitudinal grooves, or riblets, on a surface to interfere with the coherent structures in turbulence and reduce drag [8, 25]. Polymer surface coatings for drag reduction are also an effective passive control scheme [27, 118, 119].

The efficacy of passive control schemes is limited, however, due to the complex
evolution and structure of laminar flows, not to mention the chaotic nature of turbulent flows. In open loop, active control, moving actuators [40,90] are preprogrammed to move in a way that makes them more effective, e.g. in Section 5.4. Examples of active actuators include synthetic jets [3,112] and miniature moving flaps like the one shown in Figure 1.2 [20,57]. Closed loop, or feedback, active control [13,40,89] improves the potential effectiveness of control even more by tailoring actuator motion to the state of the flow at a given instant. MEMS sensors that detect wall shear, for example those described in [89], may be used to obtain information about the flow and intelligently direct the actuator motion.

Feedback control can be classified into roughly four groups based on how sensor input is translated into actuator output [89]: physical model-based, adaptive, dynamical system-based, and optimal control.

A good example of the use of flow physics knowledge to design fluid control schemes is the active cancellation of disturbances, e.g. the selective suction of Gad-el-Hak [39] or the opposition control scheme of Choi, Moin and Kim [26]. In these methods, the
motion of high momentum fluid in the flow towards a solid is opposed with jets of fluid at the solid surface that are equal and opposite in velocity. Simulations of this scheme demonstrate about 25% drag reduction attributed to abating the rolls that carry high momentum fluid closer to the surface [22, 26, 54, 69].

Adaptive control schemes, on the other hand, may know nothing of the physics of the flow; algorithms experiment with transfer functions between sensor input and actuator output, gradually learning what is most effective. Neural networks, for example, are tools that learn by experimentation. The network represents a transfer function by manipulating the input values with various analytical functions and linear combinations to give as many output values as are needed. The weights of the linear combinations are successively adjusted with an optimization algorithm until the inputs are mapped to the outputs in the best possible way [36, 59]. In research by Lee et al. [74], a neural network was constructed and trained to emulate the opposition control scheme developed by Choi et al. [26] described above. Numerical simulations of turbulent channel flow indicate that this control scheme gives about 20% drag reduction. A detailed description of adaptive techniques and the applications of neural networks to turbulence control can be found in [13, 74, 89].

In dynamical system-based control, the first step is to mathematically simplify the flow. This is done by constructing a series of functions — or bases — that may be scaled and added together to approximate the vector-valued analytical function that describes the flow exactly [5]. Basis functions may be increasingly high frequency sines and cosines, for example, or discrete functions derived in such a way that each successive function contains the maximum amount of information about the flow possible, as in the case of Proper Orthogonal Decomposition (POD) [11, 82]. Once the basis functions are determined, a few representative modes are chosen and used as a reduced order model for the flow [58]. This simplified model may then be used
to find the controls that are most effective in accomplishing the objective [28], for example with the Ott-Grebogi-Yorke (OGY) method, which has been conceptually successful in numerical experiments [40, 89]. POD is also a useful tool in analyzing control schemes as shown in work by Prabhu et al. [104].

The final category of closed loop, active control schemes is optimal control, an exceptionally effective tool in computational fluids, but impractical for real-time, closed loop control. The first step in optimal control is to define a function describing the control objective. Then, a mathematical framework is derived that predicts the most effective control inputs for accomplishing the control objective [1]. Optimal control schemes have proven incredibly effective at reducing drag in numerical simulations [12, 21], even relaminarizing the flow for low Reynolds numbers. The drawback of optimal control theory is that it requires not just information about the flow obtained from an array of sensors, but complete knowledge of the flow, thus preventing its use as an on-line feedback control algorithm.

1.3 Heuristic Approach to the IBC

In this work, the Immersed Boundary Condition (IBC) is used to facilitate modeling of MEMS actuators for flow control in numerical simulations. This section is intended to introduce the method in a heuristic way to elucidate the subsequent review of literature.

The motion of a fluid is governed by the Navier-Stokes equations. For the case of an incompressible fluid, the Navier-Stokes equations are derived using Hooke's law and conservation of momentum, then simplified with conservation of mass. For incompressible, Newtonian fluids with constant viscosity, the non-dimensionalized
Navier-Stokes — or momentum — equations can be written with the Einstein summation convention in Cartesian tensor notation as

\[ u_{i,t} + (u_i u_j)_{,j} - \frac{1}{Re} u_{i,jj} + p_{,i} + f_i = 0 \]  \hspace{1cm} (1.1)

and are closed by an expression of conservation of mass, the continuity equation:

\[ u_{i,i} = 0. \]  \hspace{1cm} (1.2)

Here, \( u_i \) represents the three components of velocity, \( p \) represents pressure, and \( Re \) is the Reynolds number, a non-dimensional parameter that quantifies the relative effects of viscous forces and inertial forces.

The Navier-Stokes equations also include the possibility that fluids are subject to a momentum source, or body force \( f_i \). Gravity, for example, is a body force, as are magnetic and electrodynamic effects. Some actuators in the context of flow control produce body forces, as well [29, 79, 92]. For example, salt water has unique properties that allow engineers to manipulate it with body-force actuators, real world applications of which might be submarine propulsion [86] or the removal of the acoustic waves from the salt water very close to a military submarine hull to improve sonar listening capability [40].

To see how body forces may be used in flow control and ultimately how they are used in the IBC, consider a simulation of flow over an airfoil covered with an array of flush actuators capable of generating an arbitrary force-field in the surrounding fluid. Suppose that the flow has a known, turbulent state at time \( t_{n-1} \) and that we want to use the body force actuators to effect a specific velocity profile \( u^{\text{desired}} \), or \( u^d \), from the next time-step forward.

Simplifying (1.1) by giving the name \( Q_i \) to the convective, diffusive, and pressure
terms, the momentum equation becomes

\[ u_{i,t} + Q_i + f_i = 0, \quad (1.3) \]

with \( Q_i = (u_i u_j)_{,j} - \frac{1}{Re} u_{i,jj} + p_{,i} \). Using superscript \( n - 1 \) for the state at \( t_{n-1} \) and superscript \( n \) for the state at \( t_n = t_{n-1} + \Delta t \), discretization in time with Crank-Nicholson gives

\[ \frac{u_i^n - u_i^{n-1}}{\Delta t} + \frac{1}{2} [Q_i^n + f_i^n + Q_i^{n-1} + f_i^{n-1}] = 0. \quad (1.4) \]

It is now easy to see what force the actuators must apply to get the desired state. Plugging in \( u_i^d \) for \( u_i^n \) and solving for the force,

\[ f_i^n = -2 \frac{u_i^d - u_i^{n-1}}{\Delta t} - [Q_i^d + Q_i^{n-1} + f_i^{n-1}]. \quad (1.5) \]

Employing this force distribution in the discrete state equation to advance from state \( n - 1 \) to state \( n \) guarantees the desired velocity profile at time \( t_{n-1} + \Delta t \). Note that as long as \( u_d \) is known on the entire computational domain, the term \( Q_i^d \) is also known; pressure may be computed by solving the Poisson equation that results from taking the divergence of the momentum equations. This procedure can be used at every time step with any \( u^d \) that satisfies the continuity equation \((1.2)\). We could even choose to specify the \( u^d \) on part of the domain and solve Navier-Stokes without a body force everywhere else. This is the very concept we use to implement an Immersed Boundary Condition \([121]\). In some regions of the computational domain, we use a body force to select the velocity profile desired at the next time step. In other regions of the domain, the state equations are solved without a body force. This, however, presents a complication in the calculation of the body force; in the example above, the term \( Q_i^r \) is known \textit{a priori} only if the entire velocity field is specified in \( u_d \). Otherwise, the body force must be calculated implicitly or with an iterative scheme as discussed in
Chapter 3.

In simulations of flow over a deformed surface, conventional methods dictate choosing the computational domain to include only the space occupied by the fluid. No-slip and no-penetration boundary treatments are applied at the edges of the computational domain — the interface between the fluid and the solid. In an Immersed Boundary Condition simulation, the computational domain is chosen before the simulation begins as a convenient, regular shape that contains both fluid and solid regions. The equations of motion for a fluid, however, are solved on the entire domain, both in portions that contain a fluid and in those that should contain a solid. The portion of the domain that should contain a solid is generally termed the virtual solid, and the interface between the fluid and the virtual solid is termed the virtual interface or virtual boundary. The Immersed Boundary Condition relies on the fact that if there is no fluid motion relative to the virtual interface — i.e. there is no slip and no penetration at the virtual boundary — the velocity and pressure distributions in the portion of the domain containing the fluid will be the same in the limit as if the equations of motion were solved in the conventional way. The IBC treatment maintains the discrete no-slip, no-penetration boundary condition at a virtual interface with a body force.

To illustrate this concept, Figure 1.3a shows streamlines over a circular cylinder. At the interface between the solid and the liquid, the fluid velocity both tangential and normal to the surface is zero. In Figure 1.3b, the entire domain contains a fluid, and the interface between the liquid and the solid is replaced by a virtual interface. If a body force is employed that constrains the components of velocity normal and tangential to the virtual interface to be zero, the streamlines around the virtual solid must be identical to those in Figure 1.3a [46]. The flow inside the virtual solid can be disregarded, as it has no physical meaning.
Figure 1.3: An entirely fluid domain can be used to model flow over a circular cylinder as long as the all-fluid domain has a no-slip and no-penetration boundary condition at the virtual interface — the location of the fluid-solid interface in the traditional simulation. This concept is commonly used in potential flow, as shown above, to model a cylinder in uniform flow by overlapping a uniform stream with a doublet to satisfy a penetration-free boundary, as in [128].

When the virtual interface falls on a grid site, we impose a body force at this location and set \( u^d \) to satisfy the no-slip, no-penetration boundary condition. Otherwise, since flow in the virtual solid is meaningless, we impose a body force in this region and set \( u^d \) such that interpolation consistent with the numerical method to the virtual interface satisfies the boundary conditions. For example, as shown in Figure 1.4, the components of velocity tangential to the virtual interface can be reversed across the interface to create a zero crossing and a smooth velocity profile [87,88]. Likewise, the normal component of velocity is reflected and forced such that interpolation at the interface is zero, satisfying the no-penetration condition.

The normal component of velocity is reflected instead of reversed in order to satisfy the continuity equation (1.2); for example, in an orthogonal coordinate system with the \( x_{i1} \) and \( x_{i2} \) axes tangential to a flat virtual interface and the \( x_n \) axis normal to
Figure 1.4: A no-slip, no-penetration boundary condition could be satisfied, for example, by antisymmetric reflection of \( u \) and \( w \) at the virtual wall and a symmetric reflection of \( v \). To maintain these profiles, a body force is typically employed at several grid sites immediately inside the virtual solid. Additional forcing is sometimes applied to maintain smooth velocity profiles at the first grid site outside of the virtual solid if the virtual interface is moving.

the surface, the continuity equation reads

\[
\frac{\partial u_{t1}}{\partial x_{t1}} + \frac{\partial u_{t2}}{\partial x_{t2}} + \frac{\partial u_n}{\partial x_n} = 0. \tag{1.6}
\]

Because \( u_{t1} \) and \( u_{t2} \) are zero and do not change along the virtual wall, \( \frac{\partial u_{t1}}{\partial x_{t1}} = \frac{\partial u_{t2}}{\partial x_{t2}} = 0 \). As a consequence, \( \frac{\partial u_n}{\partial x_n} \) must also be zero in order to avoid creating a mass source or sink; therefore we reflect the normal velocity component. Forcing is typically employed at several grid sites immediately inside the virtual solid to avoid discontinuous velocity profiles and their adverse effect on numerical methods. For the same reason, additional forcing is sometimes applied to maintain smooth velocity profiles at the first grid sites outside of the virtual solid if the virtual interface is moving or highly irregular. The velocity profile at the first grid site outside of the virtual interface would be forced to give the proper discrete boundary condition when interpolated.
between its neighbor and the virtual interface. Clearly, the position of the virtual boundary must be known at all times in this formulation to know where to apply the boundary treatment. Details can be found in Chapter 3

1.4 Review of IBC Literature

Having gained a rough understanding of the Immersed Boundary Condition from the heuristic explanation of Section 1.3, this section provides an academic and historical context for the method developed later in this work. Literature on IBC has become plethoral, particularly since 1990, and covering all of the work in which the method has been developed and applied is a non-trivial task. The goal of this section is to highlight similarities and differences between several major formulations and reference a few applications of each.

Without strong correlation to the distinctions between them, the Immersed Boundary Method, Virtual Boundary Method, Fictitious Domain Method, Embedded Domain Method, and Immersed Interface Method are all names that have been applied to numerical schemes with several features in common:

1. The equations governing fluid flow are solved on the entire computational domain, those that contain fluid and solid alike;

2. The effect of solids on the flow is introduced through a momentum-source like term in the fluid governing equations; and,

3. The computational grid on which the fluid governing equations are solved is not deformed to match the motion or specific geometry of solids in the flow.

Methods of this type can be roughly divided into two camps: those that prescribe the motion of liquid-solid interfaces in the flow and those that determine the motion
of the interface by solving governing equations for both the fluid and the solid. We discuss them in roughly chronological order, grouped by formulation.

The first virtual boundary method appeared in a 1969 paper by J.A. Viecelli [122]. Viecelli based his method on the marker and cell techniques [126] previously used for simulating the motion of free surfaces and interfaces between fluids. He simply handles all interfaces as free surfaces and treats pressure at the interface such that the fluid moves tangentially to the solid surface, in this way simulating a free-slip boundary. Viecelli solved the 2-D problems of a slug of fluid falling onto the bottom of a semi-circular tank and plane supercavitating flow around a disk. Viecelli later presented an improved method for solving the governing equations, as well as examples of jet flows impinging on a wedge and an angled flat plate [123]. The last notable appearance of Viecelli's method was in 1971.

Whereas Viecelli's method is convenient for solving problems where the location of the liquid-solid interface is known at all times, the immersed boundary method introduced in 1972 by Charles Peskin [97] is useful for problems in which neither the motion of the fluid nor the solid boundaries is known, only the initial state of the fluid and the solid. Thereafter, the equations of motion for the fluid are used in conjunction with the governing equations for an elastic body to determine the liquid-solid interaction. The equations of motion of the fluid are solved on the entire computational domain; the effect of the solid on the flow is introduced through the body force term in the Navier-Stokes equations. The boundary force between the liquid and solid is given by the governing equations for the solid. This boundary force is then spread onto the fluid's computational domain with a discrete delta function and used in the Navier-Stokes equations to calculate a fluid velocity field. The velocity field is then interpolated to the locations of the interfaces where the no-slip condition dictates that this is the velocity of the virtual surface. With the velocity
of the surface determined, the boundary location may be updated and Hooke's law governing the motion of elastic bodies gives the force at the liquid-solid interface, which is again spread onto the fluid domain with the discrete delta function. In short, the equations of motion of the fluid must be solved, along with the equations of motion of an elastic solid. The two are coupled by the no-slip condition relating the motion of the fluid and solid and the force term relating the stresses in the elastic material to the interfacial force. A discrete delta function spreads the interfacial force to the fluid domain; interpolation is used to approximate the motion of the fluid at the virtual interface.

A prime application of Peskin's method is to biological flows, for example blood passing through the mitral valve of the heart; the mitral valve leaflets are passably modeled as elastic membranes. Requiring 1.5 hours to simulate a single heartbeat on a 2-D, 20x20 mesh, Peskin was able to show results with qualitative correspondence to experiment [97]. Forgoing successive over-relaxation to solve the pressure Poisson equation, in [98] Peskin took advantage of his time-invariant, rectilinear grid to employ a fast Laplace solver. Solving coupled fluid-solid system equations at each time step with a Newton iteration, adding an improved discrete delta function, and using a more sophisticated spring-dashpot model for muscle fiber, Peskin used a 64x64 grid to simulate an entire cross section of the heart, valves and heart wall included. By representing the interfacial forces as derivatives of an energy function, Peskin and McQueen were able to simulate the interaction of fluid and stiff solids in [99], modeling flow around several types of artificial heart valves, albeit still in 2-D and not yet at truly physiological Reynolds numbers. In 1989 [100,101], the method was applied in 3-D simulations, testing models of elastic and contractile fiber models for use in a complete model of the heart. A pseudo-spectral/upwinded finite difference method was employed with the immersed boundary method in [4] to model arteriolar fluid
flow and advection and diffusion of nitric oxide. Peskin and co-workers also did work on honing the numerics themselves, comparing explicit, implicit, and semi-implicit calculation of the body forces in [120], improving satisfaction of continuity in [102], addressing the issues of vectorization of the method in [85], exploring order of accuracy and numerical viscosity in [71], and including adaptive mesh refinement [106].

Peskin's basic formulation has been applied independently by others to higher Reynolds number fluid-solid interaction [77, 78], 2-D simulations of the inner ear [14], models of cytoskeletal material that exhibit realistic non-Newtonian behavior [15], deformation of blood cells [34], and simulations the mitral valves in the heart that led to conclusions about the cause of systolic anterior motion disease [129]. Outside of the biomedical field, Peskin and Lai [71] have applied the method to flow over a stationary cylinder by assuming the cylinder walls may deform elastically about their original position. Others have modeled aquatic animal locomotion [30, 37] with the method. Sulksy and Brackbill [117] have used it for suspension flows, and Stockie [115, 116] has simulated flexible fiber suspensions. Laveque and Li [80] developed a slight variant on Peskin's method applicable to problems with discontinuous solutions, for example, at the interface between fluids in two-phase flows. In the way of more exotic applications, Lapenta [72, 73] has applied Peskin's immersed boundary technique to the simulation of plasma flows with unusual geometries and dielectric dust particles.

In 1993, Goldstein, Handler and Sirovich [50] presented an alternative to Peskin's method more appropriate for flows around bodies with prescribed locations. The body force is still spread from the location of the virtual interface to nearby grid points via a discrete delta function, but the interfacial force itself is calculated from a feedback loop rather than state equations of an elastic solid:

\[
F(x_b, t) = \alpha \int_0^t [U(x_b, \tau) - U^d(x_b, \tau)] d\tau + \beta[U(x_b, t) - U^d(x_b, t)],
\] (1.7)
where $U$ is the velocity interpolated to the virtual interface, $U^d$ is the desired velocity of the virtual interface, and $\tau$ is a placeholder variable for time. The parameters $\alpha$ and $\beta$ must be hand-selected to give reasonable results. Goldstein's method employs what would be termed a Proportional-Integral (PI) scheme in control theory to adjust the body force based on error in the boundary condition. The $\beta$ term increases the force applied proportional to the error and the $\alpha$ term increases force proportional to the time-integral of the error. Thus, the body force becomes increasingly strong as error persists. The force is updated at each time step. In unsteady fluid simulations, it is clear that there will always be error in the boundary treatment at the virtual interface. Furthermore, oscillation in the body force due to the spring-like feedback loop introduces unnatural time scales in the flow. Nevertheless, this method is employed effectively in simulations of a startup laminar flow, startup flow around a cylinder, and flow over both flat and ribbed walls in turbulent channel flow [50]. The authors note that although there is some error in their no-slip walls due to the time response of the forcing from the feedback loop, it does not affect their statistical results. Their later work studies two-dimensional synthetic jets [76] and turbulent flow over riblets [51].

Saiki and Biringen adopt Goldstein's method to simulate vortex shedding in uniform flow over a cylinder at low Reynolds numbers [107] and the effect of a spherical particle on transition in the boundary layer on a flat plate [108].

Another immersed boundary method suitable for flow over solids with known locations and trajectories was introduced by Mohd-Yusof in 1997 [35,87]. The method of Mohd-Yusof was used as the heuristic explanation in Section 1.3. Body forces are applied primarily on the virtual solid and chosen to satisfy the immersed boundary condition at every time step while alleviating as many computational difficulties as possible. The body force at each time step is found as described in Section 1.3,
essentially by rearranging the momentum equations and preliminarily calculating the state at the next time step using the desired velocity field. Mohd-Yusof initially applied his method to solve for drag on riblets in a laminar channel [87], comparing results to the solution of Choi, Moin, and Kim on a body-fitted grid [24]. Mohd-Yusof’s simulation converges impressively to the proper drag in the channel with considerably less resolution in the spanwise direction than the body-fitted simulation. In later work [88], Mohd-Yusof extends his method to moving geometries, simulating turbulent flow in a channel over unsteady walls. The unsteady walls are simply flat walls with four areas replaced by circular bumps of sinusoidal amplitude that oscillate in time. Results, however, were preliminary, and Mohd-Yusof does not cite drag-reduction numbers.

Verzicco et al. [35, 121] used Mohd-Yusof’s method on the problem of an axisymmetric cylinder with a moving piston. Verzicco et al. claim to be the first to apply the immersed boundary method to a real flow of practical interest that includes moving, complex boundaries and incorporates a turbulence model. Comparison of results for mean and root-mean squared velocity profiles with experimental measurements show good correspondence.

The final formulation addressed in this review is that of Glowinsky [44, 46, 47], which elegantly unifies the approaches of Peskin and Mohd-Yusof. Glowinsky derives his method in the variational form, giving him a solid framework for mathematical analysis of the immersed boundary technique to determine, for example, a priori error estimates [43]. In [47], Glowinsky begins with the problem of a single, stationary solid completely immersed in a fluid. He then formulates two problems: The first is on the portion of the domain occupied by the fluid, simply that the state equation of the fluid must be obeyed. The second is on the domain of the solid, stating that the governing equation of the fluid must be obeyed there as well, and that on the
boundary of the solid, the fluid velocity must be zero. Glowinsky then writes the
two systems in variational form and reformulates them as a single problem — the
state equations, which must be satisfied on the entire domain subject to a Lagrange
multiplier, and a constraint equation that the fluid velocity on the surface of the
virtual solid must go to zero. The Lagrange multiplier, he points out, plays the role
of a body force. Using a conjugate gradient algorithm to solve the sample problem
of a sphere in Stokes flow, Glowinsky and co-workers show solutions to be second
order accurate. Glowinsky et al. then use operator splitting to solve the 2-D Navier-
Stokes equations for flow over disks and elliptic airfoils [46, 47]. Glowinsky points
out further that the immersed boundary technique is ideal for shape optimization
problems because the cost of regridding is eliminated when the virtual solid changes
shape during a simulation. To illustrate, he formulates and solves a problem that
creates an airfoil with a given surface pressure distribution [46].

In [48], Glowinsky et al. extend the fictitious domain approach to moving bod-
ies with given trajectories in an incompressible, viscous flow, illustrating with the
example of a moving, solid disk. The method described in [48] is remarkably like a
variational analog of what is described in Section 3.1, except that we do not con-
strain rigid body motion on the virtual solid. Finally, in [44, 45, 49], the loop be-
tween Mohd-Yusof and Peskin's formulations is closed as Glowinsky et al. use the
Lagrange-multiplier/constraint formulation to compute solutions to the 3-D Navier-
Stokes equations for multiple solid bodies interacting with a fluid. The motion of
the bodies is calculated from interfacial forces with the fluid, and the motion of the
fluid is dependent on the location, velocity, and rotation of the solids. Glowinsky
and co-workers simulate sedimentation of over 1000 circular particles in two dimen-
sions and document the kiss-and-tumble phenomena of two 3-D spheres falling in
incompressible, viscous flow [45].
CHAPTER 1. INTRODUCTION

With the historical and academic context of this work established, we quickly summarize our objectives and accomplishments before embarking on the derivation of our underlying numerical method and its coupling with an immersed boundary condition.

1.5 Objectives and Accomplishments

The objective of this work is to design, implement, and employ an immersed boundary condition to simulate fluid flow over deformed surfaces modeling discrete, MEMS-like actuators for flow control. This goal is accomplished by

1. Deriving two formulations of the IBC — one implicit and one explicit — for a fractional step, pseudospectral method commonly used in simulating turbulent fluid flow with Large Eddy Simulation;

2. Presenting methods for solving the state equations that have been augmented with the Immersed Boundary Condition and implementational details for using the IBC to model unsteady, irregular walls in channel flow;

3. Validating the method and exploring its parameters by using the IBC to simulate flows with known solutions, both analytical and numerical, to an array of problems with stationary, unsteady, flat, and deformed walls, including: (a) turbulent flow with an LES model in a steady channel with flat walls, (b) laminar flow in a steady channel with flat walls, (c) laminar flow in a channel under wall-normal acceleration, (d) laminar flow in a grooved channel, and (e) evolution of laminar flow containing synthetic instability waves;

4. And employing the IBC to simulate fluid motion over actuators modeled as deformed surfaces in flow control experiments for drag reduction in a turbulent
channel, including: (a) studying the effects of a single actuator on the surrounding flowfield, (b) exploring the bearing of physical parameters of an array of discrete, MEMS-type actuators on open loop control efficacy, (c) conducting closed loop control with flowfield information using continuous and discrete actuators, and (d) testing a closed loop control scheme using only wall-mounted sensors with discrete actuators.
Chapter 2

Numerical Method for Simulation of Turbulent Flow

The Immersed Boundary Condition is, in some ways, no more than a way of augmenting an existing numerical method to make it more versatile. This chapter explains the underlying numerical method that the IBC augments for this research, along with the geometry of the flow to which it is applied. The Immersed Boundary Condition is then explained in Chapter 3 using this foundation.

2.1 Geometry and Non-Dimensionalization

The numerical method used in this research is designed to simulate turbulent planar channel flows. In this context, the channel is composed of two parallel, infinite walls. The fluid between the two walls flows due to an applied pressure gradient. As shown in Figure 2.1, the direction parallel to the walls and along the pressure gradient is the $x$, $x_1$, or streamwise direction; the direction parallel to walls and normal to the flow is the $z$, $x_3$, or spanwise direction; and the direction orthogonal to the walls is the $y$, $x_2$, or wall-normal direction. Because numerical methods are generally not well suited to simulating infinite domains, an assumption of periodicity is made. Under this assumption, fluid that leaves one end of the channel re-enters on the opposite side. Likewise, fluid that sloshes through the side of the channel re-enters the other side. The assumption of periodicity is valid as long as fluid velocity and pressure either do not change in the directions of periodicity or the domain is big enough relative to the size of the largest flow structures that periodicity does not cause unphysical effects. An appropriate no-slip or no-shear boundary condition is applied at the channel walls;
Figure 2.1: Coordinate system and periodicity of the turbulent channel flow.

net mass flux through the walls is prohibited.

Unless otherwise indicated, the size of the computational domain for simulations reported here is \((4\pi\delta, 2\delta, 4\pi \delta/3)\) in the \(x_1\), \(x_2\), and \(x_3\) directions respectively, where \(\delta\) is the channel half-height — half of the distance between the channel walls. For nondimensionalization, we choose \(\delta\) as the reference length scale and \(u_\tau\) as the reference velocity, given by

\[
 u_\tau = \sqrt{\frac{\tau_w}{\rho}},
\]

where \(\tau_w\) is the time-averaged shear stress on the walls and \(\rho\) is the fluid density, assumed constant — \(u_\tau\) is called the friction velocity. Accordingly, the reference convective time scale is then \(\delta/u_\tau\) and the Reynolds number is \(Re = Re_\tau = u_\tau \delta/\nu\).

### 2.2 The State Equation and LES

The motion of a fluid is governed by the Navier-Stokes equations. For the case of an incompressible fluid, the Navier-Stokes equations are derived from fundamental
principles, including conservation of mass and momentum. For incompressible, Newtonian fluids with constant viscosity, the non-dimensionalized Navier-Stokes — or momentum — equations can be written with the Einstein summation convention in Cartesian tensor notation as

\[ u_{i,i} + (u_i u_j)_j - \frac{2}{Re} S_{ij,j} + p_i + f_i = 0 \quad (2.1) \]

and are closed by an expression of conservation of mass, the continuity equation:

\[ u_{i,i} = 0. \quad (2.2) \]

Here, \( u_i \) represents the three components of velocity, the strain-rate tensor \( S_{ij} = (u_{i,j} + u_{j,i})/2 \), \( p \) represents pressure, \( f_i \) represents the three components of a body force, and \( Re \) is the Reynolds number based on friction velocity, a non-dimensional parameter that quantifies the relative effects of viscous forces and inertial forces. Flows with small Reynolds numbers \( (Re < 100) \) are dominated by viscous effects; flows with large Reynolds numbers are dominated by inertial effects and may become turbulent. Turbulent flows are marked by chaotic evolution of coherent structures over a range of length scales. They are common in industrial applications, so simulating them is essential, but also quite difficult. Resolving the smallest structures of a turbulent flow requires very fine grids and commensurately plenteous computation time.

To reduce the computational time required to simulate turbulent flows, we employ Large Eddy Simulation (LES). LES expends computing power only on resolving the largest, most energetic structures in the flow. In order to preserve the statistical and physical accuracy of a simulation, an additional term is added to the Navier-Stokes equations as shown below that incorporates the effects of the unresolved scales on the flow. This additional term is called the sub-grid scale model and tends to act like an
CHAPTER 2. NUMERICAL METHOD FOR TURBULENT FLOW

additional component of viscosity.

Whereas Direct Numerical Simulation (DNS) requires appropriately fine computational grids to resolve all scales of a flow, LES is computed on a relatively coarse grid. The equations of motion solved in an LES can therefore be thought of as the result of applying a low-pass filter to the governing equations. The filtered Navier-Stokes equations, using an overbar to denote the filter, become

\[ \bar{u}_{i,t} + (\bar{u}_i\bar{u}_j)_{,j} - \frac{2}{Re} \bar{S}_{ij,j} + \bar{p}_i + \bar{f}_i = 0. \]  

(2.3)

The term \( \bar{u}_i\bar{u}_j \) is problematic, however; it says that \( u_i \) must be known so that a filter can be applied to \( u_iu_j \). In LES, however, we do not want to ever fully resolve the field, \textit{i.e.} calculate \( u_i \), so we approximate \( \bar{u}_i\bar{u}_j \approx \bar{u}_i\bar{u}_j - \tau_{ij} \). Here, \( \tau_{ij} \) is given the responsibility of modeling the effects of the unresolved scales on the resolved flow. It is called the Sub-Grid Scale (SGS) Reynolds stress. Using the Smagorinsky model for \( \tau_{ij} \) as described in [111], these effects can be taken into account as a viscosity that is a function of both space and time, \( \nu_T = C \bar{\Delta}^2 (2\bar{S}_{ij}\bar{S}_{ij})^{1/2} \). Details regarding the calculation of the Smagorinsky SGS model coefficient \( C \) [41] are left to the descriptions in [21] and references therein. Employing this tool, the momentum equation becomes

\[ \bar{u}_{i,t} + (\bar{u}_i\bar{u}_j)_{,j} - 2 \left[ \left( \frac{1}{Re} + \nu_T \right) \bar{S}_{ij} \right]_{,j} + \bar{p}_i + \bar{f}_i = 0. \]  

(2.4)

Combining the two viscosity terms into \( \nu \), the governing equation becomes simply

\[ \bar{u}_{i,t} + (\bar{u}_i\bar{u}_j)_{,j} - 2 \left[ \nu \bar{S}_{ij} \right]_{,j} + \bar{p}_i + \bar{f}_i = 0, \]  

(2.5)

which looks like Navier-Stokes for an incompressible, Newtonian fluid with variable viscosity.

Henceforth, we drop the overbar, leaving it as understood that all variables are
CHAPTER 2. NUMERICAL METHOD FOR TURBULENT FLOW

filtered in the context of LES. This brings us to the final form of the equations of state:

\[ u_{i,t} + (u_i u_j)_{,j} - 2[\nu S_{ij}]_{,j} + p_{,i} + f_i = 0 \]  

(2.6a)

\[ u_{i,i} = 0. \]  

(2.6b)

2.3 A Numerical Method for Channel Flow

This section describes how the state equations are discretized and then solved without the Immersed Boundary Condition, as in [2,21,114]. The equations will be discretized in time, the non-linearity treated with a Newton iteration, and the resulting system solved with a fractional step method. The discrete system is then formulated as a whole and the solution procedure outlined. Finally, additional information is provided on discretization in space and the determination of pressure to second-order accuracy. Rigorous development of this algorithm facilitates presentation of the Immersed Boundary Condition in Chapter 3.

2.3.1 Temporal Discretization

In order to maximize the efficiency of simulations, the computational grid is refined in \( y \) to best resolve boundary layers in the flow. In an explicit time discretization, this spatial refinement can cause prohibitively tight stability constraints. As a consequence, the problematic terms — wall normal convection and diffusion — are treated implicitly with Crank-Nicholson. What remains is advanced in time with a third order Runge-Kutta method. For the time being, the body force will be treated as known at every sub-step.

Collecting all of the explicit terms in \( A_i \) and all of the implicit terms in \( B_i \), the momentum equations are

\[ u_{i,t} = A_i + B_i - p_{,i} - f_i \]  

(2.7)
with

\[ A_1 = \frac{\partial}{\partial x_1} [2\nu S_{11} - u_1 u_1] + \frac{\partial}{\partial x_3} [2\nu S_{13} - u_1 u_3] - \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_2}{\partial x_1} \right] \]  

(2.8a)

\[ A_2 = \frac{\partial}{\partial x_1} [2\nu S_{12} - u_1 u_2] + \frac{\partial}{\partial x_3} [2\nu S_{23} - u_2 u_3] \]  

(2.8b)

\[ A_3 = \frac{\partial}{\partial x_1} [2\nu S_{13} - u_1 u_3] + \frac{\partial}{\partial x_3} [2\nu S_{33} - u_3 u_3] - \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_2}{\partial x_3} \right] \]  

(2.8c)

and

\[ B_1 = -\frac{\partial u_1 u_2}{\partial x_2} + \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_1}{\partial x_2} \right] \]  

(2.9a)

\[ B_2 = -\frac{\partial u_2 u_2}{\partial x_2} + 2\frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_2}{\partial x_2} \right] \]  

(2.9b)

\[ B_3 = -\frac{\partial u_2 u_3}{\partial x_2} + \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_3}{\partial x_2} \right]. \]  

(2.9c)

Employing the three-step time advancement scheme [2, 114],

\[ \frac{u_i^k - u_i^{k-1}}{\Delta t} = \beta_k \{ B_i^k + B_i^{k-1} \} + \gamma_k A_i^{k-1} + \eta_k A_i^{k-2} - 2\beta_k p_i^k - 2\beta_k f_i^k \]  

(2.10a)

\[ u_{i,i}^k = 0 \]  

(2.10b)

where the superscript \( k \) represents the Runge-Kutta sub-step index, which runs from 1 to 3. Using superscript \( n \) to represent a full time-step, \( u^{k-1} = u^{(n-1)} \) for \( k = 1 \) and \( u^k = u^{(n)} \) for \( k = 3 \). The coefficients, \( \beta_k, \gamma_k, \) and \( \eta_k \) are selected such that time advancement is third order accurate for explicit terms and second-order accurate for implicit terms [2, 114]:

\[ \beta_1 = 4/15, \quad \beta_2 = 1/15, \quad \beta_3 = 1/6, \]
\[ \gamma_1 = 8/15, \quad \gamma_2 = 5/12, \quad \gamma_3 = 3/4, \]
\[ \eta_1 = 0, \quad \eta_2 = -17/60, \quad \eta_3 = -5/12. \]
As expected, they satisfy

$$\sum_{k=1}^{3} 2\beta_k = \sum_{k=1}^{3} (\gamma_k + \eta_k) = 1.$$  

The explicit treatment of streamwise and spanwise terms leads to a stability constraint, $SC$, given by

$$SC = \Delta t \left\{ \frac{|u|}{\Delta x} + \frac{|w|}{\Delta z} + 4\nu \left( \frac{1}{\Delta x^2} + \frac{1}{\Delta z^2} \right) \right\}_{\text{max}}. \quad (2.11)$$

The fact that cross-terms are also treated explicitly and are not taken into account in the derivation of (2.11) leads to a more restrictive limit on $SC$ than the $\sqrt{3}$ typical of third order Runge-Kutta schemes. For the Reynolds numbers considered here, $\Delta t$ is typically on the order of 0.001 leading to $SC \approx 0.2$.

### 2.3.2 Treatment of the Non-Linearity

Due to the implicit treatment of $B$ in (2.10), the time-discretized momentum equation is non-linear in $u_2$. The non-linearity may be handled with a Newton iteration. If one only performs a single step of this Newton iteration, the numerical method reduces to that described in [21], which remains second-order accurate.

Dropping the substep index $k$ for simplicity and employing Newton iteration index $m$, the non-linear term can be expanded

$$\frac{\partial u_2^m u_2^m}{\partial x_2} = 2 \frac{\partial u_2^m u_2^{m-1}}{\partial x_2} - \frac{\partial u_2^{m-1} u_2^{m-1}}{\partial x_2} + O(\Delta t^2) \quad (2.12)$$

and $B_2^k$ in (2.10) becomes

$$B_2^m = -2 \frac{\partial u_2^{m-1} u_2^{m-1}}{\partial x_2} + \frac{\partial u_2^{m-1} u_2^{m-1}}{\partial x_2} + 2 \frac{\partial}{\partial x_2} [\nu \frac{\partial u_2^k}{\partial x_2}] + O(\Delta t^2). \quad (2.13)$$
2.3.3 Fractional Step Method

We use the fractional step method [2, 33, 96] to solve the temporally discretized equation (2.10). Multiplying through by $\Delta t$, adding a $-2\beta_k \Delta t p_{i-1}$ to each side, and taking advantage of operator notation, the system becomes

\begin{equation}
\mathbf{u}^k - \beta_k \Delta t \mathbf{M}(\mathbf{u}^k) + 2\beta_k \Delta t \mathbf{G}(p^k - p^{k-1}) = \mathbf{r}^{k-1} - 2\beta_k \Delta t \mathbf{f}^k \tag{2.14a}
\end{equation}

\begin{equation}
D(\mathbf{u}^k) = 0 \tag{2.14b}
\end{equation}

where $\mathbf{G}$ is the discrete gradient operator, $D$ is the discrete divergence operator and $\mathbf{M}$ is defined

\begin{align}
M_1(\mathbf{u}^k) &= \left( \frac{\partial}{\partial x_1} \left[ \nu \frac{\partial}{\partial x_2} \right] - \frac{\partial u_2^k}{\partial x_2} \right) u_1^k, \tag{2.15a}
M_2(\mathbf{u}^k) &= 2 \left( \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial}{\partial x_2} \right] - \frac{\partial u_2^{m-1}}{\partial x_2} \right) u_2^k, \tag{2.15b}
M_3(\mathbf{u}^k) &= \left( \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial}{\partial x_2} \right] - \frac{\partial u_2^k}{\partial x_2} \right) u_3^k. \tag{2.15c}
\end{align}

The right-hand-side, $\mathbf{r}$, is given by

\begin{align}
r_1^{k-1} &= u_1^{k-1} + \beta_k \Delta t \left\{ \frac{\partial}{\partial x_1} \left[ \nu \frac{\partial u_1^{k-1}}{\partial x_2} \right] - \frac{\partial u_1^{k-1} u_2^{k-1}}{\partial x_2} \right\} \\
&\quad + \gamma_k \Delta t A_1^{k-1} + \eta_k \Delta t A_1^{k-2} - 2\beta_k \Delta t \frac{\partial p^{k-1}}{\partial x_1}, \tag{2.16a}

r_2^{k-1} &= u_2^{k-1} - \beta_k \Delta t \frac{\partial u_2^{m-1} u_2^{m-1}}{\partial x_2} + \beta_k \Delta t \frac{\partial u_2^{k-1} u_2^{k-1}}{\partial x_2} + \beta_k \Delta t \left\{ 2 \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_2^{k-1}}{\partial x_2} \right] \right\} \\
&\quad + \gamma_k \Delta t A_2^{k-1} + \eta_k \Delta t A_2^{k-2} - 2\beta_k \Delta t \frac{\partial p^{k-1}}{\partial x_2}, \tag{2.16b}

r_3^{k-1} &= u_3^{k-1} + \beta_k \Delta t \left\{ \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u_3^{k-1}}{\partial x_2} \right] - \frac{\partial u_2^{k-1} u_3^{k-1}}{\partial x_2} \right\} \\
&\quad + \gamma_k \Delta t A_3^{k-1} + \eta_k \Delta t A_3^{k-2} - 2\beta_k \Delta t \frac{\partial p^{k-1}}{\partial x_3}. \tag{2.16c}
\end{align}
Using $\delta p^k = p^k - p^{k-1}$, equation (2.14) can be written in system form as

$$
\begin{bmatrix}
I - \beta_k \Delta t M & 2\beta_k \Delta t G \\
D & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^k \\
\delta p^k
\end{bmatrix}
= 
\begin{bmatrix}
r^{k-1} - 2\beta_k \Delta t f^k \\
0
\end{bmatrix}.
$$

(2.17)

Note that pressure may be viewed as a Lagrange multiplier allowing satisfaction of the continuity equation. In order to facilitate LU decomposition, we introduce the following approximation for the pressure gradient term:

$$
(I - \beta_k \Delta t M)2\beta_k \Delta t G \delta \phi^k = 2\beta_k \Delta t G \delta p^k.
$$

(2.18)

Clearly,

$$
G(\delta p^k - \delta \phi^k) = -\beta_k \Delta t MG \delta \phi^k,
$$

so the pseudopressure update $\delta \phi$ is a first-order approximation to the pressure update $\delta p$. Applying the pseudopressure approximation in the system,

$$
\begin{bmatrix}
I - \beta_k \Delta t M & (I - \beta_k \Delta t M)2\beta_k \Delta t G \\
D & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^k \\
\delta \phi^k
\end{bmatrix}
= 
\begin{bmatrix}
r^{k-1} - 2\beta_k \Delta t f^k \\
0
\end{bmatrix}.
$$

(2.19)

Note that the momentum equations retain second-order accuracy since a Taylor series expansion of the pseudopressure update gives $\delta \phi^k = O(\Delta t)$ and $\delta \phi$ is a first-order approximation to $\delta p$. The approximate system (2.19) can be factored into the block LU decomposition

$$
\begin{bmatrix}
I - \beta_k \Delta t M & 0 \\
D & -2\beta_k \Delta t DG
\end{bmatrix}
\begin{bmatrix}
I & 2\beta_k \Delta t G \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^k \\
\delta \phi^k
\end{bmatrix}
= 
\begin{bmatrix}
r^{k-1} - 2\beta_k \Delta t f^k \\
0
\end{bmatrix}.
$$

(2.20)

which is equivalent to the two-step system:

$$
\begin{bmatrix}
I - \beta_k \Delta t M & 0 \\
D & -2\beta_k \Delta t DG
\end{bmatrix}
\begin{bmatrix}
\hat{\mathbf{u}}^k \\
\delta \phi^k
\end{bmatrix}
= 
\begin{bmatrix}
r^{k-1} - 2\beta_k \Delta t f^k \\
0
\end{bmatrix}.
$$

(2.21a)
\[
\begin{bmatrix}
I & 2\beta_k \Delta t G \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^k \\
\delta\phi^k
\end{bmatrix} = 
\begin{bmatrix}
\hat{\mathbf{u}}^k \\
\delta\hat{\phi}^k
\end{bmatrix}.
\]  
(2.21b)

These two equations can be written as:

\[
\hat{\mathbf{u}}^k - \beta_k \Delta t \mathbf{M}(\hat{\mathbf{u}}^k) = \mathbf{r}^{k-1} - 2\beta_k \Delta t \mathbf{f}^k
\]  
(2.22a)

\[
DG(\delta\phi^k) = \frac{1}{2\beta_k \Delta t} D\hat{\mathbf{u}}^k
\]  
(2.22b)

\[
\mathbf{u}^k = \hat{\mathbf{u}}^k - 2\beta_k \Delta t G(\delta\phi^k).
\]  
(2.22c)

The solution procedure is as follows:

1. Solve the momentum equations (2.22a) for the intermediate velocity field, \(\hat{\mathbf{u}}^k\).

   Note from (2.22c) that \(\hat{\mathbf{u}}^k\) is a second order approximation to the solenoidal velocity field \(\mathbf{u}^k\), thus the same boundary conditions can be used for the two velocity fields without loss of accuracy [2]. From (2.15b), it is clear that the \(y\)-momentum equation can be solved first since it does not contain either \(u_3^k\) or \(u_1^k\). The \(y\)-momentum equation is solved with the Newton iteration. To start the Newton iteration, \(u_2^{k-1}\) is used as \(\hat{u}_2^{m-1}\). Note in (2.12) that using \(k = m\) and \(k - 1 = m - 1\) gives a second order approximation to the non-linear term, so one iteration is sufficient to preserve accuracy. Once converged, \(\hat{u}_2^k\) --- a second-order approximation to \(u_2^k\) --- is used to solve the \(x\)- and \(z\)-momentum equations.

2. The divergence of the intermediate velocity field yields the right-hand-side of the Poisson equation (2.22b), which is a system of tridiagonals and relatively inexpensive to solve.

3. With \(\delta\phi^k\) now known, the intermediate velocity field is projected onto a divergence free velocity field using (2.22c).
Thus, the final set of temporal discretized equations becomes:

\[
\begin{align*}
\left\{ 1 - 2\beta_k \Delta t \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial}{\partial x_2} \right] + 2\beta_k \Delta t \frac{\partial u_2^{k-1}}{\partial x_2} \right\} \dot{u}_2^k &= r_2^{k-1} - 2\beta^k \Delta t f_2^k \\
\left\{ 1 - \beta_k \Delta t \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial}{\partial x_2} \right] + \beta_k \Delta t \frac{\partial u_2^{k-1}}{\partial x_2} \right\} \dot{u}_1^k &= r_1^{k-1} - 2\beta^k \Delta t f_1^k \\
\left\{ 1 - \beta_k \Delta t \frac{\partial}{\partial x_2} \left[ \nu \frac{\partial}{\partial x_2} \right] + \beta_k \Delta t \frac{\partial u_2^{k-1}}{\partial x_2} \right\} \dot{u}_3^k &= r_3^{k-1} - 2\beta^k \Delta t f_3^k
\end{align*}
\]  
(2.23a)

\( \frac{\partial^2 \delta \phi^k}{\partial x_j \partial x_j} = \frac{1}{2\beta_k \Delta t} \frac{\partial \dot{u}_i^k}{\partial x_i} \)  
(2.23d)

\( u_i^k = \dot{u}_i^k - 2\beta_k \Delta t \frac{\partial \delta \phi^k}{\partial x_i} \)  
(2.23e)

It is interesting to note that, as pointed out by Perot [96], \( \phi \) may be used throughout a simulation in lieu of pressure without loss of accuracy to the velocity field, even though \( \phi \) is only a first order approximation to \( p \). This is not a feature we generally take advantage of when employing the Immersed Boundary Condition in the simulation due to the IBC's introduction of additional error terms. Instead, we choose to add a fourth step to the algorithm above and compute pressure to second order accuracy as described in the next section.

### 2.3.4 Computation of Pressure to Second Order Accuracy

Rather than using the pseudopressure \( \phi \) as a first order approximation to pressure, we instead improve to second order accuracy by solving a Poisson equation. Taking the divergence of the momentum equation (2.5) and simplifying with continuity gives the new equation to solve for pressure \( p \):

\[
\nabla^2 p^k = -(u_i^k u_j^k)_{,ij} + (2\nu S_{ij}^k)_{,ij} - f_{i,i}^k.
\]  
(2.24)
Equation (2.24) is subject to a Neumann boundary condition found from the y-momentum equation at the wall:
\[
\frac{\partial p^k}{\partial x_2} \Bigg|_{wall} = \left[ -u_{2,t}^k + (2\nu S^k_{2j}) - f^k_2 \right]_{wall}.
\] (2.25)

Fully discretized, (2.24) and (2.25) are a set of tridiagonal systems that can be solved quite efficiently.

### 2.3.5 Spatial Discretization and Grid Generation

The LES momentum and continuity equations are solved on a rectangular, staggered grid [2,13,55] as shown in Figure 2.2. The grid is uniform in the streamwise and spanwise directions and stretched in the wall-normal direction with a hyperbolic tangent function to better resolve boundary layers in the flow:
\[
y = \frac{\tanh(\xi \tilde{y})}{\tanh(\xi)},
\] (2.26)

with $\xi$ the stretching factor and $\tilde{y}$ the unstretched grid sites. Generally, $\xi = 1.75$.

A Fourier spectral method [17,93–95] is used to discretize in planes parallel to the walls, a natural choice that simultaneously enforces the periodicity assumptions and delivers exponential accuracy. As in [21], $3/2$ dealiasing is used with the discrete Fourier transform in the $x$- and $z$-directions; the Fourier-to-real transformation is
\[
u(x_m, y, z_n) = \sum_{k_z=-N_z/2+1}^{N_z/2} \sum_{k_z'=-N_z/2+1}^{N_z/2} \hat{u}(k_z, y, k_z') e^{2\pi i(k_z x_m/L_x + k_z z_n/L_z)},
\] (2.27)

and the real-to-Fourier transformation is
\[
\hat{u}(k_z, y, k_z) = \frac{1}{N_z N_z} \sum_{m=0}^{N_x-1} \sum_{n=0}^{N_z-1} u(x_m, y, z_n) e^{-2\pi i(k_z x_m/L_x + k_z z_n/L_z)}.
\] (2.28)

In equations (2.27) and (2.28), $i = \sqrt{-1}$, and $N_x$ and $N_z$ are the number of collocation
Figure 2.2: Grid with stretching in wall-normal direction. Here, $\xi = 1.20$ for illustrative purposes. Commonly, $\xi = 1.75$.

Points in the $x$- and $z$-directions, given by

$$
\begin{align*}
    x_m &= mL_x/N_x, \quad m = 0, \cdots, N_x - 1, \\
    z_n &= nL_z/N_z, \quad n = 0, \cdots, N_z - 1,
\end{align*}
$$

(2.29)

where $L_x$ and $L_z$ are the domain size in the $x$- and $z$-directions. For additional details, refer to [17].

Second-order finite differences are used in the wall-normal direction on the stretched grid, a departure from the fully spectral code of Kim et al. [67] and the commonly used and more accurate Chebyshev approach. The second-order finite differencing originally facilitated implementation of wall-normal velocity boundary conditions and is used here in the Immersed Boundary Method. As discussed previously, the implicit Crank-Nicholson method is employed for time advancement of wall-normal terms in order to mitigate severe CFL time step restrictions.

The computational grid in the wall normal direction is staggered as shown in Figure 2.3 to prevent over-determination of pressure. The $z_2$-velocities live at the
Figure 2.3: A cartoon depicting the staggered, stretched grid in the wall-normal direction. Pressure, $p$, and velocities $u$ and $w$ live on crosses. Velocity $v$ lives on the circles.

circles, indexed $j = 0, 1, 2, \ldots, N_y$. Pressure, $x_1$-, and $x_3$-velocities live at the crosses, half-way between pairs of circles. We index the crosses $j - \frac{1}{2}$ for $j = 1, 2, \ldots, N_y$.

Thus, the first-order $y$-derivatives are

\[
\frac{\partial u}{\partial x_2} \bigg|_j = \frac{u_{j+\frac{1}{2}} - u_{j-\frac{1}{2}}}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}} \quad (2.30a)
\]

\[
\frac{\partial v}{\partial x_2} \bigg|_{j-\frac{1}{2}} = \frac{v_j - u_{j-1}}{y_j - y_{j-1}} \quad (2.30b)
\]

and the second-order $y$-derivatives are calculated

\[
\frac{\partial}{\partial x_2} \left[ \nu \frac{\partial u}{\partial x_2} \right]_{j-\frac{1}{2}} = \frac{1}{y_j - y_{j-1}} \left[ \nu_j \frac{u_{j+\frac{1}{2}} - u_{j-\frac{1}{2}}}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}} - \nu_{(j-1)} \frac{u_{j-\frac{1}{2}} - u_{j-\frac{3}{2}}}{y_{j-\frac{1}{2}} - y_{j-\frac{3}{2}}} \right] \quad (2.31a)
\]

\[
\frac{\partial}{\partial x_2} \left[ \nu \frac{\partial v}{\partial x_2} \right]_j = \frac{1}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}} \left[ \nu_{j+\frac{1}{2}} \frac{v_{j+1} - v_j}{y_{j+1} - y_j} - \nu_{j-\frac{1}{2}} \frac{v_j - v_{j-1}}{y_j - y_{j-1}} \right] \quad (2.31b)
\]
Chapter 3

Immersed Boundary Condition

Numerical simulations of flows over irregular surfaces have typically relied on boundary-fitted grids so that standard treatments may be used to discretely satisfy analytical boundary conditions, for example in the work of Tezduyar and colleagues \[9,10,64,65\], Lumley's group \[19\], and Choi et al. \[66\], just to cite a few. For moving, irregular interfaces, the expense of modifying the grid each time the boundary moves is often as expensive as solving the state equations themselves. The Immersed Boundary Condition, however, eliminates this expense by cleverly augmenting the state equations to simulate flows over irregular boundaries on an unchanging, structured rectilinear grid.

Having rigorously derived a numerical method for solving the state equations with standard boundary treatments and introduced the Immersed Boundary Condition in a heuristic way in Section 1.3, it is now possible to develop the IBC mathematically into both explicit and implicit algorithms.

3.1 Explicitly Constrained State Equation

The Immersed Boundary Condition amounts to constraining the flow such that interpolation across the virtual interface is consistent with a no-slip, no-penetration boundary treatment. We enforce this constraint with a body force defined on only some portions of the domain, generally those occupied by the virtual body. Expressing each constraint as a linear combination of the discrete velocities in the vector \( \mathbf{u} \), we can write the entire set of constraints as the matrix equation \( \mathbf{H} \mathbf{u} = \mathbf{u}_* \), where \( \mathbf{H} \)
is the constraint operator and the vector $u_s$ allows for the possibility that the constraints are not homogeneous. For example, suppose that the vector $u$ of velocities on an $(N_x, N_y, N_z)$ grid is arranged as in Section 2.3.3

$$
\mathbf{u} = \begin{bmatrix}
    u \\
    v \\
    w
\end{bmatrix} = \begin{bmatrix}
    u(1,1,1) \\
    u(1,2,1) \\
    u(1,3,1) \\
    \vdots \\
    u(1,N_y,1) \\
    u(2,1,1) \\
    \vdots \\
    u(N_x,N_y,1) \\
    u(1,1,2) \\
    \vdots \\
    u(N_x,N_y,N_z) \\
    u(1,1,1) \\
    \vdots 
\end{bmatrix} \quad (3.1)
$$

and the only constraint we wish to impose is that the average of $u_{(1,1,1)}$ and $u_{(1,2,1)}$ is one. By setting

$$
\mathbf{H} = \begin{bmatrix}
    0.5 & 0.5 & 0 & \cdots
\end{bmatrix} \quad \text{and} \quad \mathbf{u}_s = [1] \quad (3.2)
$$

the constraint equation $\mathbf{H} \mathbf{u} = \mathbf{u}_s$ gives $0.5u_{(1,1,1)} + 0.5u_{(1,2,1)} = 1$ when written out.

To implement the IBC in the pseudo-spectral, fractional step code described in Chapter 2, the first step is to augment the system formulation (2.17) with a set of linear constraints on the velocity field $u^k$:

$$
\begin{bmatrix}
    \mathbf{I} - \beta_k \Delta t \mathbf{M} & 2\beta_k \Delta t \mathbf{G} \\
    \mathbf{D} & 0
\end{bmatrix}
\begin{bmatrix}
    \mathbf{u}^k \\
    \mathbf{\delta p}^k
\end{bmatrix} =
\begin{bmatrix}
    r^{k-1} - 2\beta_k \Delta t f^k \\
    0 \\
    \mathbf{u}_s
\end{bmatrix}. \quad (3.3)
$$

Whereas the left-hand-side matrix in (2.17) is square, adding constraints to this
system as we have done in (3.3) clearly overdetermines the system — there are more equations than unknowns. By adding a set of Lagrange multipliers to the system called $f_s$, one for each constraint, we can make the new left-hand-side matrix square, although not necessarily invertible:

$$\begin{bmatrix}
I - \beta_k \Delta t M & 2\beta_k \Delta t G & J \\
D & 0 & 0 \\
H & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
\delta p^k \\
\delta p^k \\
\end{bmatrix}
= \begin{bmatrix}
r^{k-1} \\
f_s \\
0 \\
u_s \\
\end{bmatrix}. \tag{3.4}
$$

These Lagrange multipliers appear only in the momentum equation, where they are the source term $Jf_s$. This source term allows us the flexibility to not satisfy some of the momentum equations, thus keeping the number of unknowns and equations the same. Note that we have dropped the $-2\beta_k \Delta t f^k$ on the right-hand-side. The body force was intended to satisfy the immersed boundary condition by allowing the velocity field to be set independently of the momentum equations. This is exactly the role of the new $Jf_s$ term, so we have set

$$f = \frac{1}{2\beta_k \Delta t} Jf_s. \tag{3.5}$$

Note that naturally-occurring body forces such as a mean pressure gradient, gravity, or electrohydrodynamic forces must be taken into account with the Runge-Kutta method and included in $r^{k-1}$ via (2.8).

To illustrate how the body force term $Jf_s$ works, recall the example constraint in (3.2) on the velocity field (3.1). Let us say that we want to apply a body force only at $(1, 1, 1)$. In this case, we would define

$$J = \begin{bmatrix}
1 \\
0 \\
0 \\
\vdots \\
\end{bmatrix}. \tag{3.6}$$
so that only the momentum equation at (1, 1, 1) has a non-zero momentum source term. The role of matrix \( J \) is to determine where forces are applied in the flow. Where \( J \) has zeroes, no body force is applied. Matrix \( J \) also gives us the freedom to set force distributions; each column of \( J \) gives one force distribution. For example, to apply force in a linear ramp over the first four points in \( u \), the choice of \( J \) would be

\[
J = \begin{bmatrix}
1 \\
2 \\
3 \\
4 \\
0 \\
\vdots
\end{bmatrix}.
\] (3.7)

The magnitude of the entries in \( J \) does not matter since \( J \) is scaled by \( f_s \) in the momentum equations; \( f_s \) is selected to satisfy the constraints. Note that regardless of the force distributions and the number of locations affected by each force distribution in a column of \( J \), there must be only one column of \( J \) for each Lagrange multiplier and one Lagrange multiplier for each constraint, or row of \( H \). To reiterate, a necessary but insufficient condition for a determinate state matrix is that for every row in \( H \) there is a corresponding column in \( J \). Furthermore, each row in \( H \) must be linearly independent — likewise with each column of \( J \). Intuition also suggests that contradictory constraints are problematic. For example, the flow is required by the second line of the state matrix to be divergence-free. Applying a set of constraints that result in a mass source would lead to an intractable problem.

To solve the new state system, (3.4) can be approximately split into two systems that closely resemble the ones solved in Section 2.3.3:

\[
\begin{bmatrix}
I - \beta_k \Delta t M & J \\
H & 0
\end{bmatrix}
\begin{bmatrix}
u^{m+1/2} \\
f_s^{m+1/2}
\end{bmatrix}
= \begin{bmatrix}
r^{k-1} - 2\beta_k \Delta t G \delta p^{m-1} \\
u_s
\end{bmatrix}
\] (3.8a)
\[
\begin{bmatrix}
I - \beta_k \Delta t M & 2\beta_k \Delta t G \\
D & 0
\end{bmatrix}
\begin{bmatrix}
u^m \\
\delta^m
\end{bmatrix}
= \begin{bmatrix}
r^{k-1} - Jf_s^{m+1/2} \\
0
\end{bmatrix}.
\] (3.8b)

The decoupling of (3.4) has been accomplished by replacing the substep index \(k\) with iteration index \(m\), lagging the pressure and force terms in (3.8a) and (3.8b), and moving them to the right hand side. The two systems can then be solved independently, iterating between the two until convergence. It is beyond the scope of this work to show rigorously that this approximate splitting converges to a solution of the whole system, or converges at all. However, as shown in Chapter 4 through numerical experiments, the algorithm is successful and converges at second order. Although the systems (3.8a) and (3.8b) could be solved in any order, we choose to solve (3.8b) last in order to guarantee a divergence-free solution. Divergence-free solutions are required to preserve the accuracy and stability of the method. To begin, the pressure update term is taken as zero and (3.8a) is solved. The resulting velocity and force fields are used to update \(M\) and the Newton iteration terms in \(r^{k-1}\); equation (3.8b) is then solved for a new velocity and pressure field. The operator \(M\) and right hand side \(r^{k-1} - 2\beta k \Delta t G\delta^m\) are updated in (3.8a) and a new solution is obtained, and so on.

The systems (3.8a) and (3.8b) are individually solved just as in Section 2.3.3, with an approximate factorization. Similar to the approximation of pressure with pseudopressure (2.18), force is approximated

\[
Jf_s \approx (I - \beta_k \Delta t M)Jg_s
\]

(3.9)

so that

\[
Jf_s - Jg_s = -\beta_k \Delta t MJg_s,
\]

(3.10)

and \(g_s\) is a temporally first-order approximation to \(f_s\). With this approximation,
system (3.8a) becomes

\[
\begin{bmatrix}
I - \beta_k \Delta t M & (I - \beta_k \Delta t M)J \\
H & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^{m+1/2} \\
\mathbf{g}_s^{m+1/2}
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{r}^{k-1} - 2\beta_k \Delta t \mathbf{G}\delta \phi^{m-1} \\
\mathbf{u}_s
\end{bmatrix}.
\] (3.11)

Equation (3.11) is factored

\[
\begin{bmatrix}
I - \beta_k \Delta t M & 0 \\
H & -HJ
\end{bmatrix}
\begin{bmatrix}
I & J \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^m \\
\mathbf{g}_s^m
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{r}^{k-1} - 2\beta_k \Delta t \mathbf{G}\delta \phi^{m-1} \\
\mathbf{u}_s
\end{bmatrix},
\] (3.12)

which is equivalent to the two-step system

\[
\begin{bmatrix}
I - \beta_k \Delta t M & 0 \\
H & -HJ
\end{bmatrix}
\begin{bmatrix}
\tilde{\mathbf{u}}^m \\
\tilde{\mathbf{g}}_s^m
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{r}^{k-1} - 2\beta_k \Delta t \mathbf{G}\delta \phi^{m-1} \\
\mathbf{u}_s
\end{bmatrix}.
\] (3.13)

\[
\begin{bmatrix}
I & J \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^m \\
\mathbf{g}_s^m
\end{bmatrix}
= 
\begin{bmatrix}
\tilde{\mathbf{u}}^m \\
\tilde{\mathbf{g}}_s^m
\end{bmatrix}.
\] (3.14)

Finally, noting that \( \mathbf{g}_s^m = \tilde{\mathbf{g}}_s^m \) and dropping the tilde, the system is solved with the following algorithm:

**A1.** Solve for \( \tilde{\mathbf{u}}^m \) in the tridiagonal system \((I - \beta_k \Delta t M)\tilde{\mathbf{u}}^m = \mathbf{r}^{k-1} - 2\beta_k \Delta t \mathbf{G}\delta \phi^{m-1}\).

**A2.** Find the approximate Lagrange multiplier \( \mathbf{g}_s \) from \( HJ\mathbf{g}_s^m = -\mathbf{u}_s + H\mathbf{\tilde{u}}^m \).

**A3.** Update the velocity field to satisfy the constraints: \( \mathbf{u}^m = \mathbf{\tilde{u}}^m - J\mathbf{g}_s^m \).

To solve the other half of the system, equation (3.8b), pressure is approximated with pseudopressure (2.18) leading to

\[
\begin{bmatrix}
I - \beta_k \Delta t M & (I - \beta_k \Delta t M)2\beta_k \Delta t \mathbf{G} \\
-2\beta_k \Delta t D & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^m \\
\delta \phi^m
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{r}^{k-1} - J\mathbf{g}_s^{m+1/2} \\
0
\end{bmatrix}.
\] (3.15)

Equation (3.15) can be factored

\[
\begin{bmatrix}
I - \beta_k \Delta t M & 0 \\
-2\beta_k \Delta t D & 0
\end{bmatrix}
\begin{bmatrix}
I & 2\beta_k \Delta t \mathbf{G} \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^m \\
\delta \phi^m
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{r}^{k-1} - J\mathbf{g}_s^{m+1/2} \\
0
\end{bmatrix},
\] (3.16)
which is equivalent to the two-step system

\[
\begin{bmatrix}
1 - \beta_k \Delta t M & 0 \\
D & -2 \beta_k \Delta t D G
\end{bmatrix}
\begin{bmatrix}
\delta \phi^m \\
\delta \phi^m
\end{bmatrix}
= \begin{bmatrix}
\mathbf{r}^{k-1} - \mathbf{J} \mathbf{g}_s^{m+1/2} \\
0
\end{bmatrix}
\] (3.17)

\[
\begin{bmatrix}
1 & 2 \beta_k \Delta t G \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}^n \\
\delta \phi^m
\end{bmatrix}
= \begin{bmatrix}
\hat{\mathbf{u}}^m \\
\hat{\delta \phi}^m
\end{bmatrix}.
\] (3.18)

Noting that \(\delta \phi^k = \hat{\delta \phi}^k\) and dropping the hats, the system is solved with the following algorithm:

**B1.** Solve for the approximate velocity field \(\hat{\mathbf{u}}^m\) with \(\hat{\mathbf{u}}^m - \beta_k \Delta t M(\hat{\mathbf{u}}^m) = \mathbf{r}^{k-1} - \mathbf{J} \mathbf{g}_s^m\).

**B2.** Calculate the pseudopressure update from \(D G(\delta \phi^m) = \frac{1}{2 \beta_k \Delta t} D \hat{\mathbf{u}}^n\).

**B3.** Project the velocity field to a divergence free space: \(\mathbf{u}^m = \hat{\mathbf{u}}^m - 2 \beta_k \Delta t G(\delta \phi^m)\).

In summary, the overall algorithm for simulation of flows with the fractional step method augmented with an explicit Immersed Boundary Condition is as follows:

**Step 1.** Assemble \(\mathbf{M}\) and \(\mathbf{r}^{k-1}\) as described in Chapter 2.

**Step 2.** Complete steps **A1** through **A3**, solving for \(\hat{\mathbf{u}}^m_2\) first.

**Step 3.** Use the information from **Step 2** to reassemble \(\mathbf{M}\) and \(\mathbf{r}^{k-1}\).

**Step 4.** Complete steps **B1** through **B3**, solving for \(\hat{\mathbf{u}}^m_2\) first.

**Step 5.** If not converged, return to **Step 1**. Otherwise, \(m = k\).

**Step 6.** Update the pressure with either \(\phi^k = \phi^{k-1} + \delta \phi^k\) or by calculating the applied body force from (3.5), \(2 \beta_k \Delta t f = \mathbf{J} f_s\), and then solving the pressure Poisson equation, (2.24).

**Step 7.** Increment \(k\) and advance another substep, beginning with **Step 1**.
3.2 Implicitly Constrained State Equation

Depending on the selection of $J$ in (3.4), it is possible to avoid the complicated solution procedure of Section 3.1. For example, if we choose to apply a body force at only one location for each constraint — that is, if each column of $J$ contains only one non-zero value — the constrained system can be solved implicitly with no more effort than solving the unconstrained system. Say, for example, that a stationary virtual interface passes halfway between grid sites $(10,5,10)$ and $(10,6,10)$. The interpolation of $u$ halfway between these grid locations should be zero, so we add a constraint equation, making $Hu = u$, read

$$
\begin{bmatrix}
\vdots \\
u_{(10,4,10)} \\
u_{(10,5,10)} \\
u_{(10,6,10)} \\
u_{(10,7,10)} \\
\vdots 
\end{bmatrix}
\begin{bmatrix}
\vdots \\
0.5 \\
0.5 \\
0 \\
\vdots 
\end{bmatrix}
= \begin{bmatrix}
\vdots \\
0 
\end{bmatrix}.
\tag{3.19}
$$

If $(10,5,10)$ is inside the virtual body, we may choose to satisfy the constraint by using a body force at this location only, thus adding a column of zeroes to $J$ with a 1 in the row for the $u$-momentum equation at $(10,5,10)$. This is equivalent to applying whatever body force is required at this location to satisfy this constraint. In other words, solving the momentum equation is not necessary at this point; solving the constraint equation is sufficient. By this logic, we can replace the row in the state matrix that represents momentum equation with the row that describes the constraint. The constraint line in $H$ is then redundant, and the corresponding Lagrange multiplier is not needed, either. If a body force is applied only at a single location for every constraint on the flow, it is possible to remove all of the Lagrange multipliers by replacing the momentum equations with body forces in them with constraint equations.
In order to avoid trouble with the approximate factorization, however, we choose to factor the discrete state equation (2.21) first, then constrain the hat velocities. Recall the unconstrained, factored two-step system (2.21) with a general body force:

\[
\begin{bmatrix}
I - \beta_k \Delta t M & 0 \\
D & -2\beta_k \Delta t DG
\end{bmatrix}
\begin{bmatrix}
\hat{u}^k \\
\delta \hat{\phi}^k
\end{bmatrix}
= \begin{bmatrix}
\bar{r}^{k-1} - 2\beta_k \Delta t f^k \\
0
\end{bmatrix} \tag{3.20a}
\]

\[
\begin{bmatrix}
I & 2\beta_k \Delta t G \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\hat{u}^k \\
\delta \phi^k
\end{bmatrix}
= \begin{bmatrix}
\hat{u}^{k-1} \\
\delta \phi^{k-1}
\end{bmatrix}. \tag{3.20b}
\]

Replacing the momentum equations in (3.20a) that contain non-zero body forces with constraint equations, \( I - \beta_k \Delta t M \) becomes \( I - \beta_k \Delta t \bar{M} \) and \( r^{k-1} - 2\beta_k \Delta t f^k \) becomes \( \bar{r}^{k-1} \), where

\[
\bar{M} = \begin{cases} 
M & \text{for momentum equations with no body forces} \\
\frac{H-1}{\beta \Delta t} & \text{for rows replaced with constraint equations}
\end{cases} \tag{3.21}
\]

and

\[
\bar{r}^{k-1} = \begin{cases} 
r^{k-1} & \text{for momentum equations with no body forces} \\
\frac{\hat{u}^k}{\beta_k \Delta t} & \text{for rows replaced with constraint equations}
\end{cases} \tag{3.22}
\]

Note that one of the key features of our method is that matrix \( I - \beta_k \Delta t M \) is tridiagonal, minimizing the storage requirements and computational expense of solving this system; with careful selection of \( H \), this structure can be maintained in \( I - \beta_k \Delta t \bar{M} \).

The implicitly constrained version of (3.20) is simply

\[
\begin{bmatrix}
I - \beta_k \Delta t \bar{M} & 0 \\
D & -2\beta_k \Delta t DG
\end{bmatrix}
\begin{bmatrix}
\hat{u}^k \\
\delta \hat{\phi}^k
\end{bmatrix}
= \begin{bmatrix}
\bar{r}^{k-1} \\
0
\end{bmatrix} \tag{3.23a}
\]

\[
\begin{bmatrix}
I & 2\beta_k \Delta t G \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\hat{u}^k \\
\delta \phi^k
\end{bmatrix}
= \begin{bmatrix}
\hat{u}^{k-1} \\
\delta \phi^{k-1}
\end{bmatrix}. \tag{3.23b}
\]
Once (3.23a) has been solved, the body force that gives the resulting velocity distribution can be calculated by simply rearranging the first line of (3.20a):

\[ f^k = \frac{1}{2\beta_k \Delta t} \left( r^{k-1} - (I - \beta_k \Delta t M) \hat{u}^k \right). \]  

Equations (3.23) and (3.24) yield the algorithm for solving the state equations subject to an implicit Immersed Boundary Condition. Replacing the substep index \( k \) on \( \delta \phi \), \( u \) and the pressure term in \( r \) with an iteration index \( m \) allows the system to be iterated on the non-linearity and the approximate factorization. Noting that \( \hat{\delta \phi} = \delta \phi \), we drop the hat:

**C1.** Solve for the constrained velocity field with \((I - \beta_k \Delta t \tilde{M}) \hat{u}^m = \tilde{r}^{k-1}\).

**C2.** Calculate the pseudopressure update, \( \delta \phi^m \) from \( 2\beta_k \Delta t DG \delta \phi^m = D \hat{u}^m \).

**C3.** Project the field to a divergence-free space: \( u^m = \hat{u}^m - 2\beta_k \Delta t G \delta \phi^m \).

**C4.** If desired, update \( M \) and the Newton iteration and pressure terms in \( r^{k-1} \) and return to **C1**.

**C5.** Update the pseudopressure with \( \phi^k = \phi^{k-1} + \delta \phi^k \). Alternatively, use (3.24) to find the body force and solve the Poisson equation (2.24) to obtain pressure to second-order accuracy.

We know that \( \hat{u}^k \) must satisfy the constraints. It is clear from **C3**, though, that \( u^k \) does not exactly satisfy the constraints and that the error term is \(-2\beta_k \Delta t G \delta \phi^k\). Clearly, if \(-2\beta_k \Delta t G \delta \phi^k = 0\) on the portion of the computational domain where a body force is applied, the constraints will be satisfied exactly. This can be done when the desired velocity field includes points on the edge of the computational domain. The boundary treatment of **C2** specifies that \( G \delta \phi^k \big|_{\text{boundary}} = 0 \); selecting the desired velocity field to be divergence free causes the source term of **C2**, \( D \hat{u} \), to also
be zero. With this boundary treatment and source term, $G \delta \phi^k = 0$ everywhere on the virtual solid and the IBC is satisfied exactly.

### 3.3 Implementational Details

With two general frameworks — one implicit, one explicit — of the Immersed Boundary Condition, a few extra details must be discussed regarding their implementation to obtain the results presented in Chapters 4 and 5. The implicit and explicit algorithms discussed in the previous sections allow some flexibility regarding how forces are applied; this section is intended to give the details of the forcing schemes employed in the remainder of this work. As the intention is ultimately to employ the Immersed Boundary Condition to model flow control actuators in wall bounded flows, all of the discussion here is in the context of establishing two immersed boundaries in the flow with mean locations that are parallel to the existing walls but some distance inside of the computational domain as shown in Figure 3.1, thus establishing a virtual channel.

The reference length for non-dimensionalization remains $\delta$ for IBC cases, but delta is defined as half the distance from the mean location of the lower virtual wall to the mean location of the upper virtual wall. The grid stretching is changed for IBC simulations such that the minimum grid spacing occurs at the mean location of the virtual wall:

$$y = \begin{cases} 
\frac{\tanh(\xi(\tilde{y}+2\delta))}{\tanh(\xi)}, & \tilde{y} < \delta \\
\frac{\tanh(\xi\tilde{y})}{\tanh(\xi)}, & -\delta < \tilde{y} < \delta \\
\frac{\tanh(\xi(\tilde{y}-2\delta))}{\tanh(\xi)}, & \tilde{y} > \delta 
\end{cases} \quad (3.25)$$

where $\xi$ is the stretching factor and $\tilde{y}$ the unstretched grid sites. The mean pressure gradient, if required, is turned on only in the virtual channel.
3.3.1 Details of the Explicit IBC

Because the Immersed Boundary Condition plays the role of a less expensive alternative to deforming-grid methods, the computationally intensive explicit formulation of the IBC is not applied to a large number of problems in Chapter 4 other than to show the effect of choosing different force profiles on a laminar flow; therefore forcing is required only in the streamwise direction. The flow is constrained such that linear interpolation in the $y$ direction of the $u$ velocity between the first points inside and outside the virtual solid give zero velocity at the virtual interface. As such, for every line of grid sites with the same $x$ and $z$ coordinate, there are two constraint equations in $H$ — one for the upper virtual boundary and one for the lower virtual boundary. The rows of $H$ that describe these constraints are linearly independent. Note that in order to solve for the approximate Lagrange multiplier $g_s$, the matrix $HJ$ must be invertible. This is not surprising: it means that there must be a unique force profile for each constraint. One Gaussian force cannot be made to force the flow to have a particular value at two locations. Furthermore, force profiles must be non-zero at no fewer than one of the points in the constraint equation it will satisfy. Otherwise, the force has no authority to act at the site of the constraint.
Two force profiles are tested in the following chapter: a Gaussian distribution and a sharp cosine distribution, both shown in Figure 3.1. There is one such distribution for each constraint. The force profiles are a function of \( y \) and non-zero only for the \( x \) and \( z \) location where the corresponding constraint lies. The distribution of force associated with a constraint about a virtual wall at \( y_w \) is given for the Gaussian as function of \( y \):

\[
f_g = e^{-\frac{(y-y_w)^2}{\sigma^2}}
\]

with \( \sigma = (y_w - c)/2.15 \), where \( c \) is the location where the forcing is reduced to 1\% of its maximum value. We choose \( c \) to be the \( y \) value at a grid point 5 cells away from the virtual wall. The cosine distribution function is given by

\[
f_c = \begin{cases} 
1 + \cos \left( \pi \frac{y-Y_c}{y_w-Y_c} \right), & \text{on the virtual solid} \\
0, & \text{in the virtual channel}
\end{cases}
\]

where \( y_c \) is the location of the edge of the computational domain nearest the virtual wall at \( y_w \).

### 3.3.2 Details of the Implicit IBC

Three different implicit IBC schemes — shown in Figure 3.2 — are used to obtain results in Chapters 4 and 5. In all of these schemes, the \( u, v \) and \( w \) velocities are constrained in lines of constant \( x \) and \( z \) so that only linear interpolation is required.

**Implicit Scheme 1** In the first scheme, body forces are applied only on the virtual solid. The \( u \) and \( w \) components of velocity are constrained such that interpolation of the velocity field to the virtual boundary is zero. This linear profile is extrapolated through all \( x-z \) planes that are intersected by the virtual interface. For \( x-z \) planes in the virtual solid from this location to the edge of the computational domain, the off-diagonal terms in the constraint equation that
Figure 3.2: Implicit IBC schemes.
would continue this linear extrapolation are multiplied by a linear function that ranges from one to zero at the edge of the computational domain. This tapers the velocity profile to satisfy no-slip on the edge of the computational domain. At the first point inside the virtual interface, the \( v \) component of velocity is constrained to satisfy the no-penetration boundary condition. The \( v \) component is also required on the entire virtual solid to be the divergence-free complement to the \( u \) and \( w \) profiles. This is accomplished by performing two full iterations of algorithm C. In the first iteration, the \( v \) component of velocity on the virtual solid is lagged by one substep. The \( u \) and \( w \) components are constrained as indicated. Based on the velocity profile from the first iteration, \( v \) is then calculated and another iteration is performed. The wall-normal transpiration boundary treatment at the edge of the computational domain is updated to be consistent with the forced \( v \) profile.

**Implicit Scheme 2** The second scheme is identical to the first, except that body forces are also applied at the first \( u \) and \( w \) grid sites outside of the virtual solid. The velocity field at this location is constrained to give a linear profile between the zero velocity at the virtual interface and the second point outside virtual solid.

**Implicit Scheme 3** The third scheme applies a body force on the \( u \) and \( w \) components of velocity only at the first point inside and the first point outside the virtual solid. A linear velocity profile is forced between the first and second points outside the virtual solid, the virtual interface, and the first point inside the virtual solid. Forcing is applied on the \( v \) component of velocity everywhere inside the virtual solid just as in the first scheme, so the same iteration is required.
Applying body forces to \( u \) and \( w \) outside of the virtual solid is sometimes necessary to generate smooth velocity profiles. Smooth velocity profiles are required to maintain the stability and accuracy of the numerical method. Whereas applying body forces on the virtual solid is consistent with the concept that if the boundary condition is satisfied, the flow field outside the virtual solid is accurate, applying a body force inside the flow is not. However, linearizing the velocity profile over a small section of the domain only has a detrimental effect on accuracy equivalent to increasing the grid spacing. The effect is no worse than doubling the grid spacing for the first cell outside of the virtual solid. Furthermore, the grid is generally well refined in the portion of the domain where the virtual boundary may move during a simulation. Applying a body force inside the flow is critical when the virtual interface is moving or highly deformed. When points that were previously inside the virtual solid are suddenly inside the flow as the virtual interface passes them, the velocity tends to hold its value from inside the virtual solid for some time before flow phenomena smooth it out. Instead, linearizing the velocity profile at this location causes the fluid to assume a velocity consistent with the surrounding flow immediately. Likewise, when the virtual surface is highly deformed, adjacent points in spectral directions are on opposite sides of the virtual interface, frequently causing high gradients in the flow and causing spurious oscillations, or Gibbs phenomena. Smoothing of the velocity profile immediately above the virtual solid alleviates this problem, as well.

### 3.3.3 Use of the IBC

Having established all of the details of applying the Immersed Boundary Condition to the pseudospectral, fractional step algorithm, it remains to validate the method and apply it to problems of significance. The next two chapters are devoted to extensive validation of the IBC and its applications in just a minute fraction of the problems
where it shines as a less expensive alternative to methods with deforming grids.
Chapter 4

Validation Studies

The purpose of this chapter is to apply the Immersed Boundary Condition to several problems with known solutions in order to establish the accuracy of the method. By comparing solutions generated with IBC to analytical results and prior computations that have been established as correct, not only do we gain confidence that results for new problems will be valid, but we also have a chance to learn more about the method and explore its characteristics — how much grid refinement is required to adequately resolve wall features, the convergence rate in space and time, whether the method is compatible with the LES model, and which forcing schemes work best for which types of problems.

4.1 Laminar Flow in a Channel

The staggered grid and discretization discussed in Chapter 2 are carefully designed so that the numerical method conserves mass and momentum to machine precision. It is critical to both the accuracy and stability of simulations that the IBC not defeat this feature of the numerical method. If the IBC can maintain a laminar profile, it is a good indication that the numerics remain conservative. This problem also provides a good opportunity to explore the explicit and implicit formulations of the IBC and experiment with different forcing schemes.

For laminar problems, we non-dimensionalize using the streamwise velocity at the center of the channel, $u_{\text{max}}$, as the reference velocity. For problems with an Immersed Boundary Condition, the reference length scale is the virtual channel half height, $\delta$. 
The reference time is then just $\delta/u_{\text{max}}$, and the Reynolds number is

$$Re = \frac{\delta u_{\text{max}}}{\nu}. \quad (4.1)$$

In this section, we compare the performance of the Immersed Boundary Condition to a simulation with conventional boundary treatments and the analytical solution for laminar flow in channel:

$$u(y) = \frac{p_x Re}{2} (y^2 - 1), \quad v = 0, w = 0 \quad (4.2)$$

We apply a pressure gradient in the $x$-direction such that $u_{\text{max}} = 1.$ or $p_x = -2/Re$. For this experiment, we use $Re = 100$. Resolution is $(4,116,4)$ in the streamwise, wall-normal, and spanwise directions, respectively. Note that velocity is constant in the streamwise and spanwise directions, so four Fourier modes are more than sufficient in $x$ and $z$. The virtual interface is located at the tenth $v$ node from the top and bottom of the computational domain. As always, we use the $3/2$ dealiasing rule when moving between Fourier and real space.

The experimental procedure is to initialize the code using the analytical velocity profile in the virtual channel and an arbitrary velocity field in the virtual solid such that there is a zero crossing at the virtual interface. In fact, we simply set the velocity at the first point below the interface to satisfy the no-slip condition at the virtual interface and everywhere else to be zero. The $v$ and $w$ components are initialized to zero. The flow is then advanced in time to steady state using the largest possible time step and, once a steady-state solution is obtained, the flow is examined to determine whether the boundary condition has been satisfied and how much the flow has crept from the analytical solution. As a reference, we perform the same experiment on the channel code using conventional boundary treatments.
We first test the explicit IBC using the Gaussian and sharp cosine force distributions described in Section 3.3.1. We then test the implicit IBC using the three forcing schemes described in Section 3.3.2: Scheme 1, which creates a smooth velocity distribution on the virtual solid with no forcing in the virtual channel; Scheme 2, which creates a smooth velocity distribution on the virtual solid and applies a smoothing force inside the channel; and Scheme 3, which applies force only near the virtual interface both inside the virtual channel and the virtual solid.

Figure 4.1 shows the steady-state velocity profiles using the explicit IBC. In Figure 4.1a, the velocity profile is shown over the entire domain, virtual channel and virtual solid alike; the analytical profile and velocity distribution with conventional boundary treatment are also shown for reference. The virtual interfaces are located at -1 and 1 and indicated with a dashed line. Qualitatively, all of the velocity profiles are correct — they all appear to have zero velocity on the virtual wall and a maximum velocity of 1. Figure 4.1b provides a more detailed view of the virtual channel wall. The dashed vertical lines on the plot have been shifted to show simulation resolution; a solid line has been added at the virtual interface. The zero-crossing that satisfies the IBC is clear for both the sharp cosine and Gaussian force distributions. However, it is apparent that the two implicit schemes have different slopes at the virtual interface — the sharp cosine distribution preserves the wall shear of analytical solution and the simulation with conventional boundary treatment whereas the Gaussian distribution increases drag at the wall. The reason for this will become more apparent as we compare the distribution of body force required to hold the virtual boundary condition for the explicit and implicit schemes.

As can be seen in 4.1c, the increase in wall shear in the Gaussian forcing case has the effect of decreasing the maximum velocity in the channel under constant pressure-gradient conditions. For a turbulent case at a higher Reynolds number where bulk
Figure 4.1: Comparison of laminar channel flow profiles with explicit IBC to analytical solution $u = 1 - y^2$ (——) and channel code without virtual boundary ----. Gaussian scheme indicated with --- and sharp cosine scheme indicated with ----. No-slip boundaries located at $y = 1$ and -1 (a) Profile on whole domain. (b) Close-up of flow near virtual solid. The Gaussian forcing scheme clearly does not preserve slope at the virtual wall. (c) Blow up of channel centerline. Error in slope at the wall causes Gaussian forcing to have erroneous maximum velocity. (d) Convergence history showing residual as a function of number of state iterations. Second-order convergence slope provided for reference.
flow has been held constant, using Gaussian forcing would require higher pressure
gradients to drive the flow and give a misleadingly high measure of drag on the
channel walls. Note that both the sharp cosine distribution and the analytical solution
provide excellent estimates of the maximum velocity in the channel; the velocity
profile predicted by the IBC with a cosine distribution of force is even closer to the
analytical solution than the channel code with conventional boundary treatments. We
attribute this to the extensive iteration procedure required to solve the explicit IBC
state equation. During the iteration, all of the errors in the approximate factorization
are reduced considerably, leaving primarily truncation error to cause a discrepancy
with analytical results. The channel code with conventional boundary treatments
has errors due not only to truncation, but also to the approximate factorization.
Furthermore, with the virtual wall inside the computational domain, it is subject to
second order accurate finite differencing whereas the code with conventional treatment
uses one-sided, first order accurate schemes on the boundary.

Figure 4.1d shows convergence of a single substep of the iterative procedure de-
scribed in Section 3.1. The plot shows convergence at slightly better than second
order and saturation near machine zero, approximately $10^{-18}$. In other words, for
this problem, the iterative procedure for the explicit IBC is able to drive solutions of
the split and factored systems to a solution of the whole.

Figure 4.2 shows results from testing the implicit formulation. Plots (a) and
(b) show that all of the forcing schemes give velocity profiles nearly identical to the
analytical solution and the simulation with conventional boundary treatment. In (b),
the difference between treatments on the virtual solid is evident. Clearly viscosity
is insufficient to smooth out the transition from forced grid sites to unforced grid
sites in the virtual solid when using Scheme 3. This appears to have little effect,
however, on the accuracy of the interpolated boundary condition. All of the methods
Figure 4.2: Comparison of laminar channel flow with implicit IBC Schemes 1, 2, and 3 to analytical solution \( u = 1 - y^2 \) and channel code without virtual boundary. No-slip boundaries at \( y = 1 \) and \(-1\); flow on virtual solid is shown. (a) Curves are indistinguishable to the eye. (b) Blow up of flow on virtual solid in region indicated in (a). The smooth profiles are Schemes 1 and 2; the sharp profile is Scheme 3. (c) Error between analytical solution, Schemes 1, 2, and 3, and channel code without the IBC. Error for Schemes 2 and 3 overlap and have negative values near the boundary; error for Scheme 1 and channel code without IBC overlap and have positive error near boundary.
Figure 4.3: Force profiles for the various IBC schemes: diamonds are explicit Gaussian, pluses explicit sharp cosine, triangles implicit Scheme 3, boxes implicit Scheme 2, and crosses implicit Scheme 1. In implicit Schemes 2 and 3, forcing inside the channel \((y > -1)\) in nearly undetectable. Schemes 1 and 2 are nearly identical.

...preserve shear at the wall, including those that apply a force inside the virtual channel.

It is apparent that the velocity profile is nearly linear in this region, anyway, so the approximation is valid. Figure 4.2c shows the errors in the solutions with conventional boundary treatment and the implicit IBC schemes relative to the analytical solution. In all cases, error is less than 0.05%. Scheme 1, with forcing only on the virtual solid, preserves the error structure of the code with conventional boundary treatment.

The schemes with forcing inside the virtual channel slightly underestimate velocity, particularly near the boundary. In some sense, this is a manifestation of the same problem incurred with the Gaussian distribution of force in the explicit IBC case.

Some insight can be gained on the failure of the Gaussian force and the relative success of the explicit sharp cosine force and all of the implicit schemes by examining Figure 4.3. In short, all of the successful force profiles add a negligible amount of momentum inside the virtual channel. Assuming that force is applied inside the
virtual solid such that there is a no-slip boundary at the virtual wall, viscous friction inside the channel slows down the fluid near the wall in precisely the right way to create the correct boundary layer. Any additional forcing inside the channel only adds or removes momentum and will change the near-wall velocities and alter the wall shear. The Gaussian force clearly applies a considerable amount of force inside the virtual channel, so the problem is worst in this case. The implicit schemes that apply force to linearize the velocity profile inside the channel are removing only a negligible amount of momentum from the flow and the effect is minimal. Nevertheless, it can be detected in a detailed plot such as Figure 4.2c.

These results suggest that the IBC has the potential to be as accurate as the underlying numerical method, a claim supported more firmly in Section 4.2. With schemes that either do not apply force inside the virtual channel or are careful to preserve the near linearity of the velocity profile near the wall, the implicit and explicit formulations are comparable in accuracy. Therefore, the explicit IBC — which accrues considerable computational expense in iteration to obtain good solutions to the unfactored state matrix — will not be examined any further. It does teach the lesson, however, that forcing schemes must be carefully examined to ensure they do not destroy the flow physics. Forcing inside the virtual channel clearly has the potential to destroy the accuracy of a simulation, particularly if the force profile is not chosen carefully.

4.2 Tollmien-Schlichting Wave Growth

The Tollmien-Schlichting wave problem is excellent for evaluation of the accuracy of a numerical simulation: it is exceptionally sensitive to mean flow profile and shear at the channel walls. In this problem, the Immersed Boundary Condition is again used to describe two parallel, infinite flat walls bounding a pressure-driven laminar
flow. Then Tollmien-Schlichting waves — small, carefully chosen disturbances in the pressure and velocity distributions — are added to the mean flow. The evolution of these disturbances in both time and space is known analytically [32].

There is extensive reference material available on the topic of Tollmien-Schlichting waves and the stability of flows to supplement the presentation here. For an introduction, refer to [32, 127] and the references therein. In summary, the exact nature of these disturbances is determined in the following way: A divergence-free solution, \((u, v, w, p)\), to the incompressible Navier-Stokes equations with constant density and viscosity is chosen; here, we use the laminar base flow described in (4.2). The continuity and momentum equations are then written for the base flow plus a small disturbance, \(u + \epsilon u', v + \epsilon v', w + \epsilon w'\) and \(p + \epsilon p'\), where the disturbances have been normalized by the maximum amplitude of \(u'\) and \(\epsilon\) is a small number; we use \(10^{-4}\) to ensure linear evolution over the time-span of the simulations. The equations describing the base flow are subtracted off, leaving the governing equations for the disturbance, which can then be linearized. These equations are further simplified by assuming that the base flow is locally parallel, \(i.e.,\) that derivatives in the stream-wise direction are negligible. For the channel flow case, the velocity component \(v\) is zero; for other shear flows, the component of velocity normal to the plane of shear is negligible. Complex exponentials are then used to describe disturbances that grow in time, are sinusoidal in \(x\), and are a function of space in \(y\), \(i.e.,\) Tollmien-Schlichting waves:

\[
\begin{pmatrix}
\dot{u}' \\
\dot{v}' \\
\dot{w}' \\
\dot{p}'
\end{pmatrix}
= \begin{pmatrix}
\hat{u}(y) \\
\hat{v}(y) \\
\hat{w}(y) \\
\hat{p}(y)
\end{pmatrix}
\exp[i\alpha x - i\omega t] + c.c.
\tag{4.3}
\]

Here, \(\omega\) is the wave frequency, \(\alpha\) is the wave number, \(i = \sqrt{-1}\), and \(c.c.\) denotes the complex conjugate. The functions \(\hat{u}(y), \hat{v}(y)\) and \(\hat{w}(y)\) (not to be confused with the
Figure 4.4: Initial profiles of $\hat{u}_r(y, x = 0)$, $\hat{v}_r(y, x = 0)$, and $\hat{p}_r(y, x = 0)$. The disturbances are constant in $z$ and are periodic $x$. All disturbances are normalized by $|\hat{u}|_{\text{max}}$. Before being added to the mean flow profile, the disturbances are multiplied by a scalar $\epsilon$ that is three to four orders of magnitude smaller than the maximum base flow velocity.

non-solenoidal hat velocities of Chapter 2) have both real and imaginary components, as do $\omega$ and $\alpha$, in general. Use of solutions of this type in the simplified disturbance equations and extensive further massaging yield the Orr-Sommerfeld equation, a homogeneous, linear, fourth-order differential equation [32]. Fixing the streamwise wavenumber, $\alpha$, and allowing the disturbance to grow in time, the Orr-Sommerfeld equation is an eigenvalue problem in $\omega = \omega_r + i\omega_i$ and the disturbance stream function, which is subject to no-slip boundary conditions. Some solutions grow in time, others decay, and still others are neutrally stable. For example, in the channel with $Re = 10,000$ considered here, the unstable solution has $\omega = 0.23752649 + 0.00373967i$ for $\alpha = 1.0$; the real parts of $\hat{u}(y), \hat{v}(y)$ and $\hat{w}(y)$ are shown in Figure 4.4.\footnote{Orr-Sommerfeld solution provided by Alexander Dobrinsky, Rice University, (2000).} The maximum amplitude of the wave in $x$ varies in time as $e^{\omega t}$ — if $\omega_i > 0$, the disturbance is temporally unstable.

For this validation study, we add Tollmien-Schlichting disturbances to the laminar channel flow and compare their temporal and spatial evolution with the analytical solution. We consider the solution given above for $Re = 10,000$ with $\epsilon = 10^{-4}$ and the channel non-dimensionalized as discussed in Section 4.1. We begin by using the IBC
code to compute a steady-state laminar base flow. As described in Section 4.1, the code is initialized using the analytical solution and then stepped forward in time to a steady profile. The numerical error caused by solving discretized equations causes the steady-state laminar flow profile to differ slightly from the analytical solution, and these transients must be eliminated as they are on the order of the Tollmien-Schlichting waves that are added to the profile. After the simulation has sufficiently converged, the Tollmien-Schlichting waves are scaled and added to the base flow, taking care to satisfy both the immersed and the conventional boundary conditions to reduce transients. The evolution of the disturbances is then tracked as a function of space and time and compared to the analytical solution. By testing the IBC with an array of different time steps and spatial resolutions, the order of the dominant error terms in the discrete state equation can be determined.

The procedure of placing a laminar flow in the channel and stepping through the transient to a steady-state profile, however, requires too much CPU time to perform for each spatial resolution we wish to test. Instead, we choose to identify the numerical source terms that cause each simulation to differ from the analytical solution and intentionally subtract them from the discrete state equation so that there is no start-up transient. To verify that this does not significantly alter the results of the experiment, several trials were performed by attaining a steady-state profile without subtracting the numerical source terms; results were comparable. Determining the source term required to eliminate the numerical transient is relatively trivial and much the same as calculating the body force required to satisfy an immersed boundary condition [60]. We simply write the discrete momentum equation that we normally solve, e.g. from (3.20)

\[(I - \beta_k \Delta t M) \hat{u}^k = r^{k-1} - 2\beta_k \Delta t f^k,\]

then write the momentum equation with analytical velocity profile \( \hat{u}^a \) and a numerical
source \( s^k \)

\[
(I - \beta_k \Delta t M) \hat{u}^a = r^{k-1} - 2\beta_k \Delta t f^k + s^k,
\]

then subtract to find \( s^k \):

\[
s^k = (I - \beta_k \Delta t M)(\hat{u}^a - \hat{u}^k).
\]

Because the profile does not change in \( x \) or \( z \) and there is no \( v \) component, the solution is already divergence free and remains unmodified by the projection to a divergence-free space. A source term must be calculated for each of the three RK substeps, but may be used throughout the simulation. We chose to define the source term only on the virtual channel and allow the IBC to dictate the flow in the virtual solid. As can be seen in Figure 4.5, we did not initialize the flow on the virtual solid to satisfy the IBC; as a consequence, the numerical source term is different for the first full time step from the remaining time steps. Thus we calculate a source term for the first two time steps and re-use the source term from the second time step for the remainder of a simulation.

As shown in Figure 4.5, the IBC channel code does a remarkable job predicting the spatial and temporal evolution of disturbances even though they are four orders of magnitude smaller than the base flow. Figures 4.5ace show that the simulated disturbances match the analytical solution to visual precision. The accuracy of the no-slip constraint is evident in the magnified plot of the \( u \) disturbance in Figure 4.5a. Note that the initialization error on the virtual solid is visible for the initial disturbance curve; the virtual boundary condition eliminates this error in the first substep. In the detail plot of the \( v \) disturbance, Figure 4.5d, the accuracy of the no-penetration boundary condition is visible. Like the \( v \) profile itself, the disturbance profile is locally parabolic, satisfying continuity. Also note that the \( u \), \( v \) and \( p \) disturbances are
smooth across the virtual boundary and on the virtual solid for Scheme 1. In Figure 4.5d, \( \omega_i \) is shown as calculated from the \( u, v \) and \( p \) disturbances along with the analytical solution. The steady state error in \( \omega_i \) for this case is approximately 0.3%.

By employing \( \omega_i \) as an error metric and noting how it varies as a function of the spatial and temporal resolution, we can determine the error structure of the numerical method augmented with IBC. Running two sets of simulations, one where we resolve very well in space, varying time step and another where we resolve very well in time, varying spatial resolution, we are able to isolate the two dominant error terms. Note that the code is exact in the \( x \) and \( z \) directions for this problem due to the spectral method. In the temporal resolution study, \( N_y = 400 \) and \( \Delta t = 0.4, 0.3, 0.2, \) and 0.1. For the spatial resolution study, \( \Delta t = 0.01 \) and \( N_y = 50, 100, 200, 400, \) and 1000. Since \( \omega_i \) is known analytically and can be calculated from the amplitude of the disturbances in the channel, we use the discrepancy between \( \omega_i \) in the channel and the analytical reference as the error metric. By simply locating and recording the maximum amplitude of the disturbances in \( u, v \) and \( p \) in the virtual channel, \( \omega_i \) can be calculated \textit{a posteriori} using second order finite differencing in time. For example, denoting the maximum amplitude of the disturbance in \( u \) as \( \hat{u}_{\text{max}} \), from (4.3),

\[
\hat{u}_{\text{max}}(t) = \hat{u}_{\text{max}}|_{t=0} e^{\omega_i t}. \tag{4.4}
\]

Then

\[
\frac{d\hat{u}_{\text{max}}}{dt} = \hat{u}_{\text{max}}|_{t=0} \omega_i e^{\omega_i t}. \tag{4.5}
\]

Dividing (4.5) by (4.4),

\[
\frac{d\hat{u}_{\text{max}}/dt}{\hat{u}_{\text{max}}} = \omega_i, \tag{4.6}
\]

so \( \omega_i \) can be calculated with second order finite differences at time \( t_n \)

\[
\omega_i(t_n) = \frac{\hat{u}_{\text{max}}(t_n + \Delta t) - \hat{u}_{\text{max}}(t_n - \Delta t)}{2\hat{u}_{\text{max}} \Delta t}. \tag{4.7}
\]
Figure 4.5: (a)-(e) Example results for $N_y = 400$ and $\Delta t = 0.01$ for implicit IBC Scheme 1, ____, compared to analytical solution, ----. All results normalized by initial maximum amplitude of $u$-disturbance, $\epsilon = 10^{-4}$. The disturbance convects through channel with a period of $t \approx 26$. Profiles shown are at $t = 0, 3, 5, 6, 8, 10, 12$ for $x = z = 0$. (a) Streamwise disturbance across channel. (b) Blow-up of boxed region from (a) examining no-slip condition at wall. (c) Wall-normal component of velocity. (d) Blow up of boxed region from (c) examining no-penetration condition. (e) Pressure disturbance. (f) Example of $\omega_i$ as a function of time, showing a steady state error of approximately 0.3%: —— analytical solution, ---- $\omega_i$ calculated from $u$, ------ $\omega_i$ calculated from $v$, ------- $\omega_i$ calculated from $p$. Results shown are for $N_y = 100$, $\Delta t = 0.01$ to best display transient.
Figure 4.6: (a) Convergence plots, $N_y$ on right side, $\Delta t$ on left side. First and second order convergence reference lines shown in upper left corner. Lines for implicit IBC Schemes 1, 2, and 3 labeled S1, S2, and S3. Lines for code with conventional boundary treatment labeled N.

Note that analytically, $\omega_i$ is a constant. Numerical error, however, causes a transient in $\omega_i$ as shown in Figure 4.5f. The transient can be considered finished when all measures of $\omega_i$ are equal. Our error metric uses the steady-state value of $\omega_i$, which has generally converged by the time the disturbance has convected through the channel twice.

The results of the resolution study on the channel code with conventional boundary treatments, implicit IBC Scheme 1, Scheme 2, and Scheme 3 are shown in Figure 4.6. For reference, the slopes for first- and second-order convergence are provided. As expected, the code with conventional boundary treatments is second order in space.
and time. IBC Schemes 1, 2 and 3 all preserve the spatial error structure, but Scheme 3 is clearly not quite second order in time. It appears to converge at approximately order 1.5. The discontinuity in the velocity field derivatives where the streamwise force is abruptly turned off in the virtual solid accounts for the loss in accuracy.

4.3 Laminar Flow with Wall-Normal Acceleration

In order to validate the code for moving walls and flat walls that are not coincident with grid sites, we introduce the model problem of incompressible laminar channel flow under wall-normal acceleration. Again, we non-dimensionalize as in Section 4.1. The initial condition in this problem is a steady-state laminar flow in the $x$ direction between two parallel, infinite flat plates a distance $2\delta$ apart in $y$ as discussed in Section 4.1. The initial streamwise velocity profile is constant in the $x$ and $z$ directions and varies in $y$ as

$$u(x_2) = u_m \left[1 - (x_2 - C_L)^2\right],$$

where $C_L$ is the centerline of the channel at $t = 0$ and $u_m$ is the streamwise velocity at the channel centerline. The flow is driven by a steady streamwise pressure gradient.
that would maintain the maximum $u$-velocity $u_m$ in a steady channel:

$$
\frac{\partial p}{\partial x_1} = -\frac{2u_m}{Re}.
$$

(4.9)

Immediately after the flow has been initialized, the channel walls begin to oscillate in the $y$ direction; they remain flat and parallel, a distance of $2\delta$ apart as shown in Figure 4.7. Whereas this problem was first used as a qualitative test of the IBC, the discrete solutions suggest a likely form for the analytical solution that can be tested in the governing equations and validated. The discrete solution, which appears in Figure 4.8 along with the analytical solution, appears to maintain a parabolic profile in $u$ between the two moving walls. The wall normal velocity, $v$, is equal to the wall velocity throughout the channel and the spanwise component of velocity, $w$, is zero. One might guess, then, that (4.8) is still valid for the streamwise component of velocity in the moving channel, except that $C_L$ is a function of time. This possible solution may be plugged into the Navier-Stokes (2.1) and continuity equations (2.2) to check that it is valid. First, because the proposed velocity field is independent of $x$ and $z$, the $x_1$ and $x_3$ derivatives are dropped; all $w$ terms are also dropped as $w = 0$ in the proposed velocity field. Thus the continuity equation becomes

$$
\frac{\partial v}{\partial x_2} = 0,
$$

(4.10)

which would not only prove that the wall-normal velocity must be constant in space, but also gives another simplification to the momentum equations, which are now just

$$
\frac{\partial u}{\partial t} + v_w(t) \frac{\partial u}{\partial x_2} + p_{x_1} - \frac{1}{Re} \frac{\partial^2 u}{\partial x_2^2} = 0
$$

(4.11a)

$$
\frac{\partial v}{\partial t}(t) + p_{x_2} = 0,
$$

(4.11b)
where \( v_w(t) \) is the velocity of the wall as a function of time and \( p_{x_1} \) is the steady streamwise pressure gradient. Clearly, once \( v_w(t) \) is specified, the wall-normal pressure gradient \( p_{x_2} \) is also determined. Plugging the proposed velocity distribution (4.8) with \( C_L = C_L(t) \) into the left hand side of (4.11a) gives the expression

\[
\frac{\partial}{\partial t} \left[ u_m - u_m (x_2 - C_L)^2 \right] + v_w(t) \frac{\partial}{\partial x_2} \left[ u_m - u_m (x_2 - C_L)^2 \right] + \\
p_{x_1} - \frac{1}{Re} \frac{\partial^2}{\partial x_2^2} \left[ u_m - u_m (x_2 - C_L)^2 \right].
\]  

(4.12)

Evaluating the derivatives, the left-hand-side expression becomes

\[
\left[ -2u_m (x_2 - C_L) \left( -\frac{\partial C_L}{\partial t} \right) \right] + v_w(t) \left[ -2u_m (x_2 - C_L) \right] + \\
p_{x_1} - \frac{1}{Re} \left( -2u_m \right).
\]  

(4.13)

Noting that \( \frac{\partial C_L}{\partial t} = v_w(t) \) and \( p_{x_1} = -2u_m/Re \) from equation (4.9), it is clear that the first and second terms cancel as well as the third and fourth, which means that the left and right sides of equation (4.11a) balance and this is a valid solution. Note that \( v_w \) was never specified, so it need not be sinusoidal at all.

For this validation study of the IBC, however, we prescribe the motion of the walls

\[
y_{wl} = -\delta + \alpha \sin \omega t \]  

(4.14a)

\[
y_{wu} = \delta + \alpha \sin \omega t \]  

(4.14b)

where \( y_{wl} \) and \( y_{wu} \) are the locations of the lower and upper walls, respectively. Their velocities are clearly

\[
v_w = \alpha \omega \cos \omega t
\]  

(4.15)

so that

\[
p_{x_2} = -\alpha \omega \cos \omega t.
\]  

(4.16)
Figure 4.8 compares the implicit IBC Scheme 3 with the analytical solution for sinusoidally oscillating channel flow with $\omega = 2\pi$, $Re = 100$, and the streamwise pressure gradient set such that $u_m = 1$. Figure 4.8a shows the streamwise velocity profile at several instants during the first period of oscillation; comparison is excellent between the analytical and computational solutions. Figure 4.8b shows that the immersed boundary is held perfectly and that wall shear is preserved. The time histories of $p_y$ and $v$ in Figures 4.8c and 4.8d are also dead-on. The success of IBC in predicting the wall-normal pressure gradient is evidence that pressure has been handled properly. Results using the Poisson equation to find pressure and using pseudopressure as an approximation give equally satisfactory results in this case.
Figure 4.8: Velocity and pressure gradient plots for a channel in wall-normal oscillation. Implicit IBC with Scheme 3: --- . Analytical solution: ---- . (a) IBC and analytical streamwise velocity at \( t = 0.0, 0.1, 0.2, 0.6, 0.7 \). (b) Enlargement of box in (a) to show virtual boundary. (c) IBC and analytical wall normal velocity as a function of time. (d) IBC and analytical pressure gradients \( p_y \) as a function of time.
4.4 Laminar Flow in a Grooved Channel

To validate the IBC code with deformed immersed boundaries, we simulate laminar flow in a ribbed channel. The solution to this problem is well documented both by Mohd-Yusof [87], who modified the B-Spline/Fourier pseudospectral code of Kravchenko et al. [70] for immersed boundaries, and by Choi, Moin, and Kim [24], who give both analytical similarity solutions to the problem and computational results using a body-fitted, multigrid method.

As shown in Figure 4.9, fluid is driven by a streamwise pressure gradient between two infinite, periodic, riblet-covered planar walls. The riblets are aligned such that the peaks and valleys run in the streamwise direction. To minimize the computational cost of solving this problem, we solve for flow over only one riblet, allowing the natural periodicity in the Fourier directions to account for the adjacent riblets. The distance between riblet valleys is defined as $L_z$ and the distance from the lower riblet centerline to the upper riblet centerline is $2\delta$. The angle of the triangular riblets is measured with respect to the $x$-$z$ plane. As such, the problem of laminar flow in a channel with flat walls described in Section 4.1 is the degenerate case of this problem with $\alpha = 0$. Note that for a fixed $L_z$ and $\delta$, the channel area is constant regardless of $\alpha$.

The problem performed by both Mohd-Yusof [87] and Choi et al. [24] was to set the channel aspect ratio, $L_z/\delta$, and vary the riblet angle $\alpha$ to determine the impact on bulk flow. One can plot the bulk flow as a function of spanwise resolution to determine how dense the computational grid must be to resolve a deformed wall with sharp corners — or, more accurately in the case of IBC, determine how dense the constraint field on a flow must be to properly determine the effect of flow over a deformed virtual interface. When using an Immersed Boundary Condition, the resolution of a simulation may be perfectly suited to resolving fluctuations in the pressure and velocity field, but ill-suited to resolving the immersed boundary, or vice
versa. Consider that in the streamwise and spanwise directions, the fractional step code described in Chapter 2 uses a Fourier method capable of resolving a periodic variation in the flow with just two modes. Two constraints on the flow, however, is clearly insufficient for forcing the flow to have an immersed boundary of nearly any type, even if the flowfield required to satisfy the immersed boundary condition is just a sinusoid. On the other hand, the IBC may be used to simulate a wire suspended in the flow, a problem requiring many Fourier modes to resolve the cusp in the streamwise velocity component, but only a few constraints.

In the explicit formulation of the IBC, one can place as many constraints on the flowfield as necessary, regardless of the resolution of the simulation, as long as there are enough linearly independent force profiles. In the implicit formulation, however, one can only put forces at grid sites and there can only be as many constraints as forces. The shape of the virtual wall is determined by the constraints. So, no matter how well-resolved the flow is in Fourier space, the virtual wall must still be resolved in real space on the collocation points. One must always keep in mind that the virtual wall is resolved with the number of collocation points even if the flow is resolved to spectral accuracy. The problem of laminar flow over a riblet-covered wall addresses
the issue of how many collocation points are required to resolve a virtual surface.

The streamwise pressure gradient is chosen such that \( u_{\text{max}} = 1 \) for \( \alpha = 0^\circ \). We investigate riblets with \( \alpha = 30^\circ, 45^\circ, \) and \( 60^\circ \) for the \( L_z/\delta = 5 \). As shown in [24], the laminar velocity field in the channel is independent of Reynolds number, but to guarantee laminar flow, \( Re = 100 \) here. For \( \alpha = 30^\circ, N_y = 116 \), for \( \alpha = 45^\circ, N_y = 144 \), and for \( \alpha = 60^\circ, N_y = 292 \). The number of wall normal grid points was generally chosen such that further increase in resolution had negligible impact on the steady-state bulk flow of the case most well-resolved in the spanwise direction. Each simulation was initialized with zero velocity and advanced in time to steady state. The final bulk flow was then normalized by the bulk flow with \( \alpha = 0^\circ \) in a laminar channel of the same cross sectional area with the same streamwise pressure gradient and plotted in Figure 4.10 along with the results of Mohd-Yusof [87] and Choi et al. [24]. Note that Choi et al. employed \( N_y = 129 \) in his wall-normal, finite-difference direction. Mohd-Yusof does not specify the number of points used in the B-spline/Fourier spectral code, although plots showing resolution of riblets on the computational grid indicate \( N_y \approx 300 \).

As shown in Figure 4.10, spanwise resolution required for grid-independence is a strong function of \( \alpha \), indicating that for a given computational grid, there is an upper limit on the allowable deformation of the wall before the fluid/virtual-wall interaction becomes under-resolved. This is a binding, but certainly not surprising requirement.

Roughly 32 spectral modes, or 48 collocation points on the 3/2 dealiased mesh, were sufficient to resolve the wall-fluid interaction of both the 30\(^\circ\) and 45\(^\circ\) riblets — 16 modes were sufficient for convergence to less than 0.2\% error and 8 modes were sufficient for less than 2\% error. This is quite impressive, especially in light of the sharp points of the riblets causing high gradients in the flow. It is conceivable that smoother wall structures with similar amplitudes and widths would require even
Figure 4.10: Convergence of laminar flow over riblet-covered walls: ratio of bulk flow for riblets with $\alpha = 30^\circ$, $45^\circ$ and $60^\circ$ to bulk flow with $\alpha = 0^\circ$ as a function of spanwise resolution. Pluses are Mohd-Yusof [87], diamonds are Choi et al. [24], and crosses are Scheme 3.

less resolution for comparable quality results. The sharpest riblet, with $\alpha = 60^\circ$, proved a more difficult problem. Even with $N_y = 292$ and $N_z = 32$, the riblet-flow interaction shows signs of being under-resolved, as evidenced by the slight overshoot in Figure 4.10. Additional resolution in $y$ would eliminate the overshoot. The slower convergence as a function of spanwise resolution is attributable to the increasingly high number of Fourier modes required to reproduce the sharper cusp in the flowfield above the riblet point; as shown in Figure 4.11, there are certainly enough constraints to adequately enforce the Immersed Boundary Condition.

The implicit Scheme 3 was the only viable IBC method for this problem due to the sharpness of the riblet tips and valleys. Schemes 1 and 2 both interpolate the
Figure 4.11: Contour plots of the streamwise velocity profile in the half-channel, showing the $u = 0$ isosurface representing the immersed boundary. Contours are from $-0.493$ to $0.986$ spaced by $0.017$. (a-c) Contours for $\alpha = 30^\circ$ with $N_z = 8$, 16, and 32, respectively; (d-f) for $\alpha = 45^\circ$ with $N_z = 8$, 16, and 32, respectively; and (g-i) for $\alpha = 60^\circ$ with $N_z = 8$, 16, and 32, respectively.
flow in the $y$ direction across the virtual interface, thus reproducing the sharpness of the riblet in the $u$ velocity profile along $x-z$ planes. Naturally, the Fourier method was unable to resolve the cusps forced in the velocity profile on the virtual solid and unmitigated by viscosity; the resulting Gibbs phenomena permeated the channel.

This validation study proved quite educational. Although it showed the IBC effective at solving very challenging computational problems, it also revealed several limitations in the method. It illustrated that implicit IBC Schemes 1 and 2 are ineffective for describing sharp boundaries because they reproduce the sharp features of the wall in the flowfield on the virtual solid, causing trouble with the spectral method. The case study also revealed that the resolution requirements of the flowfield and the virtual wall are not necessarily the same and that walls with sharp and high aspect ratio features may require exceptional grid resolution to achieve converged solutions. On the other hand, even for such computationally unfriendly geometries as riblets, engineering-accuracy solutions are accessible at very reasonable computational grids — error in the bulk flow ratio for the troublesome 60° riblet was less than 4% with only 8 collocation points in the spanwise direction.
Figure 4.12: Geometry for a turbulent channel flow with Immersed Boundary Conditions.

4.5 Turbulent Flow Statistics

As one of the prime applications of the Immersed Boundary Condition in this code is to simulate turbulent flow over irregular surfaces, it is critical to validate the IBC code for turbulent flow. Because of the chaotic nature of turbulent flows, however, analytical solutions are not available; thus, common practice is to compare flow statistics of a new code to an established reference. In this case, we compare statistics for IBC turbulence simulations with the original code, which was extensively validated as shown in [21]. We simply use the unaugmented code to simulate a turbulent channel flow and calculate first- and second-order statistics. The computational domain is then enlarged in the $y$ direction and two flat virtual walls are added so that the size of the virtual channel is precisely the same as in the original code. The simulation is then run again, and the statistics are compared.

In the conventional simulation, the computational domain is $(4\pi\delta, 2\delta, 4\pi\delta/3)$, where $\delta$ is the channel half-height; in the IBC simulation, the computational domain is approximately $(4\pi\delta, 2.4\delta, 4\pi\delta/3)$, where two extra slabs of size $(4\pi\delta, 0.2\delta, 4\pi\delta/3)$ at the top and bottom of the domain are occupied by a virtual solid. Dimensions in the
figures are cited in wall, or plus, units, where
\[ t^+ = \frac{u^2}{\nu}, \quad y^+ = \frac{y_1}{\nu} \quad \text{and} \quad u^+ = \frac{u}{u^*}. \] (4.17)

The Reynolds number for these simulations is
\[ Re_r = \frac{u_r \delta}{\nu} = 100. \] (4.18)

In the streamwise and spanwise directions, 32 Fourier modes are used with 3/2 rule
dealiasing. For the conventional simulation, 48 points are used in the wall-normal
direction; 68 points for the IBC simulation. The extra 10 planes parallel to the
virtual wall at the top and bottom of the domain are in the virtual solid.

To facilitate presentation of the statistics, we define a few new symbols: Angular
brackets, e.g. \( < u > \), represent a quantity averaged in the periodic directions. An
apostrophe is used to denote the fluctuation of a quantity about its periodic mean,
e.g. \( u' = u - < u > \). An overbar, e.g. \( \bar{u} \), represents a quantity that has been averaged
in the periodic directions and then in time. The statistics we present, all functions of
\( y \), are

- mean flow, \( \bar{u} \)
- root-mean-square \( u \)-velocity, \( u_{rms} = (\overline{u'u'})^{1/2} \)
- root-mean-square \( v \)-velocity, \( v_{rms} = (\overline{v'v'})^{1/2} \)
- root-mean-square \( w \)-velocity, \( w_{rms} = (\overline{w'w'})^{1/2} \)
- Reynolds stress, \( \tau_{Re} = \overline{u'v'} \)
- total stress, \( \tau = \frac{1}{Re_r} \frac{\partial \varepsilon}{\partial y} - \overline{u'v'} \)

and

mean Smagorinsky SGS model coefficient, \( \tilde{C} \).

The correspondence between statistics for the IBC simulation and the original
simulation is excellent. The first order statistics — mean-flow in Figure 4.13 and
total stress in Figure 4.16a — correspond so well that individual lines on the plots
Figure 4.13: Mean flow profile on half of the channel in wall units with $Re_t = 100$ using (a) a regular scale, and (b) a semi-log scale with the law of the wall plotted as $\cdots$. Log law is not shown because the buffer region extends to the center of the channel for $Re_t = 100$. Statistics from a simulation without the Immersed Boundary Condition are indicated $\cdots$, with body forcing on the virtual interface and virtual solid only indicated $\cdots\cdots\cdots$, and body forcing at the virtual interface, on the virtual solid, and at the first point inside the flow indicated $\cdots\cdots\cdots$.

are indistinguishable. Reynolds stress, shown in Figure 4.15b, shows the same degree of correspondence.

Second order statistics — $u_{rms}$ in Figure 4.14a, $v_{rms}$ in Figure 4.14b, and $w_{rms}$ in Figure 4.15a — also show commendable correspondence. Second order statistics generally require a considerably larger number of samples in time to converge than first order statistics, accounting for some of the discrepancy seen in the plots. The largest discrepancy occurs in the center portion of the channel where the wall-normal grid is the coarsest and the flow structures are largest, taking longest to evolve in time and average out. In any case, problems caused by the IBC would most likely appear near the virtual boundary where the statistics have converged perfectly to the reference case.

The mean statistics of the LES model coefficient described briefly in Section 2.2
Figure 4.14: Root-mean squared profiles in wall units on the half channel for $Re_\tau = 100$ of (a) the $u$ component of velocity and (b) the $w$ component of velocity. Statistics from a simulation without the Immersed Boundary Condition are indicated ———, with body forcing on the virtual interface and virtual solid only ————, and with body forcing at the virtual interface, on the virtual solid, and at the first point inside the flow ————.

deserve specific comment. Whereas the IBC is specifically designed to generate acceptable velocity profiles, no specific effort is made to ensure that the model coefficient, as predicted by the dynamic model [41, 81], performs correctly. From Figure 4.16b, however, one can see that the mean value of the model coefficient is indeed the proper function of $y$ on the virtual channel, converging to zero at the virtual wall. Similar performance has also been noted by Verzicco et al. [121]. The LES model coefficient is computed from spatial averages in the homogeneous planes — flat walls in these planes do not defeat the spirit of this averaging. In the case of deformed walls, however, $x$-$z$ planes near the virtual interface are no longer statistically homogeneous; in fact, some points in the plane may be in the virtual solid and others well inside the viscous sublayer. Similar problems with filtering and averaging are common to other LES implementations for irregular geometries [42]. We are provided some measure of comfort that deformed walls do not cause the LES model to generate unphysical effects, though, by the fact that the model coefficient is close to zero throughout the
Figure 4.15: Profiles in wall units on the half channel for $Re_{\tau} = 100$ of (a) $u_{rms}$ and (b) $\tau_{Re}$. Statistics from a simulation without the Immersed Boundary Condition are indicated —— , with body forcing on the virtual interface and virtual solid only ---- , and with body forcing at the virtual interface, on the virtual solid, and at the first point inside the flow ··········.

viscous sublayer and on the virtual solid. Thus deformed walls in turbulent flows with modest amplitudes — e.g. virtual surfaces that model walls and flow control actuators less than five wall units tall — do not breach $x$-$z$ planes where the LES model is most active, and it is a reasonable assumption that the method remains valid.
Figure 4.16: Mean values for $Re_\tau = 100$ on the half channel of (a) the total stress, and (b) the LES model coefficient. Statistics from a simulation without the Immersed Boundary Condition are indicated —— , with body forcing on the virtual interface and virtual solid only ---- , and with body forcing at the virtual interface, on the virtual solid, and at the first point inside the flow ---------.
Chapter 5

Results

This chapter presents the results of several control experiments that employ the implicit Immersed Boundary Condition. The IBC is used to create deforming upper and lower walls in a channel flow, modeling the motion of actuators embedded in the channel wall. The actuators are ultimately used in control schemes that attempt to reduce drag on the walls. The results in this chapter represent only a brief foray along the avenues of research facilitated by the computationally efficient IBC formulation, including exploration of actuator geometries and the investigation of control schemes.

Our first attempt at flow control is made using an analog [66] to the opposition control scheme of Gad-el-Hak [39] and Choi, Moin and Kim [26]. Here, instead of using transpiration on flat channel walls, the motion of a continuously deforming boundary adds wall-normal momentum to the flow in an attempt to cancel coherent turbulent structures. The continuously deforming wall opposition control scheme, first presented by Choi and Kang [66], is easy to implement and provides insight on the characteristics of a deformed surface that is effective in flow control. Armed with this information, we then proceed to explore the use of discrete actuators.

To begin, we examine the effect of a single, stationary, wall-mounted sinusoidal bump on laminar flow. The reaction of laminar flow to a simple deformed surface provides a qualitative understanding of the advantages and disadvantages of such actuators that may be generalized to more complex surfaces in turbulent flows [83]. A bipolar version of the sinusoidal bump is then used in unsteady, open loop control. The parameters of the bump and the control scheme are investigated in some detail, considering the impact of actuator aspect ratio, oscillation frequency, and relative

86
phase on drag reduction in a turbulent channel. We then experiment with the limiting case of these sinusoidal actuators — stationary, sinusoidal riblets. These passive control structures are investigated primarily as an extension of the parameter study, although much smaller-scale triangular riblets have been employed for drag reduction by many research groups [25, 27, 31, 51]. Finally, the bipolar sinusoidal actuators are employed in closed loop, active control schemes to achieve the goal of this research — simulating discrete, MEMS-like actuators in closed loop, active control of turbulent flows.

5.1 Experimental Setup

For the turbulent flow cases in the sections that follow, the box size and non-dimensionalization are as discussed in Section 4.5, with \( \delta \) the mean channel half-height in deformed wall cases. The friction velocity \( u_r \), defined in Section 2.1 remains the reference velocity, although for control cases we continue to non-dimensionalize with the no-control \( u_r \). All turbulent flow control experiments are conducted at \( Re_r = 100 \).

Throughout this chapter the box size is \( (4\pi\delta, 2.4\delta, 4\pi\delta/3) \), where two extra slabs of size \( (4\pi\delta, 0.2\delta, 4\pi\delta/3) \) are added at the top and bottom of the domain to allow for wall deformation. The channel is discretized with 32 Fourier modes in the streamwise and spanwise directions, using 3/2 dealiasing between real and Fourier space. There are 68 finite difference grid sites in the wall-normal direction, the top and bottom 10 used to discretize the slabs added for wall deformation. Implicit IBC Scheme 3 is used in all of these tests.

In turbulent channel flow with constant mass flux, drag on the walls varies in time about a statistically steady-state mean. Under control, there is a transient after actuation begins as the flow settles into a new statistical steady state with either higher or lower average drag. Mathematically, one can show that the mean pressure
gradient and the sum of viscous and pressure drag on the walls must balance; thus, maintaining a constant mass flux through the channel requires continuous adjustment of the mean streamwise pressure gradient. Because drag on the walls and the negative of the streamwise pressure gradient that holds the bulk flow are identical, we report $-\bar{p}_{\lambda x}$ as the total drag.

The bulk flow, $\bar{U}$, is given by

$$\bar{U} = \frac{1}{\Omega} \int_{\Omega} u d\Omega,$$  \hspace{1cm} (5.1)

where $u$ is the streamwise velocity and $\Omega$ is the volume of channel, either with flat or deformed walls. For flat surfaces, computing this integral discretely in a way that is consistent with the numerics of Chapter 2 is trivial:

$$\bar{U} = \frac{1}{2\delta} \int_{-\delta}^{\delta} \hat{u}_o \, dy = \frac{1}{2\delta} \sum_{j=1}^{N_x} \hat{u}_{o_j} \Delta y_j,$$  \hspace{1cm} (5.2)

where $\hat{u}_o$ is the Fourier mode representing the mean value of $u$ in each $x$-$z$ plane and $j$ is the index of the plane in the $y$ direction.

Calculating the bulk flow for the case of deformed walls is more complicated, however, as the location of the virtual interface does not necessarily coincide with grid sites. To be consistent with the numerics, the virtual boundary must be defined as the isosurface of zero $u$ that passes through the constraint sites. Integrating $u$ over the volume between the upper and lower isosurfaces is not feasible, however. Instead, we take advantage of the fact that the location of the virtual interface is known where the $u$ component of velocity is constrained to be zero. We integrate, then, with trapezoidal rule in the $y$ direction between the constrained zero crossings of $u$. In the $x$ and $z$ directions, we would like to integrate in Fourier space in a manner consistent with the spectral method, but the virtual surface is defined in the collocation point space, so we resort to midpoint rule. The discrete version of (5.1)
is then
\[ \bar{U} = \frac{B_I(u)}{B_I(1)}, \quad (5.3) \]
where the integral operator \( B_I \) acting on \( u \) is defined in the notation of Chapter 2 as
\[ B_I(u) = \sum_{i=0}^{3/2N_x-1} \sum_{j=1}^{N_y-1} \sum_{k=0}^{3/2N_z-1} \frac{1}{2} (\bar{u}_{i,j,k} + \bar{u}_{i,j+1,k})(\bar{y}_{j+1} - \bar{y}_j) \Delta x \Delta z. \quad (5.4) \]

For grid points inside the boundary, \( \bar{y} \) are the \( y \) locations of grid points. For grid points on the virtual solid, \( \bar{y} \) is the \( y \) location of the virtual interface. The quantity \( \bar{u} \) is defined as \( u \) inside the virtual channel and zero on the virtual solid.

The mean streamwise pressure gradient, \( \bar{p}_{1,1} \), required to maintain the bulk flow can be determined by integrating the \( x \)-momentum equation (2.5) over the virtual channel with \( B_I \):
\[ B_I \left( u_{1,t} + (u_1 u_j)_j - 2 (\nu S_{1j})_j + p_{1} + f_1 \right) + B_I \left( \bar{p}_{1,1} \right) = 0. \quad (5.5) \]

Rearranging (5.5) and setting \( B_I[u_{1,t}] = 0 \) so that the bulk flow does not change as a function of time, the mean streamwise pressure gradient is simply
\[ \bar{p}_{1,1} = \frac{-B_I \left( (u_1 u_j)_j - 2 (\nu S_{1j})_j + p_{1} + f_1 \right)}{B_I(1)}. \quad (5.6) \]

The pressure gradient is calculated according to (5.6) at every iteration of every sub-step and updated on the right hand side of the state equation. Due to the numerical inconsistency between the discretization of the momentum equations and the bulk flow integral, there is unsteady error in the mean pressure gradient for moving, deformed surfaces. This unsteady error manifests itself as low amplitude, high frequency noise in the mean pressure gradient. To ease congestion in drag history plots, this noise has been removed in postprocessing with a low-pass filter.
5.2 Opposition Control with a Continuous Deforming Wall

One of the simplest closed loop control schemes for reducing drag in a turbulent channel flow is the opposition, or heuristic, scheme of Gad-el-Hak [39] and Choi, Moin, and Kim [26]. In opposition control, the wall-normal velocity component is sensed in a plane some small distance from the channel wall. The velocity sensed in this plane is then opposed with transpiration at the channel wall such that

\[ v(x, z, y = y_w) = -v(x, z, y = y_s), \]

where \( y_w \) is the location of the wall and \( y_s \) is the location of the sensing plane. In general, the distance between the wall and the sensing plane is approximately 15 wall units, which roughly corresponds to the distance from the channel walls to the centerline of the turbulent rolls primarily responsible for momentum flux in the \( y \) direction near the walls. Choi et al. [26] showed empirically that sensing planes both closer and further from the channel wall gave less effective control results. Note that this scheme is not realistically feasible, however, as MEMS sensors generally only have access to information at the wall. There is potential for eliminating this infeasibility in other opposition-control-like schemes, though; it has been shown that there is a known correlation between spanwise shear \( \partial w/\partial y \) at the wall and turbulent coherent structures in the near-wall region [75].

As in work by Choi and Kang [66] at \( Re_r = 140 \), instead of using wall-normal transpiration to impart \( y \) momentum to the near-wall region, momentum may be added by deforming the channel wall itself as shown in Figure 5.1. Initially, the wall moves such that its velocity in the \( y \) direction opposes \( v \) in the sensing plane: \( v_{wall}(x, z, y = y_w) = -v(x, z, y = y_s) \). Clearly, the wall cannot be allowed to breach
**Figure 5.1:** Schematic showing continuously deforming wall opposition control. The channel wall is deformed to oppose wall-normal velocity in a plane about fifteen wall units above the mean height of the virtual interface.

the sensing plane, however, so it is constrained to move only within a certain distance of its mean location. Limiting the amplitude of the continuously deforming wall is convenient for the IBC, as well, because sharp, high amplitude deformations require highly refined $y$ grids as discussed in Section 4.4. Note that the mean location of the wall must remain constant throughout the simulation due to the incompressible flow assumption. The boundary conditions satisfied at the wall are that $v$ must be the wall-normal velocity of the virtual surface and that the $u$ and $w$ components of velocity must be zero.

Defining $y_{wall}$ as the variation of the wall height about its mean location and $y_{wallmax}$ as the maximum allowable deformation, the unsteady motion of the channel boundary can be calculated at each substep with explicit Euler in time:

$$\bar{y}_{wall}^k(x, z) = y_{wall}^{k-1}(x, z) - 2\beta_k \Delta t v^{k-1}(x, z, y_s),$$

where the tilde denotes that this is just a preliminary calculation — we have not guaranteed that the wall stays within its spatial confines. As long as the wall does not exceed its preset maximum height, *i.e.* if $|\bar{y}_{wall}^k| < y_{wallmax}$, the preliminary wall
deformation is final and \( \bar{y}_{wall}^k = y_{wall}^k \). If the preliminary wall location exceeds the height limitation, however, the wall is rescaled according to

\[
y_{wall}^k(x, z) = \frac{y_{wall}^k(x, z)}{y_{wall}^k(x_m, z_m)} y_{wall}^{max} \tag{5.8}
\]

where \((x_m, z_m)\) is the location of the maximum amplitude of the wall. Note that once the wall has reached its maximum amplitude, the \(v\)-velocity of the wall is no longer \(-v(x, z, y_s)\), but \(-v(x, z, y_s) + v(x_m, z_m, y_s)\). Thus, no control is applied at the point of maximum amplitude. Furthermore, at locations where \(v(x, z, y_s)\) and \(v(x_m, z_m, y_s)\) have opposite signs and the magnitude of \(v(x_m, z_m, y_s)\) is greater, the control is reinforcing to coherent turbulent structures in the flow. This fact may account for the more favorable results of traditional opposition control with wall-normal transpiration. Traditional opposition control gives drag reductions of about 20% [26].

We can see the effect of the arbitrary wall deformation limit on drag reduction in Table 5.1. Drag reduction relative to the no-control case of a flat, undeformed wall is calculated for runs with \(y_{wall}^{max} = 0.01\delta, 0.02\delta, 0.04\delta\) and \(0.06\delta\). As shown in Table 5.1, time-averaged drag reduction ranged from 3.5% to 14.3%. Best results were achieved with \(y_{wall}^{max} = 0.06\delta\). Note that, unlike opposition control with wall transpiration, the surface area of a moving wall increases with deformation. Increased surface area is one possible cause for the tapering of drag reduction with large wall deformations. Large wall deformations also induce additional turbulence in the flow;

<table>
<thead>
<tr>
<th>( y_{wall}^{max} )</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01\delta</td>
<td>3.5%</td>
</tr>
<tr>
<td>0.02\delta</td>
<td>6.5%</td>
</tr>
<tr>
<td>0.04\delta</td>
<td>11.3%</td>
</tr>
<tr>
<td>0.06\delta</td>
<td>14.3%</td>
</tr>
</tbody>
</table>

**Table 5.1:** Drag reduction for opposition control with continuously deforming walls
Figure 5.2: Drag histories with deforming wall opposition control for maximum wall deformations of 1% of channel half-height, $\delta$, 2% of $\delta$, 4% of $\delta$, 6% of $\delta$ and no deformation, as marked. (a) Drag reductions over no-deformation case are 3.5% for $0.01\delta$, 6.5% for $0.02\delta$, 11.3% for $0.04\delta$, and 14.3% for $0.06\delta$. (b) The rate of drag reduction during the transient improves monotonically as maximum allowable wall deformation is increased.

it is possible that this negative effect begins to overcome the drag reducing properties of the scheme at wall deformations nearing $0.06\delta$.

At $Re_\tau = 140$, Choi et al. [66] get approximately 17% drag reduction for continuously deforming wall opposition control with a maximum wall deformation of $0.05\delta$. The slight discrepancy between our results and those of Choi et al. is attributed to differing maximum allowable wall deformations, Reynolds numbers, and sensing plane locations. We use $y_s^+ = 16$, frequently taken as the most effective sensing plane height for opposition control with wall-normal transpiration [26]; Choi et al. [66] choose $y_s^+ = 10$, a possibly more effective location for opposition control with deforming walls. Control efficacy is a strong function of sensing plane height.

Figure 5.2 shows drag histories for the no-control case of a flat, undeformed wall and deforming wall opposition control with $y_{wall max} = 0.01\delta$, $0.02\delta$, $0.04\delta$ and $0.06\delta$. The transient as the flow settles to a new statistical steady state under control is evident for $t$ approximately less than five. As shown in Figure 5.2b, the rate of drag reduction during the transient improves monotonically as the maximum allowable
wall deformation is increased.

The mean and root-mean-square statistics discussed in Section 4.5 are shown in Figure 5.3 for the base case of an uncontrolled flow with no wall deformation and flows subjected to continuously deforming wall opposition control with various maximum wall deformations. The reduction in wall shear of the controlled flows over the uncontrolled flow and the law of the wall is clearly visible in the mean velocity profiles of Figures 5.3ab. The trend that velocity fluctuations subside throughout the channel as the allowable wall deformation increases is apparent in the root-mean-square statistics of Figures 5.3cde, indicating that control actively reduces turbulence. In particular, fluctuation in the $v$ component of velocity shows not only a dramatic reduction at its peak near $y^+ = 45$, but also a local minima at approximately $y^+ = 7$, shown vividly in Figure 5.3f. This local minima near the channel wall is typical of opposition control using wall-normal transpiration, as well. As discussed by Hammond et al. [54] and elaborated on by Prabhu et al. [104], this minima represents a barrier to momentum transport in the wall-normal direction. The barrier is critical to drag reduction because a significant fraction of the drag in turbulent flow is caused by wall-normal convection of high-momentum fluid from the buffer layer toward the wall by turbulent rolls. The barrier to wall-normal momentum transport inhibits this process, thus reducing drag.

Figures 5.3cde also show that there is a small non-zero fluctuation in the $u$ and $v$ components of velocity and an even smaller fluctuation in $w$ at the mean location of the virtual wall, at $y^+ = 0$. At any given time, some points in each of the $x-z$ planes between the maximum and minimum allowable deformation of the wall may be in the virtual solid, in the virtual channel, or on the interface. Thus it is expected that averaging in all of these planes will give non-zero root-mean-square velocity fluctuations.
Figure 5.3: Statistics for no control (NC) and continuously deforming wall opposition control with $y_{wall_{max}} = 0.01, 0.02, 0.04$, and $0.06$, as labeled: (a) mean flow, (b) mean flow on semi-log scale, (c) root-mean square $u$ velocity, (d) root-mean-square $w$ velocity, (e) root-mean square $v$ velocity, and (f) root-mean-square $v$ velocity near the virtual wall. The log law is not shown in (b) because the buffer region extends to the center of the channel for $Re_r = 100$. 
Figure 5.4: Wall deformation at $t = 0.12$, before saturation, for continuously deforming wall opposition control. Scaling in $y$ is exaggerated to show wall structure.

The deformation of the lower wall just before saturation is shown in Figure 5.4. Just as noted by Choi et al. [66], the wall deformation mimics the coherent structures in the flow. As such, we can clearly see the oscillations of the wall in the spanwise direction and the long, streak-like structures in $x$. The discrete actuator that we test for the remainder of the chapter is designed to incorporate these structural features.

5.3 Sinusoidal Membrane Actuator Study

For the remainder of the active turbulence control studies in Chapter 5, we employ discrete actuators that model a deformable membrane in the shape of bipolar sinusoidal bumps, e.g. in Figure 5.5a. A single bipolar membrane actuator has deformation about its mean height given by

$$y_{dipole} = A \sin \left( \frac{\pi}{l_z} (z - z_c) \right) \cos \left( \frac{\pi}{2l_x} (x - x_c) \right),$$

(5.9)
with $x \in (x_c - l_x, x_c + l_x)$ and $z \in (z_c - l_z, z_c + l_z)$. The location of the center of the actuator is given by $(x_c, z_c)$, its streamwise length by $l_x$, and the spanwise width of one pole by $l_z$. The actuator amplitude is $A$. We generally use bipolar membrane actuators in arrays that cover the upper and lower channel walls.

First, however, we study laminar flow over an isolated sinusoidal monopole membrane actuator, simplifying the problem to gain an understanding of the effect of such actuators on a more complicated flow. The deformation of the sinusoidal monopole actuator about the otherwise undeformed channel walls is given by

$$y_{\text{monopole}} = A \sin \left( \frac{\pi z - z_c}{l_z} \right) \cos \left( \frac{\pi x - x_c}{2l_x} \right), \ x \in (x_c - l_x, x_c + l_x), \ z \in (z_c, z_c + l_z).$$

(5.10)

To further simplify the problem, the actuator is stationary so that flow around it is
qualitatively at steady state. The laminar base flow is used in order to eliminate the
difficulty of distinguishing between flow structures caused by the actuator and those
arising naturally in turbulent flow.

For this laminar problem, we non-dimensionalize with \( u \) at the channel centerline
as the reference velocity. These experiments are conducted at a Reynolds number of
\( Re = 1600 \), roughly corresponding to the \( Re_c = 100 \) of the turbulence control cases.
The resolution and box size of these simulations also reflect those of a turbulence
simulation: \( (32,68,32) \) modes and grid points in the streamwise, wall-normal, and
spanwise directions. The width of the actuators is chosen to correspond to the period
of spatial oscillation of the continuously deforming wall — approximately 6 grid points
across a pole. The actuators are arbitrarily set to be square in the computational
space and are thus rectangular in physical space with the same aspect ratio as the
channel, three. The actuator amplitude, \( A \), is 0.04\( \delta \). Note that even with the stretched
wall-normal grid, the dipole membrane actuator is no more than 4 grid points tall.

Similar studies have been performed by Mason and Morton [83] with a cosine-
squared bump and by Carlson, Berkooz, and Lumley [18,82] with a Gaussian. As
Lumley and Blossey point out, one would expect the Gaussian bump to produce a
necklace vortex such as the ones generated by telephone poles and bridge supports
— a pair of counter-rotating vortices beginning at the shoulders of the bump and
convecting downstream. In fact, they found that the Gaussian bump produced a
considerably more complex vortex structure: a weak, necklace type vortex trailing
high over the bump, a pair of vortices caused by the flow splitting around the bump,
and a pair beneath these caused by vortex interaction with the wall [18,82].

Figure 5.6 highlights the important features of the steady-state flowfield around
our sinusoidal actuator. In Figures 5.6abc, the \( u \), \( v \) and \( w \) components of velocity
can be seen in the neighborhood of the actuator, rectangle \( A \) in the schematic. The
contour representing zero $u$ velocity designates the virtual interface; it looks precisely like a sinusoidal protuberance from a flat surface, which indicates that the flow is adequately constrained to model the boundary. In Figure 5.6a, the complexity of the flow on the virtual solid is evident, but the $u$ velocity in the virtual channel is what one would expect — streamwise velocity in the $x$-$z$ planes is slower over the bump than elsewhere in the channel. Likewise, the $v$ velocity in Figure 5.6b is not surprising. There is negligible $v$ velocity in the free-stream and a pocket of wall-normal momentum as the flow courses over the bump. In Figure 5.6c, the $w$ velocity contours indicate that very close to the virtual surface, the flow is deflected around the bump. Above the pockets of non-zero $w$ velocity caused by flow around the bump, there are larger, weaker pockets of $w$ velocity of opposite sign. These are good indicators of the vortices described by Lumley and Blossey. The vortices are even more evident in Figure 5.6d, which plots streamwise vorticity over the peak of the actuator. Note that Figures 5.6de take advantage of the symmetry plane that bisects the actuator, clearly showing two stacked vortices; the most intense vortex sits directly on the shoulder of the actuator. The top vortex is more diffuse and of opposite sign. As shown in Figure 5.6e, the weaker vortex does not persist downstream at $Re = 1600$. Viscosity weakens and spreads the stronger vortex into the whole channel.

The results shown in Figure 5.6 compare qualitatively to those shown by Lumley and Blossey, despite the differing geometries and Reynolds numbers. In the words of Lumley and Blossey [82], "this complicated flow is not particularly useful from a control point of view. ... We nevertheless managed to exert some control by pushing the high-speed streak away from the wall." Indeed, the bipolar membrane actuator described by (5.9) is employed primarily to generate wall-normal velocity. This momentum is used in the sense of the opposition control scheme described in Section 5.2 to attenuate the organization of coherent structures. The flow structures generated
Figure 5.6: A schematic of the problem, showing views A at the actuator peak and B at the actuator tail. Views extend from the bottom of the computational domain to the channel centerline and are wide enough to capture the dominant features of the flowfield. (a) Streamwise velocity with 13 contours evenly spaced between $u = 0.0$ and $u = 1.2$. (b) Wall-normal velocity with 10 contours between 0.0015 and 0.006. (c) Spanwise velocity with 21 contours between 0.015 and -0.015. (d) Streamwise vorticity in half of view A, with upper vortex comprised of 21 contours from 0.00 to 0.02 and lower vortex comprised of 21 contours from -0.4 to 0.0. Vorticity is antisymmetric about the $x$-$y$ plane containing the actuator peak. (e) Streamwise vorticity in half of view B with 11 contours ranging from 0.0 to 0.04.
by the wall motion are not used for control purposes, and could in fact be detrimental
to the control effort by introducing additional small-scale disturbances in the flow.

5.4 Open Loop Actuator Parameter Study

A bipolar version of the sinusoidal membrane actuator tested in Section 5.3 can be
incorporated easily into any number of open loop control schemes without taking into
account the complexity of its effects on the flow, albeit not necessarily in an intelligent
or effective way. Still, we understand that turbulent flows increase viscous drag
on surfaces because turbulent mixing moves high momentum fluid to the near-wall
region. Furthermore, we know that the coherent structures, the rolls in particular,
are a dominant factor in that mixing. It is reasonable, therefore, to assume that
destroying, weakening, or moving the coherent structures could have the effect of
reducing turbulent mixing in the all-important near-wall region, thereby reducing
drag. As such, even actuators that introduce new noise and small-scale structures to
the flow may still have a positive impact if they weaken the structures responsible for
mixing at the wall [38, 82].

This open loop actuator study explores the possibility that drag reductions can
be achieved even without the complexity of closed-loop strategies, just by weakening
and lifting the coherent structures from the wall. To that end, we employ an array
of bipolar sinusoidal actuators as shown in Figure 5.7, covering the entire upper and
lower channel walls. The amplitude of the actuator is varied sinusoidally in time so
that the deformation of the actuator about its mean height is as given in equation
(5.9), but where $A$ is a function of time:

$$ A = A_m \sin(2\pi ft). \quad (5.11) $$

Here, $A_m$ is the maximum amplitude of the actuator and $f$ is its temporal frequency
Figure 5.7: An array of dipole membrane actuators with $AR = 3$.

in cycles per non-dimensional time unit. The wall-normal velocity of the actuator surface can then be easily calculated as

$$
\dot{y}_{\text{dipole}} = A_m (2\pi f) \cos(2\pi ft) \sin \left( \frac{\pi z - z_c}{l_z} \right) \cos \left( \frac{\pi x - x_c}{2l_x} \right)
$$

(5.12)

for $x \in (x_c - l_x, x_c + l_x), z \in (z_c - l_z, z_c + l_z)$. The bipolar actuators in Figure 5.7 have $(l_x, l_z) = (\pi \delta, \pi \delta/3)$ and are resolved with 12 collocation points in the streamwise and spanwise directions.

As discussed in Section 5.3, the spanwise period of the actuators is chosen to correspond to the period of oscillation of the wall in the continuously deforming wall opposition case discussed in Section 5.2. This spanwise period also corresponds very
well to the approximate center-to-center eddy distance in turbulent flows, approximately 100 viscous units [61]. The aspect ratio of the actuators — the ratio of the streamwise length of the actuator to its width in the spanwise direction — is chosen to give the maximum possible control resolution in the streamwise direction without grossly under-resolving the actuator. These actuators will be used later in closed loop control, so the ability to act locally is important.

The channel resolution and non-dimensionalization are as described in the beginning of the chapter. We vary the temporal oscillation frequency $f$ of the actuators, their relative phase, and their aspect ratios to determine which physical parameters are most effective in open-loop control for drag reduction. Last, we address the limiting case of actuators with infinite frequency and infinite aspect ratio, i.e. static, sinusoidal riblets. All cases are run for 40 convective time units, corresponding to about 4,000 wall units.

5.4.1 Varying Frequency

The first parameter of the sinusoidal bipolar bumps we explore is their frequency of oscillation, $f$, ranging here between four and six cycles per convective time unit. In all trials, the aspect ratio is held constant at three. The relationship between actuator aspect ratio and control efficacy is explored in Section 5.4.3. The maximum amplitude, $A_m$, of the actuators is 0.02δ for all trials. The frequency and maximum amplitude of oscillation are chosen to reflect the root-mean-square wall-normal velocity and convective timescale at $y^+ = 15$, the most effective sensing plane height for opposition control [26].

Although the range of frequencies between four and six certainly does not constitute an exhaustive study, the actuators moving at five cycles per time unit showed a marked improvement in drag reduction over either four cycles or six cycles, suggesting
CHAPTER 5. RESULTS

Figure 5.8: Time histories of the negative mean pressure gradient required to maintain initial bulk flow as actuator frequency is varied. Curves shown are for no control (NC), $f = 4$, $f = 5$ and $f = 6$.

<table>
<thead>
<tr>
<th>$f$</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0.3%</td>
</tr>
<tr>
<td>5</td>
<td>3.2%</td>
</tr>
<tr>
<td>6</td>
<td>1.3%</td>
</tr>
</tbody>
</table>

Table 5.2: Effect on drag reduction of varying the temporal frequency of sinusoidal membrane actuators for open loop control.

a local minima in this neighborhood. Table 5.2 shows drag reduction for all three cases. The drag histories may be seen in Figure 5.8.

The results of applying open loop control indicate that wall motion — even if uncorrelated with developments in the flow — has a drag-reducing effect on a turbulent channel. It is possible that the secondary flows generated by the moving bipolar bumps have a disorganizing effect on the turbulent coherent structures sufficient to reduce mixing and consequently drag, even in spite of the increase in the surface area of the channel walls. Indeed, comparing the root-mean-square velocity fluctuation for
Figure 5.9: Statistics for no control (NC) and open loop control with temporal oscillation frequencies of $f = 4$, 5, and 6, as labeled: (a) mean flow, (b) mean flow on semi-log scale, (c) root-mean-square $u$ velocity, (d) root-mean-square $w$ velocity, (e) root-mean-square $v$ velocity, and (f) root-mean-square $v$ velocity near the virtual wall. The log law is not shown in (b) because the buffer region extends to the center of the channel for $Re_\tau = 100$. In (e) and (f), the wall motion produces a significant non-zero root-mean-square $v$ velocity near the virtual interface.
the controlled and uncontrolled flows in Figures 5.9cde, there is some reduction near the maxima of $w_{rms}$ and $u_{rms}$ for the most effective control frequency of $f = 5$ over uncontrolled flow, suggesting that the turbulence production in the flow is minimally abated. However, the change in turbulence statistics between controlled and uncontrolled flow is barely evident, shedding little light on the drag reducing mechanism at play. For the remaining open loop control schemes, turbulence statistics are even less informative and not included here.

A possible cause of the drag reduction induced by open-loop control is suggested by the research of Hussain and Schoppa [110], who achieved drag reduction in a minimal flow unit channel by superimposing streamwise vortices on the fluid; as shown in Section 5.3, addition of streamwise vorticity to the flow is the most significant effect of mounting sinusoidal bumps on the channel wall. Hussain et al. claim that adding streamwise vorticity to the fluid tends to diffuse the near-wall structures and convect them toward the channel boundary [110]. As such, the low-speed streaks — caused by convection of the slowly moving fluid near the boundary into the stream by near-wall vortices — are pushed back toward the wall and spread along the surface. It is an instability of the low-speed streaks that is frequently given responsibility for generating new turbulent rolls in the flow and thus establishing a self-sustaining cycle of turbulence [53, 62, 63, 109, 124]. By eliminating or attenuating the low-speed streaks with additional streamwise vorticity, the turbulence production cycle is interrupted, or at least disrupted, resulting in a possibly dramatic reduction of turbulent drag [110]. Clearly, the drag reductions achieved by open-loop control in the experiment presented here are not so dramatic, but the vorticity caused by the discrete sinusoidal actuators has a spanwise period of approximately 50 wall units, whereas the vorticity induced by Schoppa and Hussain had a spanwise period of roughly 400 wall units, the entire width of their minimal flow unit simulation. Furthermore, minimal flow units
— the smallest computational domains in which turbulence may be sustained [61] — are ideal for studying individual turbulent coherent structures, but may not provide adequate information for quantitative prediction of drag reduction.

Goldstein and Tuan [52] suggest yet another possible mechanism for the drag reduction caused by deformed walls. For channel boundaries with small deformations relative to the size of coherent structures in the flow, e.g. riblets and, by extension, bumps, the most elevated parts of the wall act to lift the coherent structures in the flow relative to the mean location of the surface. Thus, mixing caused by the turbulent rolls is unable to convect high-speed fluid from inside the channel as close to as much of the surface area of the channel wall. Indeed, Figures 5.9cde do show some shift toward the channel centerline of the peaks of the root-mean-square velocity fluctuations. It is difficult to say, however, whether this — or any of the suggested mechanisms — is the primary cause of the drag reduction in the present simulations; additional research is required to address this issue.

5.4.2 Varying Phase

The choice in Section 5.4.1 to move all of the sinusoidal dipole actuators in phase — i.e. to line up all of the actuator peaks in the streamwise direction and alternate the peaks and valleys in the spanwise direction — was based on the fact that turbulent streaks are generally long structures in the streamwise direction and would be most dramatically altered by actuators with a similar geometry. This, however, is an arbitrary assumption, so it makes sense to test out-of-phase actuators — ones in which the peaks alternate in both the streamwise and spanwise directions as in Figure 5.10.

Choosing the most effective frequency of oscillation from Section 5.4.1, $f = 5$, the turbulent flow is simulated again, this time with out-of-phase actuators. As seen
Figure 5.10: An array of out-of-phase dipole membrane actuators with $AR = 3$. 
Figure 5.11: Time histories of the negative mean pressure gradient required to maintain initial bulk flow as the spatial phase of actuators is varied. Curves shown are for no control (NC), in-phase actuators, and out-of-phase actuators.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Phase</td>
<td>3.2%</td>
</tr>
<tr>
<td>Out of Phase</td>
<td>1.6%</td>
</tr>
</tbody>
</table>

Table 5.3: Effect on drag reduction of varying the relative phase of sinusoidal membrane actuators for open loop control.

In Figure 5.11, there is little qualitative difference between the drag histories for in-phase and out-of-phase actuators. Nevertheless, in-phase actuators are about twice as effective at reducing drag, as shown in Table 5.3.

5.4.3 Varying Aspect Ratio

Based on the fact that arranging the actuators with all of the maxima and minima aligned in the streamwise direction is more effective at reducing drag, one might infer that making the actuators themselves longer in the streamwise direction would also
Figure 5.12: Arrays of dipole actuators with (a) $AR = 6$ and (b) with $AR = \infty$.

be beneficial. To test this hypothesis, we test actuators with an aspect ratio of six (Figure 5.12a) and infinity (i.e. constant in $x$, shown in Figure 5.12b) to compare with those of Section 5.4.1.

Unfortunately, as shown in Table 5.4, lengthening the actuators has a monotonically detrimental effect on drag reduction. Again, the drag histories are qualitatively similar.

<table>
<thead>
<tr>
<th>Aspect Ratio</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3.2%</td>
</tr>
<tr>
<td>6</td>
<td>1.9%</td>
</tr>
<tr>
<td>$\infty$</td>
<td>1.4%</td>
</tr>
</tbody>
</table>

Table 5.4: Effect on drag reduction of varying the aspect ratio of sinusoidal membrane actuators for open loop control.
Figure 5.13: Time histories of the negative mean pressure gradient required to maintain initial bulk flow as actuator aspect ratio is varied. Curves shown are for no control (NC), aspect ratio $AR = 3$, $AR = 6$, and $AR = \infty$.

5.4.4 Limiting Case — Sinusoidal Riblets

Up to now, we have tested actuators with several aspect ratios and unsteady oscillation frequencies. We have shown that higher aspect ratios and longer periods generally have a detrimental impact on control efficacy. Naturally, this does not lead one to think that the limiting case — the static, infinite aspect-ratio actuators shown in Figure 5.12b — would be particularly effective at reducing drag. Indeed, as shown in Table 5.5, they are not. The drag reduction for static riblets is even less than for the active, infinite aspect ratio actuators of Section 5.4.3, also shown on the drag history plot, Figure 5.14.

Nevertheless, the motivation to study the sinusoidal riblets lies more in their historical importance. Triangular riblets, such as those in Section 4.4, have long been studied as passive drag reducing devices in turbulent flow by Karniadakis et al. [31],
Figure 5.14: Time histories of the negative mean pressure gradient required to maintain initial bulk flow for the limiting case of steady sinusoidal riblets. Curves shown are for no control (NC), aspect ratio $AR = \infty$ with $f = 5$ and aspect ratio $AR = \infty$ with $f = \infty$.

Choi et al. [25], Goldstein et al. [51], and Bechert et al. [7, 8], just to name a few. In general, the triangular riblets that are most effective are those with a peak-to-peak spacing of 10 to 20 viscous units and a height of 5 to 15 viscous units [125]. The sinusoidal riblets studied here have a peak-to-peak spacing and height of 100 and 2 viscous units, respectively, corresponding to the actuator scale in the continuously deforming wall case. Thus, a meager drag reduction of 1.1% is not surprising, even relative to the 2.2% of Goldstein et al. [51] or the 8–10% generally accepted as the

<table>
<thead>
<tr>
<th>Actuator</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>active riblet</td>
<td>1.4%</td>
</tr>
<tr>
<td>passive riblet</td>
<td>1.1%</td>
</tr>
</tbody>
</table>

Table 5.5: Drag reduction for sinusoidal riblets.
drag-reducing capability of riblets [103]. Furthermore, it is the sharp points of triangular riblets jutting out of the viscous layer that are generally given credit for their drag reducing effects — a feature missing from sinusoidal riblets [51].

5.5 Closed Loop Control

5.5.1 Discrete Heuristic Control

The first attempt at closing the control loop and employing MEMS-like, discrete actuators in feedback schemes for drag reduction harkens again to the opposition control scheme of Section 5.2. Whereas in continuously deforming wall opposition control, the entire virtual surface moved without rigid structure to oppose wall-normal velocity in a sensing plane, with discrete actuators, less sensor information is used and the motion of the wall is more constrained. In this experiment, the average $v$ velocity is sensed only above one pole of the bipolar sinusoidal membrane actuator. This sensed velocity is opposed by adjusting the actuator amplitude at each timestep. The actuator maxima beneath the patch of sensors moves in the $y$ direction at the negative of the sensed velocity. When the actuator reaches the maximum allowable deformation, it simply remains still until the average velocity in the sensing patch changes direction.

For example, as given by equation (5.9), the deformation of a bipolar sinusoidal membrane actuator located at $(x_c, z_c)$ with length $l_x$ and poles of width $l_z$ is

$$y_{dipole} = A \sin \left( \pi \frac{z - z_c}{l_z} \right) \cos \left( \pi \frac{x - x_c}{2l_x} \right),$$

where $x \in (x_c - l_x, x_c + l_x)$ and $z \in (z_c - l_z, z_c + l_z)$. Here, $A$ is the maximum amplitude of the actuator. The velocity of the actuator in the $y$ direction, then, is simply given
CHAPTER 5. RESULTS

by

\[ \dot{y}_{\text{dipole}} = \dot{A} \sin \left( \pi \frac{z - z_c}{l_z} \right) \cos \left( \pi \frac{x - x_c}{2l_x} \right). \] (5.13)

Clearly, the velocity of the pole with a positive maxima for \( A > 0 \) is simply

\[ \dot{y}_{\text{dipole}} = \dot{A}. \] (5.14)

The sensing patches are located above the pole with a positive maxima for \( A > 0 \) at \( y_s \), fifteen wall units into the channel from the mean location of the deformed surface; the velocity read by the sensors is \( v_s \), where

\[ v_s = \frac{1}{2l_z l_x} \int_{x_c-\ell_z}^{x_c+\ell_z} \int_{z_c-\ell_z}^{z_c+\ell_z} v(x, z, y = y_s) \, dz \, dx. \] (5.15)

The maximum actuator deformation rate, \( \dot{A} \), simply opposes this sensor velocity, so

\[ \dot{A} = -v_s. \] (5.16)

Using the actuator aspect ratio of three shown most effective in the parameter study of Section 5.4.3 and a maximum deformation of 0.04\( \delta \), the control scheme gave a drag reduction of 4.8\% — not nearly as effective as the 14.3\% drag reduction of the continuously deforming wall case, but better than any of the open loop control schemes considered here. To verify that opposing the wall-normal velocity in the sensing plane in an attempt to cancel coherent structures is working, we can compare the control scheme to one that attempts to reinforce the structures by using

\[ \dot{A} = v_s. \] (5.17)

As shown in Table 5.6, even reinforcing the coherent structures produced a drag reduction, but only of 1.2\%. Drag histories may be seen in Figure 5.16 and a snapshot
Figure 5.15: Array of dipole actuators at $t = 1.0$ during discrete heuristic control.

of the wall deformation at $t = 1.0$ in Figure 5.15. The deformation agrees with expectation — the actuators move approximately in phase in the $x$ direction in response to the long structures in the streamwise direction.

The success of reinforcing control at reducing drag suggests that although some of the drag-reducing capability of the actuators may arise from the intended abatement of coherent structures, the motion of the actuators alone — even in tandem with the coherent structures — has a positive impact on drag on the channel walls. Indeed, as explored earlier, actuator motion uncoordinated with developments in the flow produced drag reductions ranging from 0% to 3%. In Figures 5.17cde, it is evident that the root-mean-square velocity fluctuation is slightly greater for reinforcing
<table>
<thead>
<tr>
<th>Scheme</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancellation</td>
<td>4.8%</td>
</tr>
<tr>
<td>Reinforcement</td>
<td>1.2%</td>
</tr>
</tbody>
</table>

Table 5.6: Drag reduction for closed loop discrete heuristic control schemes.

Figure 5.16: Time history of drag for no control (NC), discrete opposition control (dhc), and discrete reinforcing heuristic control (drc).
Figure 5.17: Statistics for no control (NC), opposition control, and reinforcing control with discrete dipolar actuators: (a) mean flow, (b) mean flow on semi-log scale, (c) root-mean-square $u$ velocity, (d) root-mean-square $w$ velocity, (e) root-mean-square $v$ velocity, and (f) root-mean-square $v$ velocity near the virtual wall. The log law is not shown in (b) because the buffer region extends to the center of the channel for $Re_T = 100$. 
CHAPTER 5. RESULTS

control than for an uncontrolled flow, indicating that even though drag is slightly reduced, turbulence in the channel has been minimally exacerbated. Root-mean-square velocity fluctuation is slightly reduced for discrete opposition control.

The facts that reinforcing control only slightly increases root-mean-square velocity fluctuation in the channel and does not increase drag suggest that the control resolution, i.e. the size of the actuators relative to the coherent structures in the flow, is insufficient. Although the bipolar sinusoidal membrane actuators mimic the aspect ratio and period of spatial oscillation of the continuously deforming wall case, their location is fixed and they will not generally be situated between rolls — a major limitation.

Figure 5.17f shows no evidence of the local minima in the root-mean-square $v$ velocity fluctuation seen for continuously deforming wall opposition control in Figure 5.3, suggesting that the mechanism for drag reduction in these cases is more similar to that of the open-loop control schemes discussed in Section 5.4.1.

5.5.2 Discrete Control with Spanwise Shear Sensors

Whereas control using information gathered inside the virtual channel is easy to implement in numerical simulations, it is infeasible in practical applications. In practice, information is only available from wall-mounted sensors. Nevertheless, it stands to reason that there must be some correlation between coherent turbulent structures and wall quantities. For example, as shown in Figure 5.18, the variation of the $w$ component of velocity across a streamwise vortex leads to an easily identifiable pattern of spanwise shear at the wall. In fact, Lee, Kim and Choi [75] show that to optimally reduce an instantaneous cost functional of spanwise shear on the channel wall, the distribution of wall-normal suction and blowing required is simply proportional to a filtered $z$ derivative of spanwise shear, $\partial/\partial z(\partial w/\partial y)$. Likewise, earlier work using a
Figure 5.18: Spanwise shear $\frac{\partial w}{\partial y}$ is calculated at seven points upstream of each actuator. A linear combination of the spanwise shear is taken that approximates spanwise differentiation. The $z$ derivative of spanwise shear locates vortices.

neural network to predict actuation based on spanwise shear at the wall determined the same thing: that suction and blowing should be proportional to the linear combination of values of spanwise shear that approximate a $z$ derivative [6]. Gradients of pressure at the wall may also be used effectively in feedback control schemes [68,69].

In this scheme, the actuators are moved such that the maxima of the poles that are positive for $A > 0$ have a velocity proportional to a linear combination of values detected by spanwise shear sensors. That is, a spanwise row of sensors is placed on the line of zero deflection just upstream of each actuator, as shown in Figure 5.18. These sensors detect spanwise shear, $\partial w/\partial y$. A weighted linear combination of the spanwise shear detected by the sensors is multiplied by a gain factor and used as $\dot{A}$ in (5.13) and (5.14). The width of one pole of the sinusoidal membrane actuators is
seven collocation points, so the seven-point stencil of spanwise shear used in [75] fits perfectly into the actuator array. Thus

\[
\hat{A} = G \sum_{k=c-3}^{c+3} W_k \left. \frac{\partial w}{\partial y} \right|_k,
\]

(5.18)

where \( G \) is the gain factor and \( c \) is the center of the stencil, corresponding to the \( z \) location of the aligned actuator maxima. At location \( k \), the weights of the sensor readings \( \left. \frac{\partial w}{\partial y} \right|_k \) are \( W_k \), as given by Lee et al. [6]:

<table>
<thead>
<tr>
<th>( k )</th>
<th>( c - 3 )</th>
<th>( c - 2 )</th>
<th>( c - 1 )</th>
<th>( c )</th>
<th>( c + 1 )</th>
<th>( c + 2 )</th>
<th>( c + 3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( W_k )</td>
<td>-2/3</td>
<td>0</td>
<td>-2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2/3</td>
</tr>
</tbody>
</table>

The gain factor, \( G \), is chosen as suggested by Choi and Kang [66]: such that the root-mean-squared actuator velocity roughly approximates the RMS \( v \) velocity in the plane 15 wall units above the mean location of the virtual surface. Drag histories for control runs with gain factors between 0.001 and 0.012 are shown in Figure 5.19. As seen in Table 5.7, a gain factor of \( G = 0.004 \) gave the best drag reduction — approximately 4.9%. Thus, forgoing the practically unrealistic sensing schemes of opposition control in favor of using information available only at the wall from a discrete array of shear sensors has little detrimental impact on the efficacy of control using these bipolar sinusoidal actuators. The drag reductions are comparable.

<table>
<thead>
<tr>
<th>Gain factor</th>
<th>Drag Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.001</td>
<td>2.1%</td>
</tr>
<tr>
<td>0.002</td>
<td>3.1%</td>
</tr>
<tr>
<td>0.004</td>
<td>4.9%</td>
</tr>
<tr>
<td>0.008</td>
<td>2.0%</td>
</tr>
<tr>
<td>0.012</td>
<td>1.6%</td>
</tr>
</tbody>
</table>

Table 5.7: Effect on drag reduction of varying the gain factor \( G \) in discrete actuator control using spanwise shear sensors.
Figure 5.19: Drag history using discrete control with spanwise shear sensors. Curves shown are for no control (NC) and gain factors of $G = 0.001, 0.002, 0.004, 0.008,$ and $0.012$.

The statistics shown in Figure 5.20 for discrete control based on wall sensor information are also qualitatively comparable to the discrete opposition control statistics in Figure 5.17, suggesting that the linear combination of spanwise shear values approximating a $\partial / \partial z$ derivative do an admirable job of allowing this control scheme to emulate discrete opposition control with a sensing plane.

5.6 Summary

In work by others at various Reynolds numbers, wall-normal transpiration has been used in closed loop control schemes giving as much as 40% drag reduction with unlimited knowledge of the flow [13, 21] and as much as 20% drag reduction using information available at the wall only [74]. Passive control schemes such as riblets have given as much as 10% drag reduction [103]. Here, using opposition-control-like sensing and continuously deforming walls, we have produced a 14% drag reduction at $Re_r = 100$. Using sinusoidal membrane actuators with opposition control sensing,
**Figure 5.20:** Statistics for no control (NC) and discrete control with spanwise shear sensors using gain factors $G = 0.001, 0.002, 0.004, 0.008, \text{ and } 0.012$: (a) mean flow, (b) mean flow on semi-log scale, (c) root-mean square $u$ velocity, (d) root-mean-square $w$ velocity, (e) root-mean square $v$ velocity, and (f) root-mean-square $v$ velocity near the virtual wall. The log law is not shown in (b) because the buffer region extends to the center of the channel for $Re_r = 100$. 
drag reduction diminished to 4.8%. A drag reduction of 4.9% can maintained using only wall information. Closed loop and passive control schemes were tested giving drag reductions between 1% and 3%. So this work has certainly not shown deforming wall actuators a superior alternative to wall-normal transpiration, nor has it developed any new or particularly successful control strategies or actuator geometries. We have, however, shown that the IBC is an effective tool for inexpensively modeling unsteady, deformed walls of nearly arbitrary shape. We have also shown that the Immersed Boundary Condition is an effective tool for evaluating control strategies and the effects of actuators on near-wall flow. And we have employed the IBC to explore just one actuator geometry in only a few of an infinite number of the possible control schemes — only a first stab at control using discrete actuators.

The goal of determining the most effective actuator geometries and how they should be employed in control schemes remains to be accomplished in future work. Indeed, just designing a good discrete actuator is a difficult task — as shown here, deforming wall actuators are troublesome when attempting to reduce drag because they increase the surface area of walls, introduce small-scale disturbances to the flow, and have a limited range of control resolution. Finding a control scheme that takes advantage of well-designed actuators is also a whole field of research unto itself. In the final chapter, we suggest a few avenues of inquiry that take advantage of the IBC formulation.
Chapter 6

Future Work and Conclusions

6.1 Future Work

Although the only MEMS-type discrete actuators tested in Chapter 5 were of the sinusoidal membrane type, there are any number of possible actuator shapes that may be investigated. As shown in Section 5.3, actuators have a dramatic effect on the flowfield around them: designing an actuator to produce a specific type of flowfield is quite a difficult task. It is possible that some actuator shapes would have a minimal effect on the flow around them except to generate wall-normal momentum, and such an actuator would be ideal for opposition-control-like schemes. On the other hand, with a concrete grasp of the relationship between the flowfield created by an actuator and the actuator geometry, one could design an actuator with a shape ideal for canceling coherent structures in flow at a given Reynolds number. As suggested by Glowinsky [46], the ability of IBC to simulate flow around unsteady shapes without regridding makes this method ideal for shape optimization problems. The rigorous framework of an optimization approach would be ideal for an orderly investigation of the actuator shape that is best suited to a given flow control problem.

More work remains in the vein of the parameter study conducted here, as well. Exhaustive investigations of actuator size, for example, would be beneficial to determine the length scales of actuators that allow for sufficient control resolution while still being large enough to have a useful effect on the flow. Monopole actuators of various shapes should be investigated as well, along with a means of using them in an incompressible channel flow such that the channel volume does not change. Pursuant to the goal of modeling MEMS actuators, an in depth study of a realistic MEMS
actuator model should also be conducted.

The development of flow control schemes for use with discrete, MEMS-type actuators is not yet even in its infancy. Nearly all of the existing flow control schemes have been constructed with wall-normal transpiration in mind. Instead of adapting control schemes intended for use with suction and blowing to discrete, deforming actuators, new control schemes must be invented that are tailored to the advantages and disadvantages of actuation with unsteady, irregular wall geometries. Using discrete MEMS-like actuators merely as wall-normal momentum generators overlooks both the potential uses and the detrimental effects of the secondary flows they generate. Whereas the detailed analysis of actuator shapes and the secondary flows that result from them may heuristically suggest new approaches to control, a more organized approach would likely yield better results. For example, application of an optimal control framework to the problem of using discrete actuators to minimize a cost functional of turbulent kinetic energy or wall drag may generate control schemes that achieve drag reductions comparable to those using wall-normal transpiration and suggest effective new suboptimal control schemes.

Another means of designing control schemes specifically for use with discrete actuators is via adaptive techniques. The transfer function that links information detected by sensors to the motion of the actuators may be determined by a neural network, one means of determining transfer functions by experimentation [74, 130]. Training a neural network to move the actuators such that drag on the virtual surface is reduced would generate a control strategy that locally minimizes the control objective. An alternative to having the neural network learn the transfer function between sensor readings and actuator motion is to train the neural network to predict the behavior of the flow in response to arbitrary actuator motion. As in Model Based Predictive Control [113], an optimization algorithm could then determine at each timestep the
actuator motion for which the neural network predicts the lowest drag on the channel walls and these controls used as the simulation is advanced.

No doubt there are still many, many more avenues to explore on the path to developing successful actuators and complementary control schemes. And, of course, there are a whole slew of problems of industrial and academic interest that involve flow over irregular or moving surfaces that the IBC may be used to solve, even beyond the scope of flow control.

6.2 Conclusions

The Immersed Boundary Condition represents a powerful tool for investigating all of these issues by simulating fluid flow over moving, irregular surfaces without accruing the computational expense of recalculating the grid each time the fluid-solid interface moves. In this work, implicit and explicit formulations of the IBC have been derived for a numerical method common in turbulent flow simulations — a fractional step algorithm with finite difference and Fourier spectral discretizations in space. The Immersed Boundary Condition serves here as a complement to the dynamic turbulence model of Large Eddy Simulation, a tool that eases the computational expense of simulating turbulence by incorporating the effects of the smallest disturbances in velocity without demanding the grid refinement required to resolve these features.

The validation studies conducted here suggest that the LES model and Immersed Boundary Condition work reasonably well together and address the limitations and parameters of the IBC: that the immersed boundary must be well-resolved in the collocation space of spectral methods and that forcing schemes must be chosen carefully to avoid unphysical effects in the flow and instability in the underlying numerical method. The best IBC schemes are shown to preserve the error structure of the underlying numerical method.
CHAPTER 6. FUTURE WORK AND CONCLUSIONS

The value of the Immersed Boundary Condition to research in flow control is evident in the brief tour de force of Chapter 5, in which we examine the flowfield generated by a MEMS-like actuator modeled as a deformed surface and explore the bearing of the physical parameters of an actuator array — i.e. shape, motion, and relative phase — on drag reduction in open loop control. IBC-modeled actuators were also incorporated into closed loop control schemes using information from wall sensors and more detailed flow information. These first attempts at control with discrete actuators yielded drag reductions of 1% to 14%, generally competitive only with the most successful passive control schemes even without taking into account the cost of mechanical losses. Nevertheless, with additional investigation of actuator geometries and control schemes, the possibility of more dramatic drag reduction is promising.

Thus, the work presented here addresses both prongs of the contemporary campaign to employ computers as a first line of attack in designing engineering systems and to advance the ability of computers to efficiently simulate more realistic systems. The Immersed Boundary Condition has been successfully used to both augment existing numerical methods to inexpensively solve more complex problems and as an instrument in a foray on solving flow control problems with discrete actuators.
Bibliography


