A STRATIFIED TURBULENCE FORMULATION AND A TURBULENT INFLOW BOUNDARY CONDITION FOR LARGE-EDDY SIMULATION OF COMPLEX TERRAIN WINDS

by

Clancy Umphrey

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Mechanical Engineering

Boise State University

August 2015

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Clancy Umphrey

Thesis Title: A Stratified Turbulence Formulation and a Turbulent Inflow Boundary Condition for Large-Eddy Simulation of Complex Terrain Winds

Date of Final Oral Examination: 15 June 2015

The following individuals read and discussed the thesis submitted by student Clancy Umphrey, and they evaluated his presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Inanc Senocak, Ph.D.	Chair, Supervisory Committee
Ralph S. Budwig, Ph.D.	Member, Supervisory Committee
Trevor Lujan, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Inanc Senocak, Ph.D., Chair of the Supervisory Committee. The thesis was approved for the Graduate College by John R. Pelton, Ph.D., Dean of the Graduate College.

dedicated to my family

ACKNOWLEDGMENTS

There are many here at Boise State University (BSU) that deserve my thanks for their support during my studies. Notably, I thank Dr. Inanc Senocak for giving me the opportunity to work on the research of this thesis within the High Performance Simulation Laboratory for Thermo-Fluids (HiPerSimLab) and the tireless hours he put into advising me. The unique opportunities I have had in the HiPerSimLab have made it possible to get a career start that would not have been possible otherwise. I also thank Dr. Ralph Budwig and Dr. Trevor Lujan for serving on my thesis committee and providing me with helpful feedback as well.

I thank fellow members of the HiPerSimLab, Rey DeLeon and Micah Sandusky, for their support in my research—Rey for getting me up to speed on software development within the flow solver this work is based on, and Micah for assisting me with pre-processing for my simulations. Thanks to Jason Cook who has been an outstanding system administrator for the Kestrel HPC Cluster I have relied on here at BSU.

The Idaho Space Grant Consortium (ISGC) also deserves my thanks for awarding me a NASA EPSCoR fellowship that funded my research in part. This material is also based upon work supported by the National Science Foundation under Grant No. 1056110 and 1229709.

ABSTRACT

There has been an increased interest to forecast winds over complex terrain under realistic stability conditions using spatial resolutions that are much finer than the current practice. This goal is realizable thanks to the computational power of graphics processing units (GPUs). This thesis investigates an immersed boundary (IB) formulation and a turbulent inflow boundary condition within a multi-GPU parallel incompressible wind solver. Katabatic flows over a sloped complex terrain surface under stable stratification remain to be one of the least understood subjects in atmospheric turbulence. Prandtl's analytical solution for laminar katabatic flow is used to develop an IB formulation to impose heat flux boundary conditions, and to assess the formal accuracy of the proposed IB schemes. Direct numerical simulation of turbulent katabatic flow is then performed to investigate the applicability of proposed schemes in the turbulent regime. Additionally, a turbulent inflow boundary condition formulation based on perturbations to the buoyancy field is also developed and studied for a channel flow. Results show that a statistically neutral buoyancy field can serve as a practical method to generate turbulent inflow conditions, and turbulent katabatic flow simulations are sensitive to the specifics of the IB formulation. With these two contributions, the current flow solver is closer to simulating winds over thermally active complex terrain.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

DOE – Department of Energy

IB – Immersed Boundary

LES – Large-eddy Simulation

DNS – Direct Numerical Simulation

RE – Reynolds

RANS – Reynolds-averaged Navier-Stokes

GPU – Graphics Processing Unit

CPU – Central Processing Unit

CUDA – Compute Unified Device Architecture

MPI – Message Passing Interface

SGS – Subgrid Scale

ABL – Atmospheric Boundary Layer

WRF – Weather Research and Forecasting Model

CHAPTER 1

INTRODUCTION

Complex terrain covers 70% of the Earth's land surface [11] and its meteorological influence subsequently affects many aspects of human activity and interest. Research on wind forecasting in mountainous terrain is significant to a long list of applications such as weather prediction, air pollution and contaminant dispersion, aviation, mountain warfare, UAV mission planning, snow pack prediction for drought planning, flash flood prediction, forest fires, and wind energy; however, it is one of the overall least understood aspects of atmospheric sciences. The physics of wind prediction embodies fundamental concepts from fluid mechanics, heat transfer, turbulence, and thermodynamics. Additionally, forecasting winds using supercomputers requires efficient adoption of techniques from computational mathematics and parallel computing.

Research on wind forecasting in complex terrain is very active and continues to interest organizations such as the U.S. Department of Energy (DOE) who recently funded a multi-institution project [51] entitled "Wind Forecasting Improvement Project in Complex Terrain." It is within the mission of the DOE to see clean energy technologies, like wind power, efficiently integrated into the power grid [52]—which remains a challenge since wind power is a variable generation resource and electrical loads on the grid must be instantaneously balanced to ensure resilient operation. In order to balance electrical loads while incorporating wind energy, balancing au-

thorities must keep other energy resources, such as fossil fuel plants, on standby to compensate for a shortfall in wind power, or curtail wind power when it exceeds the forecast, wasting energy in both scenarios. Multiple studies conducted around the 2008-2011 time-frame have shown an annual projected savings in the United States (U.S.) of $1.6–$4.1 billion per year from a perfect forecast compared to the state-of-the-art wind forecasting ability [28]. These savings were based on wind energy supplying 20% of the nation's electricity demand as projected by the DOE for the 2030–2040 time-frame (4.5% was the account in 2013) [52]. The greatest relative benefit came from the first 10%–20% improvement in wind forecasts, with diminishing marginal benefits in further approaching the perfect forecast. These studies show that even incremental improvements in wind forecasting will bring significant cost savings and grid reliability for the nation's energy future.

In mountainous terrain, the wind is not only a result of the general air-mass movement (synoptic wind), but is often heavily influenced by anabatic (upslope) and katabatic (downslope) winds as well. Figure 1.1 illustrates these flows. Anabatic winds are caused by solar heating of a slope, which in turn heats the air near its surface. This surface heat flux reduces the density of the near-surface air relative to air further from the slope at the same elevation. Buoyancy drives the near-surface air up-hill where it will eventually deviate from the surface and continue rising thousands of feet into the atmosphere creating complex weather patterns in mountainous regions. Katabatic winds are caused by the opposite effect of surface cooling, typically at night, and result in downhill winds. Anabatic and katabatic winds are known to play a significant role in mountain meteorology. Upslope anabatic winds are accompanied by upvalley winds during the day, and downslope katabatic winds are accompanied by downvalley winds during the night. The valley winds are a result of temperature

variation between the valley and nearby areas causing horizontal pressure gradients, and by buoyancy forces along the valley. It is not uncommon for these thermally driven flows to overpower synoptic flow and be the primary contributor to wind within 100 to 6,000 ft of the surface [11]. These winds can also significantly affect meteorology beyond these heights. For example, anabatic winds can contribute to deep convection events that result in very damaging hail storms [5].

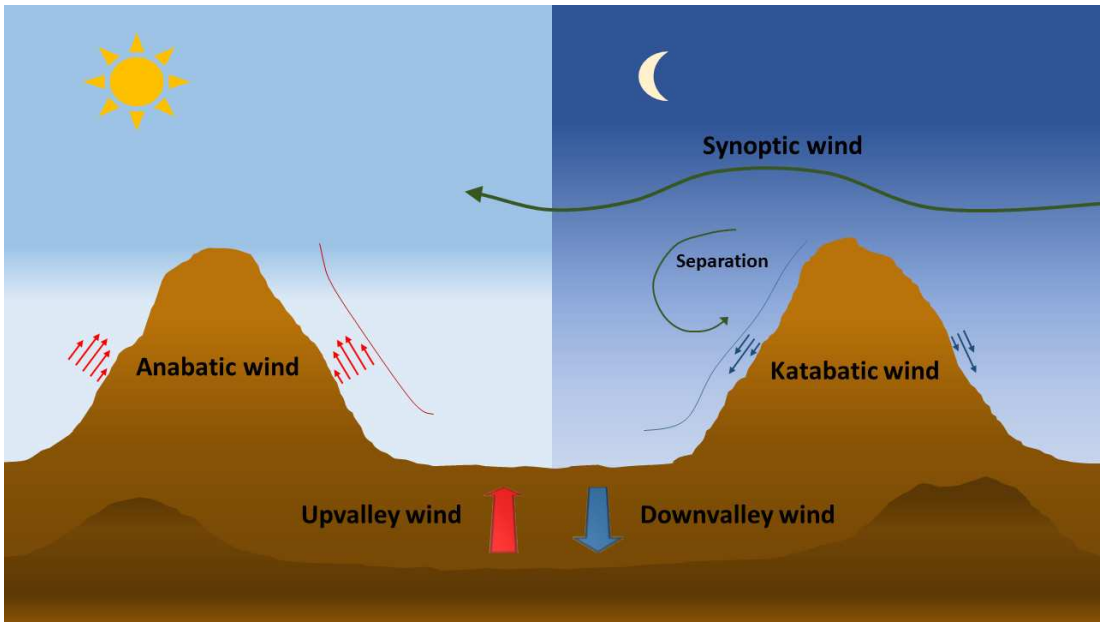


Figure 1.1: Some basic winds in complex terrain with thermally active surfaces. Daytime conditions are on the left and nighttime conditions are on the right.

Compared to anabatic winds, our scientific understanding of katabatic winds is less complete. The physics of katabatic flows is uniquely challenging, in which low-level jets form near the surface and negative buoyancy acts to suppress vertical turbulent exchange [10]. Well known katabatic winds include the mistral in southern France, which flows from the snow-capped Alps down into the Rhone River Valley, and the katabatic flow along the edge of the Greenland and Antarctic ice sheets, which frequently reaches 62 *mph* [32]. A fundamental understanding of thermally driven

flows and improved computer models are needed because wind turbines fall within the region of the atmospheric boundary layer where such interesting physics take place. Without an accurate representation of thermally active surfaces in simulations, the significant influence of these winds is absent from a model of flow over complex terrain.

Theoretical development of complex terrain flows had its start as early as 1948 when Queney published a paper on inviscid flow over hills and mountains [42, 54]. In modern times, the advent of massively parallel computing has made it possible to simulate and numerically investigate important non-linear physical processes in complex terrain flows, such as turbulence and flow separation. To date, most research has primarily focused on simulating neutrally stable flow over hills, which does not include buoyancy and thermal effects. While simulating turbulent neutrally stable flow remains challenging by itself, realistic wind simulation must include thermal effects at the surface if we are to use wind forecasting in a predictive capacity for wind energy grid integration.

1.1 Thesis Statement

This thesis will focus on developing two components of a comprehensive complex terrain wind forecasting engine designed for parallel execution on clusters of graphics processing units (GPUs). The first component of this thesis is the numerical formulation of a thermally active surface where terrain is represented with the Cartesian immersed boundary (IB) method. The second component of this thesis is the formulation of a turbulent inflow boundary condition for large-eddy simulation (LES) of winds. The IB method will be validated for fundamental katabatic flows in the laminar and turbulent flow regimes using direct numerical simulations (DNS), and

the turbulent inflow boundary condition will be validated against DNS of turbulent channel flow.

Mesh generation for simulations of wind in complex terrain must be computationally efficient to be practical and able to produce solutions at forecasting speeds. A modern approach that is both automated and efficient is a structured Cartesian mesh employing the IB method to represent complex geometry [44]. Rather than fit a mesh to a geometry surface, the IB method immerses the geometry in a Cartesian mesh, identifies nodes near the geometry surface, and then applies a reconstruction scheme to the near surface nodes that implicitly represents the effect of the surface boundary conditions on the fluid flow. Cartesian meshes are computationally more efficient than their unstructured counterparts, as they fit well to the computer architecture of modern graphics processing units. Most of the research accomplished to date has focused on reconstructing boundary conditions for the momentum field. This thesis will develop a reconstruction scheme to represent flux boundary conditions for the energy field.

Most of the practical flows relevant to engineering applications and atmospheric flows, in general, are high Reynolds (RE) number phenomena. These flows are inherently turbulent. Because direct numerical simulation of turbulent flows, where the smallest eddies need to be resolved in space and time, are prohibitively expensive and beyond our reach, turbulence has to be parameterized in numerical simulation of high-RE flows. Turbulence modeling has been the subject of intense research since the sixties. Early efforts focused on the mean quantities, which led to several Reynolds-averaged Navier-Stokes (RANS) solvers. For instance, the k-ϵ model has been a popular choice for most internal flows and the Spalart-Allmaras model has been popular for external aerodynamics [53]. The large-eddy simulation technique

gained traction in the mid eighties, with increased popularity of computers, and became applicable to a broader range of geometries with the advent of the dynamic procedure [15]. However, the LES technique is uniquely different than the RANS approach because a reliable LES simulation resolves energetic eddies of turbulent flows and the statistics of those resolved motions must obey the statistical theory of turbulence. Turbulent inflow boundary conditions for LES have been an active area of research, since providing mean quantities with random fluctuations at an inflow boundary is not sufficient to trip turbulence and lead to the so-called energy cascade where larger eddies break into smaller eddies until they are dissipated by action of viscosity [47].

Several methods have been explored for enabling turbulent inflow. A popular method is to prescribe synthetic turbulence at the inlet based on expected turbulence statistics. While this method sees much greater success than simply prescribing random fluctuations at an inlet, it only approximates certain aspects of turbulent flow, requires turbulence information that is not readily available for complex terrain wind, and can require long fetches to break down into realistic turbulence [24]. A method that perturbs the buoyancy field near the inlet in order to trip the natural evolution of turbulence within a short fetch [34, 36] will be developed and investigated in this thesis.

A challenge in weather modeling is the large-scale computing necessary to resolve microscale effects. However, it is expected that in the 2018–2020 timeframe, exascale computing will be available [14] that is capable of running a high-resolution model. The exascale is likely to be accomplished through the use of GPUs as accelerators. Currently, the worlds 2nd fastest supercomputer (Titan) utilizes GPUs for many-core computing [50]. Due to a recent rapid increase in GPU programmability and

capability, GPUs have the advantage over conventional processors in their ability to achieve massive parallelism to solve complex problems [38].

The use of GPUs for many-core computing began in 2007 and there is still a lack of simulation science software that can efficiently utilize these new supercomputing tools. Since 2007, Boise State researchers have been developing a multi-GPU parallel incompressible Navier-Stokes solver named GIN3D. GIN3D has seen as much as 15-20x speedup factors compared to conventional processors [23, 49]. This thesis will build upon the GIN3D effort and utilize GPUs as the next generation computing platform. Wind simulations on GPU clusters will be a convincing demonstration of how best to use future supercomputers.

Kestrel, a 32-node CPU/GPU high performance computing cluster [2], was the primary computing resource used for this thesis. Kestrel was acquired through a National Science Foundation Major Research instrumentation grant (Award # 1229709). Each node contains 2 Intel Xeon E5-2600 series processors (16 cores/32 threads) and 2 NVIDIA GPUs (44 Tesla K20 and 10 Quadro K5200 total). The system uses a Mellanox ConnectX-3FDR Infiniband interconnect, has a total of 2 TB of RAM with 64 TB Panasas Parallel File Storage, and uses PBSPro for complex job scheduling. Figure 1.2 shows Kestrel in its machine room on-site at Boise State University. Kestrel also powers a visualization wall of 40 connected 30-inch monitors with 100 megapixels of clarity for studying high resolution images and simulation results, as seen in Fig. 1.3.

Figure 1.2: Kestrel high performance CPU/GPU computing cluster, named after North America's smallest falcon on account of its modest size.

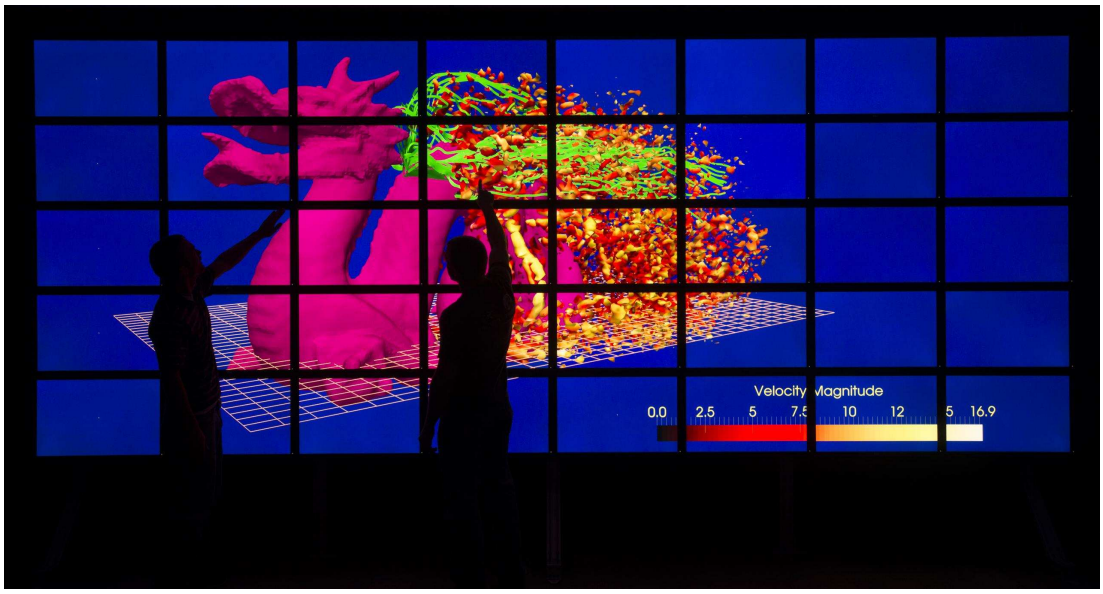


Figure 1.3: Kestrel Viz-Wall used to visualize high resolution images and simulation data.

CHAPTER 2

TECHNICAL BACKGROUND

This chapter includes governing equations and numerical methods used in the flow solver. To enable forecasting capabilities in the future, computations are performed on clusters of GPUs using a multi-GPU parallel incompressible flow solver. An overview of the computational performance from GPUs is presented.

2.1 Governing Equations

Flows with a Mach number less than 0.3 are typically treated as incompressible in engineering practice [25]. This condition is met by the wind flows of interest here; therefore, the incompressibility assumption is made. The governing equations for LES of incompressible flows are the Navier-Stokes equations in filtered form given as,

$$\frac{\partial \overline{u}_j}{\partial x_j} = 0 \tag{2.1}$$

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\overline{u}_i \overline{u}_j\right) = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(2\nu \overline{S}_{ij} - \tau_{ij}\right) + F_i, \tag{2.2}$$

where

$$S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) \tag{2.3}$$

is the deformation tensor, and

$$\tau_{ij} = \overline{u_i u_j} - \overline{u}_i \overline{u}_j \tag{2.4}$$

is the tensor representing the interaction of the subgrid-scales on the resolved large-scales. The overbar represents a filtered quantity, with the filter width typically provided by the numerical mesh as $\Delta = \sqrt[3]{dx \cdot dy \cdot dz}$ [6]. F_i is a source, or body force, that can be used to include bouyancy effects by the Boussinesq approximation for incompressible flows with small temperature variation [12, 25]:

$$F_i = g_i \rho_\infty \beta \left(T - T_\infty\right), \tag{2.5}$$

where g_i is the gravitational acceleration, ρ_∞ is the reference density, β is the thermal expansion coefficient, T is the local temperature, and T_∞ is the reference temperature. The Boussinesq approximation is commonly used in wind modeling, and if temperature differences are below 15^oC, errors are of the order of 1% for air [12].

The temperature equation can be written as [20, 48]

$$\frac{\partial T}{\partial t} + \frac{\partial}{\partial x_j}\left(T\overline{u}_j\right) = \frac{\partial}{\partial x_j}\left(\gamma \frac{\partial T}{\partial x_j}\right) + \Phi, \tag{2.6}$$

where γ is the thermal diffusivity and Φ is a source term. For convenience, bouyancy, b, can be solved for in place of T. Solving for buoyancy simply requires the substitution of b for T in Eq. 2.6 and modification of F_i and Φ. The Buoyancy form of the energy equation is presented in Chapter 3.

2.2 Numerical Methods

The governing equations were solved on a directionally-uniform Cartesian grid using the projection algorithm [4]. The second-order central difference scheme was used for spatial derivatives and time advancement was performed with the second-order Adams-Bashforth scheme. The pressure Poisson equation was solved by a geometric, three-dimensional multigrid method with a weighted Jacobi solver [21]. Large-eddy simulation was performed using a localized dynamic Lagrangian model for subgrid-scale turbulence modeling [8, 15].

2.3 GPU Computing

High performance scientific computing has typically been done on clusters of thousands of central processing units (CPUs) working in parallel to solve a problem; however, within a single modern GPU, thousands of cores are available to perform massively-parallel computations. While the original purpose of the GPU was for rendering computer graphics, the advent of NVIDIA's Compute Unified Device Architecture (CUDA) in 2007 made GPUs more accessible to scientists for accelerating scientific numerical algorithms [43]. Through the Message Passing Interface (MPI), clusters of GPUs can be enabled to work in parallel.

The work of this thesis builds upon the multi-GPU parallel wind solver GIN3D [8, 21, 22, 49] written in the C-based CUDA and MPI programming languages. MPI is used to partition the data into large sections, and CUDA uses GPUs to execute parallel instructions on individual data elements. In work by previous developers, a GPU and CPU version of GIN3D was created for taking a direct measure of

the performance of GPUs against CPUs. In these comparisons, GPUs had 15–20x speedup factors over CPUs [22, 49].

As GIN3D became more advanced, the CPU version was abandoned in order to focus on the GPU version, making direct comparison no longer possible. However, GIN3D compute-performance has continued to be gauged through code-to-code comparison with OpenFOAM, a CPU-based community model [37]. Figure 2.1 compares GIN3D performance on 2, 4, and 8 GPUs against OpenFOAM on 128, 256, and 512 CPU processors. The GPU cluster used for the simulations has two Intel Xeon E5-2670 Sandy Bridge 2.60 GHz Eight Core processors with two Tesla K20 GPU cards per compute-node. Nodes are connected with and Infiniband FDR (56 Gbps) switch. Simulation was of the well-known lid-driven cavity problem on a 256^3 mesh with GIN3D using 0.6 the time step size of OpenFOAM. Speedup results are relative to OpenFoam performance on 128 processors. For this case, the overall best performance was on four GPUs (becuase the decomposition of the domain results in a better occupancy on each GPU) with a speedup of 17$\times$ over OpenFOAM on 128 processors. Two GPUs, which occupy one node, are even faster than OpenFOAM performance on 512 processors, which occupy 32 nodes.

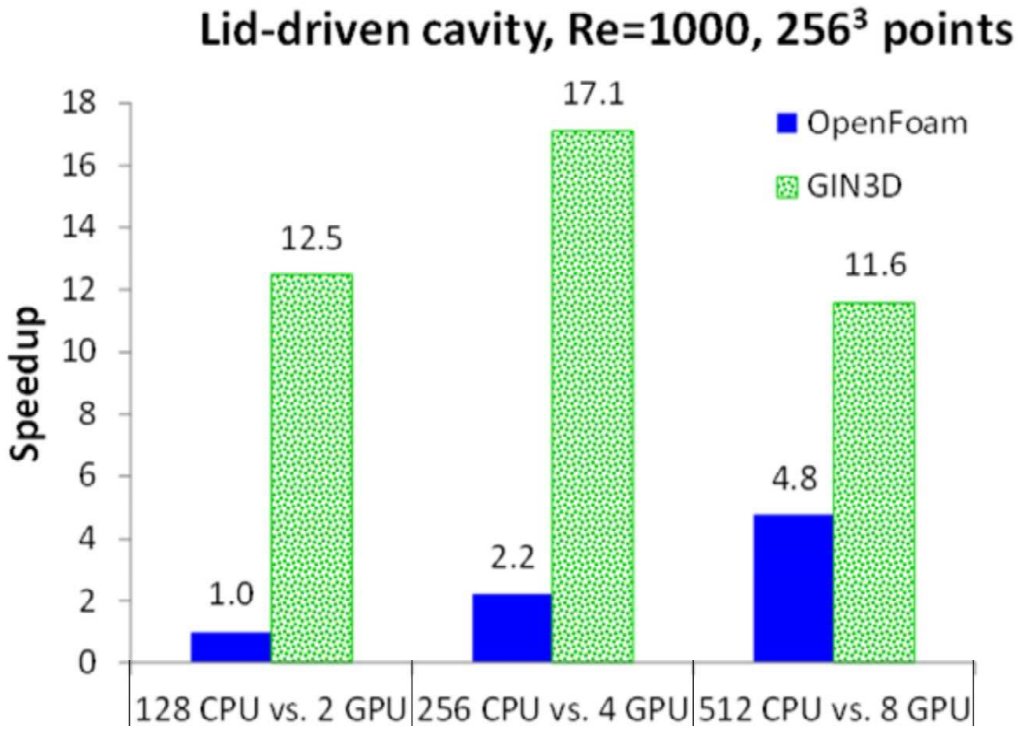


Figure 2.1: GIN3D performance over OpenFOAM for the lid-driven cavity problem.

CHAPTER 3

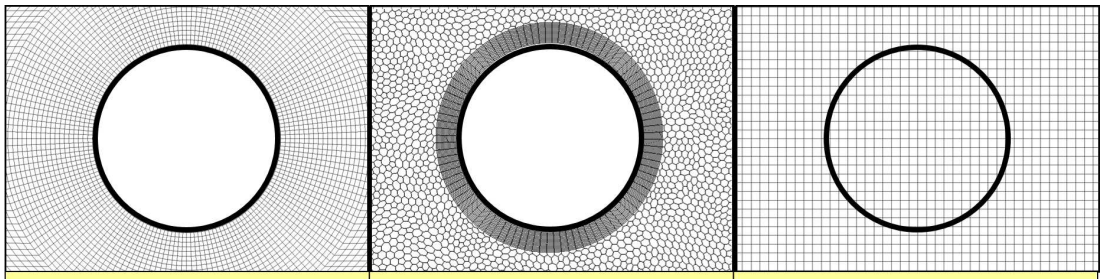
IMMERSED BOUNDARY METHOD

Mesh generation is an integral component of any CFD analysis. Depending on the flow application, different meshing strategies are needed to obtain an accurate simulation. For instance, in aerodynamic applications, such as flow over an airfoil, it is essential to create a mesh that resolves the boundary layer consistent with the vertical resolution requirement of the turbulence model at hand. It is not uncommon for a CFD engineer to spend days to weeks creating a quality mesh driven by high accuracy expectations from the CFD simulation. In aerospace design, error margins are typically very low, justifying the time spent to create a quality mesh. However, the situation is different for atmospheric flows. Field measurements often report data with large uncertainty. Incomplete initial conditions and surface parameterizations are the main source of error in atmospheric simulations. Additionally, the surface is always aerodynamically rough, making it irrelevant to resolve a viscous boundary layer in the aerodynamic sense. Therefore, atmospheric flow simulations are mainly driven by the goal of obtaining a good quality forecast. Unlike engineering CFD applications where a dedicated mesh generation software is used, in atmospheric flow simulation codes, mesh generation is embedded within the flow solver. Either single-block terrain following coordinates or an unstructured mesh are used to mesh the domain. In the case of complex terrain, these approaches are prone to introducing errors due to

skewness of individual cells.

The immersed boundary (IB) method has been proposed as an alternative meshing approach for complex terrain wind simulations [44]. The origins of the IB method go back to the work of Peskin in 1972 when he simulated the elastic motion of a heart [39]. Figure 3.1 depicts the essence of the IB method relative to other meshing strategies that are commonly adopted in engineering CFD applications. Structured meshes typically give faster CFD solution speeds because they better align with the structure of computer architecture. Classical structured body-fitted (or surface-conforming) meshes are computationally efficient, but require a manual procedure that involves significant expertise and can take days to weeks for experienced users to complete. Unstructured meshes can be generated automatically, but are not computationally efficient and it can be difficult to avoid skewed cells that introduce errors into the simulation, as mentioned previously. A modern approach that can be both automated and efficient is a structured Cartesian mesh employing the immersed boundary method to represent complex geometry. Rather than fit a mesh to a geometry surface, the IB method immerses the geometry in a Cartesian mesh, identifies nodes near the geometry surface, and then applies a scheme to the near surface nodes that will represent the effect of the geometry on the fluid flow. The IB method is independent of geometry complexity, and structured Cartesian meshes are computationally efficient, especially on GPUs. The IB method is an active area in CFD research with a variety of implementation techniques under exploration.

The core idea of the IB method is to introduce a body force term, F_i, in the momentum equations to enforce the boundary conditions on the flow field, seen in the discretized u-component of the momentum equations as



Structured Body-fitted	Unstructured Body-fitted	Cartesian Immersed Boundary
Slow mesh generation	Fast mesh generation	Fast mesh generation
Fast solution	Slow solution	Fast solution

Figure 3.1: Depictions of meshing schemes with comments on mesh generation speed and CFD solution speed below each scheme.

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = RHS_i + F_i, \tag{3.1}$$

where RHS_i includes convective, diffusive, and pressure gradient terms. Unlike body-fitted meshes, computational grid nodes in the IB method do not align with the exact location of the boundary. The issue of finding a suitable body force has been a daunting task because of numerical stability issues. In 1997, Mohd-Yusof [31] proposed a direct-forcing approach that alleviated stability constraints, which led to a surge in the application of the IB method to fluid flow problems [30]. In the direct-forcing method, velocity at the boundary can be prescribed as $u_i^{n+1} = V_i^{n+1}$, leading to body force

$$F_i = -RHS_i + \frac{V_i^{n+1} - u_i^{n+1}}{\Delta t}, \tag{3.2}$$

from Eq. 3.1. Substituting Eq. 3.2 into Eq. 3.1, the body force can be taken into account implicitly by prescribing the velocity field, V_i^{n+1}. Since complex geometry boundaries are not coincident with the computational grid nodes, reconstruction schemes are required to impose the proper velocity and other boundary conditions

on nodes near the solid geometry surface.

Previously in GIN3D, an IB method using discrete forcing with indirect imposition of the boundary conditions was implemented [7, 45]. Discrete forcing refers to imposing IB forcing terms after the governing equations are discretized, rather than calculating additional continuous terms in the governing equations. Indirect boundary imposition refers to reconstructing variables at grid nodes in the fluid domain near the solid boundary, rather than directly imposing the boundary conditions in the near boundary computational grid to create a "sharp" interface.

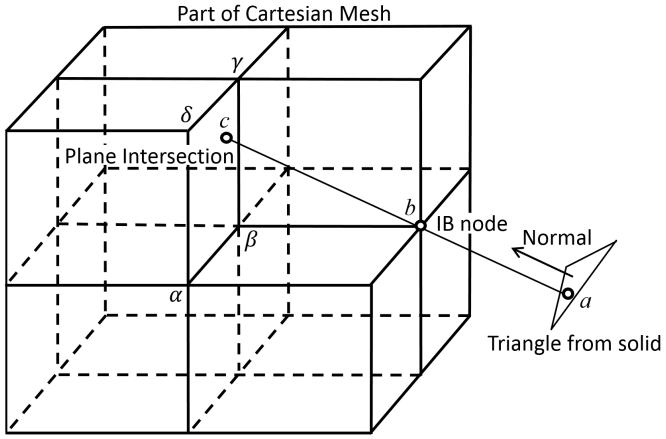


Figure 3.2: Sketch of the general indirect boundary reconstruction scheme at an IB node by projecting a line in the normal direction from the nearest triangular element of the boundary into the fluid domain.

The velocity reconstruction scheme followed the approach in Gilmanov et al. [17–19]. Figure 3.2 shows the implemented general indirect boundary reconstruction scheme that projects a line in the normal direction from the nearest triangular element of the immersed boundary (point a), through the IB node (point b), and onto a Cartesian cell face in the fluid domain (point c). This line will be referred to here as the IB line. Values at a are known as prescribed boundary conditions and values at c are reconstructed by linear interpolation from the neighboring Cartesian

grid nodes marked by Greek letters α, β, γ, and δ. In the case of linear velocity reconstruction, another linear interpolation along the IB line between a and c is performed to overwrite the current value at b.

3.1 Buoyancy Reconstruction Schemes

Much of the work done with IB methods has focused on reconstruction of the velocity field and application to low and moderate Reynolds number flows. IB treatment of boundary conditions for heat transfer has received less coverage despite their significance in many engineering flows. Heat transfer boundary conditions can be either Dirichlet (prescribed temperature) or Neumann (prescribed heat flux) type. Reconstruction schemes developed for velocity boundary conditions can be applied to impose a prescribed temperature boundary condition. In this work, focus is on the IB treatment of Neumann boundary conditions for the temperature field. The velocity reconstruction introduced in the beginning of Chapter 3 is of the IB type that reconstructs flow variables at nodes in the fluid domain, as in Fadlun et al. [9], Gilmanov et al. [19], Balaras [1], and Choi et al. [3]. Also common is the method of reconstructing flow variables at nodes on the solid side of the immersed surface instead, as in Gao et al. [13], Ghias et al. [16], Mittal et al. [29], and Lundquist et al. [27]. This thesis reconstructs flow variables at nodes in the fluid domain. Five reconstruction schemes are considered and the buoyancy quantity is solved for instead of temperature, for convenience.

Scheme 1 uses the buoyancy gradient boundary condition at the surface and computed buoyancy gradient in the fluid with a central difference to calculate the buoyancy at the IB node, following Gilmanov et al. [19], as outlined in the following

steps for a general scalar ϕ:

1. Calculate $\left(\frac{\partial \phi}{\partial x_i}\right)$ at α-β-γ-δ in all three Cartesian coordinate directions ($i = 1, 2, 3$).

2. Linear interpolate these to c to get $\left(\frac{\partial \phi}{\partial x_i}\right)_c$.

3. Then the normal gradient of ϕ at c is

$$\left(\frac{\partial \phi}{\partial n}\right)_c = \mathbf{n}_a \cdot (\nabla \phi)_c \,, \tag{3.3}$$

   where $\mathbf{n}_a$ is the surface normal vector at a.

4. With the normal ϕ gradient boundary condition at the surface, $\left(\frac{\partial \phi}{\partial n}\right)_a$, and the known normal ϕ gradient at c, $\left(\frac{\partial \phi}{\partial n}\right)_c$, linear interpolate along the IB line to the midpoint between b and c to determine $\left(\frac{\partial \phi}{\partial n}\right)_{bc}$.

5. Linear interpolate to bring ϕ from α-β-γ-δ to c, giving ϕ_c.

6. The value of ϕ at the IB node is then calculated using a central difference approximation

$$\phi_b = \phi_c - \Delta s_{bc} \left(\frac{\partial \phi}{\partial n}\right)_{bc} , \tag{3.4}$$

   where Δs_{bc} is the distance along the IB line from b to c.

In step 1, Gilmanov et al. did not propose a method for calcualting the first derivatives, so they are assumed to be approximated with a second order accurate central difference, shown here in 1D for uniform mesh spacing Δz:

$$\left(\frac{\partial \phi}{\partial z}\right)_k = \frac{\phi_{k+1} - \phi_{k-1}}{2\Delta z}, \tag{3.5}$$

where k represents grid index. Scheme 1 was proposed by Gilmanov et al. for reconstruction of pressure—it was not tested carefully for other scalars using analytical solutions.

The remaining Schemes 2 through 5 are unique to this thesis, as far as the author is aware. *Scheme 2* assumes the ϕ gradient along the IB line from a to c is constant. Analogous to Eq. 3.4, a central difference using the ϕ gradient boundary condition is used to calculate the buoyancy at the IB node,

$$\phi_b = \phi_c - \Delta s_{bc} \left(\frac{\partial \phi}{\partial n}\right)_a. \tag{3.6}$$

Scheme 3 is identical to Scheme 1, except first derivatives in Step 1 are approximated with a one-sided difference when an IB node would be included in the regular central difference stencil. Thus, the use of IB node values in the reconstruction is avoided. The second order accurate one-sided difference used is shown here in 1D for uniform mesh spacing Δz [48]:

$$\left(\frac{\partial \phi}{\partial z}\right)_k = \frac{-3\phi_k + 4\phi_{k+1} - \phi_{k+2}}{2\Delta z}. \tag{3.7}$$

Scheme 4 assumes the ϕ gradient along the IB line from a to b is constant and moves this gradient to b. To calculate ϕ at the IB node, a second order accurate one-sided difference is used that includes the ϕ gradient at b and two cell face intersection points from the fluid. This reconstruction requires an additional fluid point not shown in Figure 3.2, referred to here as d, where the IB line intersects the next cell

face after point c. Values of ϕ are brought to d through linear interpolation from neighboring Cartesian grid points in the same way values are brought to c through linear interpolation from α, β, γ, and δ. The reconstruction is performed by solving for ϕ_b in the following equation from Ferziger and Perić [12]:

$$\left(\frac{\partial\phi}{\partial n}\right)_b = \frac{-\phi_d\left(h_c-h_b\right)^2+\phi_c\left(h_d-h_b\right)^2-\phi_b[\left(h_d-h_b\right)^2-\left(h_c-h_b\right)^2]}{\left(h_c-h_b\right)\left(h_d-h_b\right)\left(h_d-h_c\right)}, \tag{3.8}$$

where h refers to the distance from the surface of each respective point b, c, and d. Equation 3.8 is the version of Eq. 3.7 for non-uniform grid spacing.

Scheme 5 is identical to Scheme 3, except in Step 4 the ϕ gradient is interpolated along the IB line to b, rather than the midpoint between b and c. The reconstruction of ϕ_b then follows Scheme 4 using Eq. 3.8.

3.2 Immersed Boundary Method Validation

Simulations of katabatic flow on an infinite plane inclined at slope angle α were used to validate the different immersed boundary reconstruction schemes for buoyancy. In 1942, Prantdtl published the one-dimensional model for laminar natural convection flow of a viscous stably-stratified fluid along a uniformly cooled or heated sloping plane [41]. Figure 3.3 illustrates the slope flow model of Prandtl for the case of a uniformly cooled sloping plane, leading to katabatic flow characterized by a low-level down-slope jet topped by weak up-slope return flow, with both velocity and bouyancy approaching zero far from the surface.

Fedorovich and Shapiro [10, 46] have published a non-dimensional form of the Prandtl model solution with the following formulation:

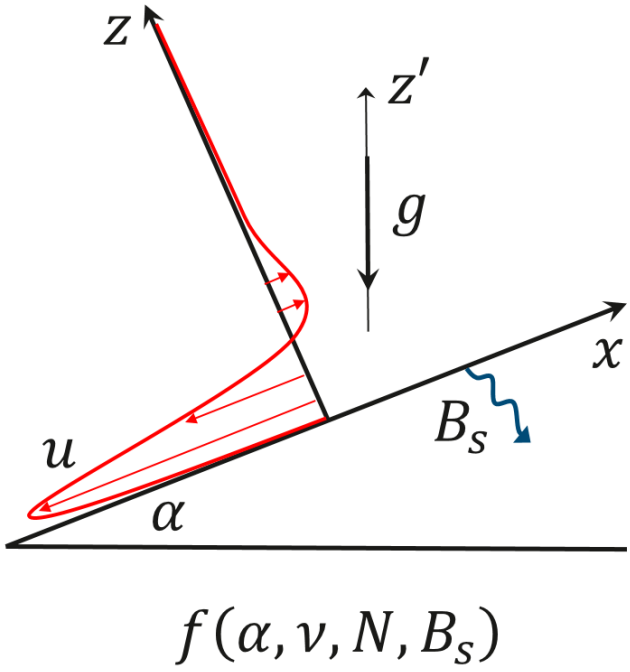


Figure 3.3: Sketch of the Prandtl model of laminar katabatic flow on an infinite plane with constant surface buoyancy flux.

$$u_n = sin(z_n/\sqrt{2})exp(-z_n/\sqrt{2}), \tag{3.9}$$

$$b_n = cos(z_n/\sqrt{2})exp(-z_n/\sqrt{2}), \tag{3.10}$$

where

$$z_n = z\nu^{-1/2}N^{1/2}sin^{1/2}\alpha, \tag{3.11}$$

$$u_n = u\nu^{1/2}N^{3/2}B_s^{-1}sin^{1/2}\alpha, \tag{3.12}$$

$$b_n = b\nu^{1/2}N^{3/2}B_s^{-1}sin^{1/2}\alpha. \tag{3.13}$$

In the above equations, subscript n refers to normalized quantities, u is velocity parallel to the slope, z is the normal distance from the slope surface, b is buoyancy ($b =$

$g\theta/\Theta_r$, g is the gravitational acceleration, θ is the potential temperature perturbation, $\Theta_r = cnst$ is the reference potential temperature value), ν is the kinematic viscosity equal to the thermal diffusivity, N is the $Brunt - V\ddot{a}is\ddot{a}l\ddot{a}$ (or buoyancy) frequency ($N^2 = (g/\Theta_r)\left(d\Theta_e/dz'\right)$, Θ_e is the environmental potential temperature), B_s is the surface buoyancy flux ($B_s = -\nu\left(db/dz\right)|_{z=0}$), and α is the slope angle. Furthermore, this analytical solution has the boundary conditions $u(0) = 0$, $\left(db_n/dz_n\right)|_{z_n=0} = -1$, and $u_n \to 0$, $b_n \to 0$ as $z_n \to \infty$.

Buoyancy effects can be included in the incompressible Navier-Stokes equations by the Boussinesq approximation with buoyancy forcing terms in the momentum equations. As shown by Fedoravich and Shapiro [10], the momentum balance equations for the slope flow case depicted in Fig. 3.3 have the following form:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z} = -\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + b\sin\alpha, \tag{3.14}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z} = -\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right), \tag{3.15}$$

$$\frac{\partial w}{\partial t} + u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z} = -\frac{\partial p}{\partial z} + \nu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) + b\cos\alpha, \tag{3.16}$$

with the heat balance given by

$$\frac{\partial b}{\partial t} + u\frac{\partial b}{\partial x} + v\frac{\partial b}{\partial y} + w\frac{\partial b}{\partial z} = \nu\left(\frac{\partial^2 b}{\partial x^2} + \frac{\partial^2 b}{\partial y^2} + \frac{\partial^2 b}{\partial z^2}\right) - N^2\left(u\sin\alpha + w\cos\alpha\right). \tag{3.17}$$

In the above equations, u, v, and w are velocity components in the up-slope, cross-slope, and slope-normal directions, respectively; p is pressure, $b\sin\alpha$ and $b\cos\alpha$ are buoyancy forcing terms, and the Prandtl number is assumed to be one, yielding equal momentum and temperature diffusivities.

3.2.1 Laminar Katabatic Flow Simulation

The formulation by Fedorovich and Shapiro of Prandtl's analytical solution for laminar katabatic flow was used to assess the formal accuracy of the different immersed boundary reconstruction schemes for buoyancy. Simulation settings were:

- Slope angle: $\alpha = 30^o$;
- Surface buoyancy flux: $B_s = -0.005\ m^2/s^3$;
- $Brunt - V\ddot{a}is\ddot{a}l\ddot{a}$ (buoyancy) frequency: $N = 1\ s^{-1}$;
- Kinematic viscosity: $\nu = 0.0005\ m^2/s$;
- Thermal diffusivity: $\gamma = 0.0005\ m^2/s$;
- Domain slope-normal length: $H = 1.27\ m$;
- Uniform Numerical grid spacing, Δ, of the grid convergence study: $0.005\ m, 0.0025\ m, 0.00125\ m$;
- Lateral boundary conditions: periodic;
- Lower boundary condition: no-slip for velocity, constant surface-flux for buoyancy;
- Upper boundary condition: free-slip for velocity, zero buoyancy.

IB schemes were tested by immersing a flat plate near the bottom of the Cartesian mesh such that none of the numerical grid nodes coincided with the surface. IB nodes were 0.25Δ above the flat plate.

Figure 3.4 shows comparison for the normalized down-slope velocity for Schemes 1–3 (results from Schemes 4 and 5 are shown for turbulent katabatic flow in Section

3.2.2 only). While Schemes 2 and 3 lie on top of the analytical solution, Scheme 1 as proposed by Gilmanov et al. [19] clearly falls short for the peak velocity of the low level jet. In Step 1 of Scheme 1, buoyancy gradient approximation in the slope-normal direction of the fluid domain is done using a central difference that includes the IB node, which is not ideal because the IB node value needs to be reconstructed. In other words, because of the extent of the central difference stencil, values in the fluid domain cannot be calculated independent of the IB node. Scheme 3 resolves this issue by using a second order accurate one-sided finite difference scheme that does not include the IB node. Central differences are still used in directions parallel to the slope. Figure 3.5 shows comparison for normalized buoyancy. Discrepancies in buoyancy are less apparent, but Scheme 1 shows low buoyancy values near the surface.

Formal investigation of the order of accuracy through a grid convergence study of a scheme reveals its suitability in different numerical simulation strategies. In large-eddy simulation, a minimum of second order accuracy globally is desirable. Furthermore, large deviations from the expected order of accuracy can indicate implementation errors. The order of accuracy of Schemes 2 and 3 was formally investigated in a grid convergence study shown in Figures 3.6 and 3.7, respectively. As expected in both schemes, first order accuracy was observed locally at the IB node by the L_∞ norm, and second order accuracy of the solution was observed globally by the L_1 and L_2 norms.

It should be noted that linear velocity reconstruction was used in Schemes 1 and 2, while parabolic velocity reconstruction was used in Scheme 3. Parabolic velocity reconstruction refers to calculating the velocity at the IB node from a solution to $u = A \cdot \Delta s^2 + B$, where A and B are solved for using the prescribed velocity at the surface and the computed velocity in the fluid. As will be shown in Section

3.2.2, parabolic velocity reconstruction gave more promising results with Scheme 2 in simulations of turbulent katabatic flow; afterwards, parabolic velocity reconstruction was studied with Scheme 3 in the laminar regime.

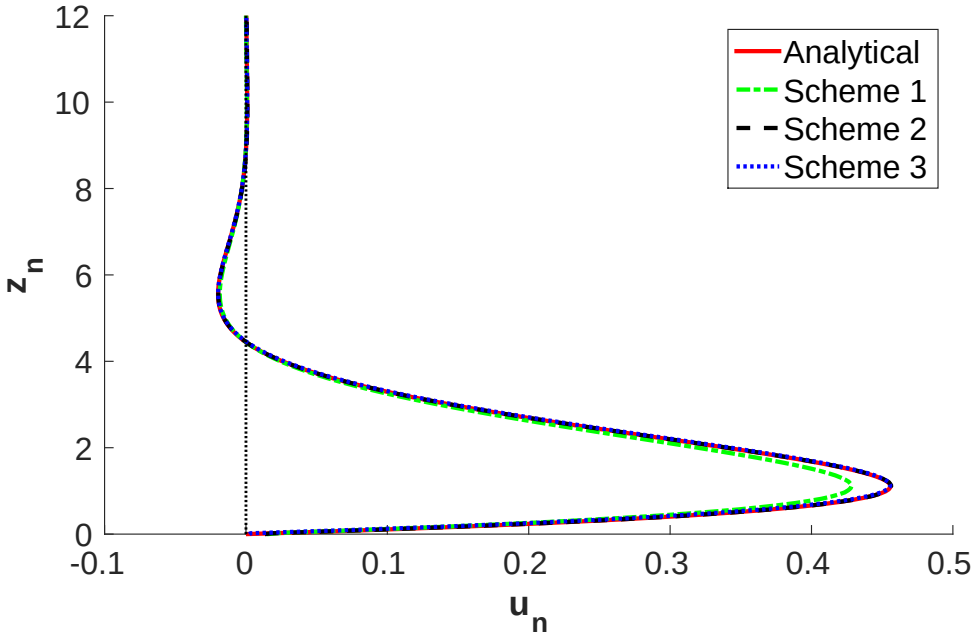


Figure 3.4: Comparison of normalized down-slope velocity from different buoyancy reconstruction schemes with the analytical solution of the Prandtl model for laminar katabatic flow.

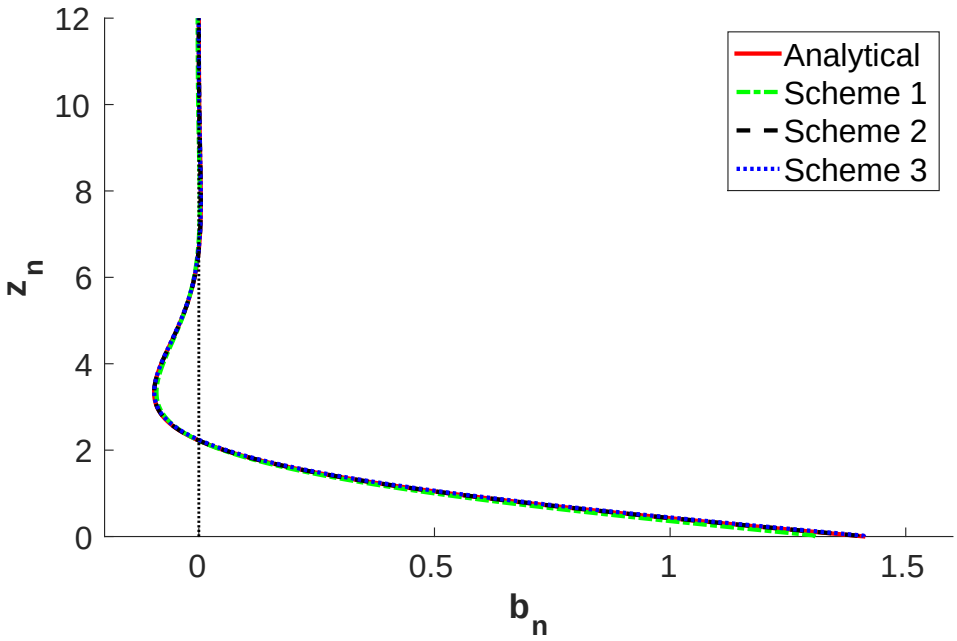


Figure 3.5: Comparison of normalized buoyancy from different buoyancy reconstruction schemes with the analytical solution of the Prandtl model for laminar katabatic flow.

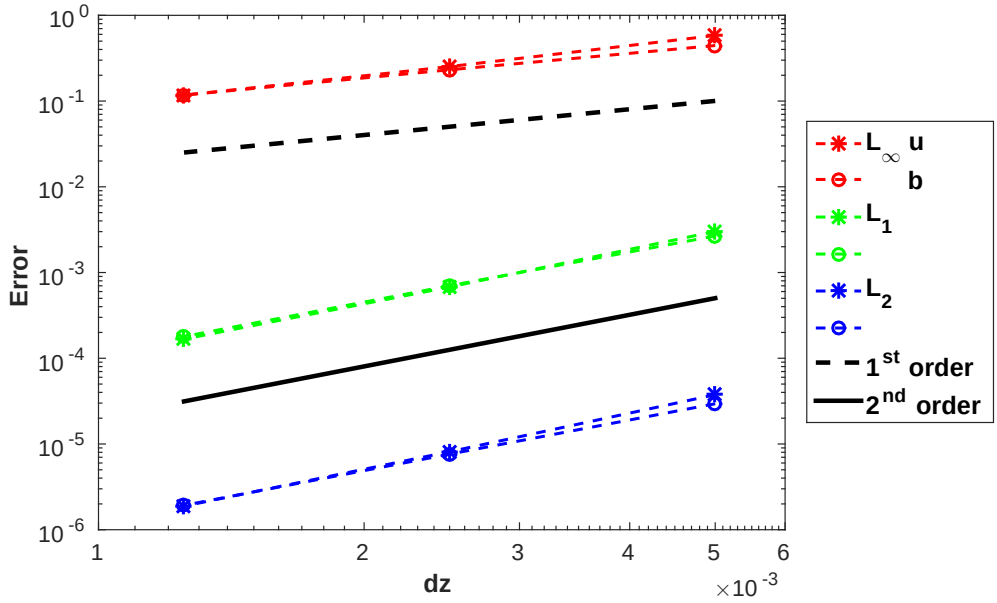


Figure 3.6: Grid convergence study of Scheme 2 with linear velocity reconstruction. The L_∞ norm shows first order accuracy locally at the IB node, and the L_1 and L_2 norms show second order accuracy globally.

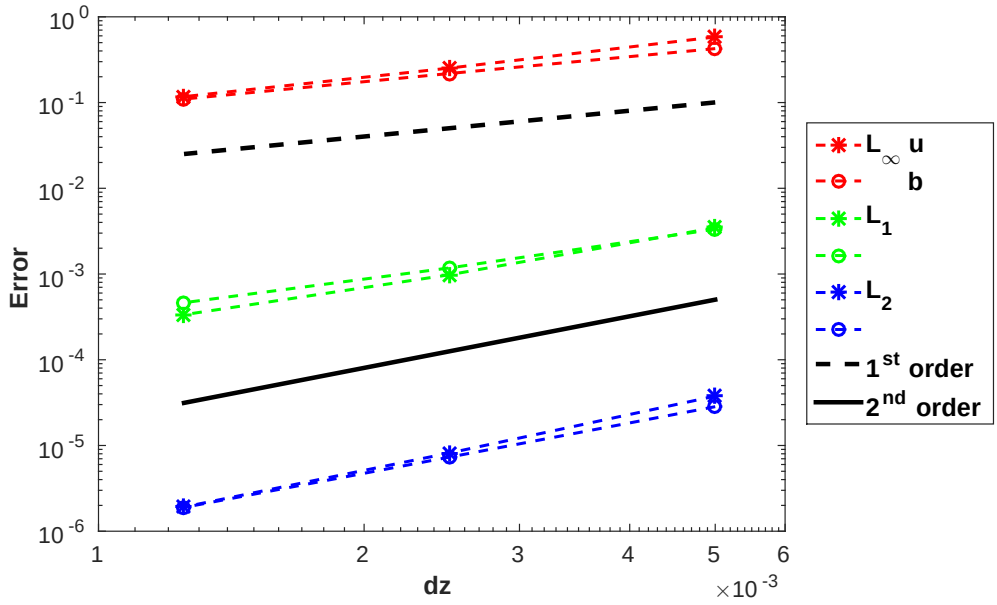


Figure 3.7: Grid convergence study of Scheme 3 with parabolic velocity reconstruction. The L_∞ norm shows first order accuracy locally at the IB node, and the L_1 and L_2 norms show second order accuracy globally.

3.2.2 Turbulent Katabatic Flow Simulation

Schemes that produce satisfactory results in the laminar regime may not readily extend to turbulent flow regimes. Because the first implementation of Scheme 1 failed to reproduce the analytical solution, it was not considered in the turbulent flow regime. Consequently, the applicability of Schemes 2–5 was considered in the turbulent regime by conducting direct numerical simulation (DNS) for the turbulent counterpart of the Prandtl model of katabatic flow. DNS of these flows has been studied extensively by Fedorovich and Shapiro [10], in which they derived an integral slope-flow Reynolds number as $Re_I = |B_s|/\nu N^2 \sin\alpha$. Values of Re_I greater than 3000 are considered to have reasonably developed turbulence. For reference, the laminar simulation in Section 3.2.1 had an $Re_I = 20$. Settings for the turbulent simulations conducted here were:

- Slope angle: $\alpha = 60^o$;
- Surface buoyancy flux: $B_s = -0.5\ m^2/s^3$;
- $Brunt - V\ddot{a}is\ddot{a}l\ddot{a}$ (buoyancy) frequency: $N = 1\ s^{-1}$;
- Kinematic viscosity: $\nu = 0.0001\ m^2/s$;
- Thermal diffusivity: $\gamma = 0.0001\ m^2/s$;
- Domain size: $(X \times Y \times Z\times) = 0.64\ m \times 0.64\ m \times 1.6\ m$;
- Numerical grid dimensions: $(NX \times NY \times NZ) = 257 \times 257 \times 641$;
- Uniform numerical grid spacing: $\Delta = 0.0025\ m$;
- Lateral boundary conditions: periodic;

- Lower boundary condition: no-slip for velocity, constant surface-flux for buoyancy;

- Upper boundary condition: free-slip for velocity, zero-flux buoyancy.

With these settings, $Re_I = 5773$. Given the analog of the Kolmogorov microscale $L_m = \nu^{3/4}|B_s|^{-1/4}$, the resolvability condition for DNS [40] is met by $\Delta \leq 2L_m$.

Figure 3.8 shows a volume rendering of the instantaneous velocity magnitude, and Fig. 3.9 shows a 2D slice of the instantaneous buoyancy field with vertical suppression of turbulence by stable stratification. Transition to turbulence occurred through a Kelvin-Helmholtz shear instability, as seen in Fig 3.10. In absence of an analytical solution for turbulent katabatic flow, IB schemes are compared to simulations using a body-fitted mesh in GIN3D. Mean velocity profiles from Schemes 2–5 are compared with a body-fitted mesh simulation in Fig. 3.11. Both linear and parabolic velocity reconstruction were used with Schemes 2–5. Though these schemes gave good agreement with the analytical solution in laminar simulations, errors in the peak velocity of the low level jet are easily seen for Schemes 2, 3, and 5 in DNS. Compared to linear velocity reconstruction, parabolic velocity reconstruction had the effect of decreasing the peak velocity for all schemes, which brought Scheme 2 close to agreement with the body-fitted result, but moved the other schemes farther away. Scheme 4 with linear velocity reconstruction agreed well with the body-fitted mesh simulation and showed only slight deviation near the up-slope return flow region. As in the laminar case, discrepancies in buoyancy are less apparent in Fig. 3.12; however, Scheme 4 with linear velocity reconstruction agreed well with the body-fitted mesh simulation for this plot as well.

Of the schemes implemented in the present work, Scheme 4, unique to this thesis, gave the best performance and has proven suitable to DNS of turbulent katabatic flow on a sloped plane. Future work would be validation of Scheme 4 in cases that include more complex geometry, such as flow over a heated cylinder or sphere. The next step would be extension to simulating wind in complex terrain with thermally active surfaces.

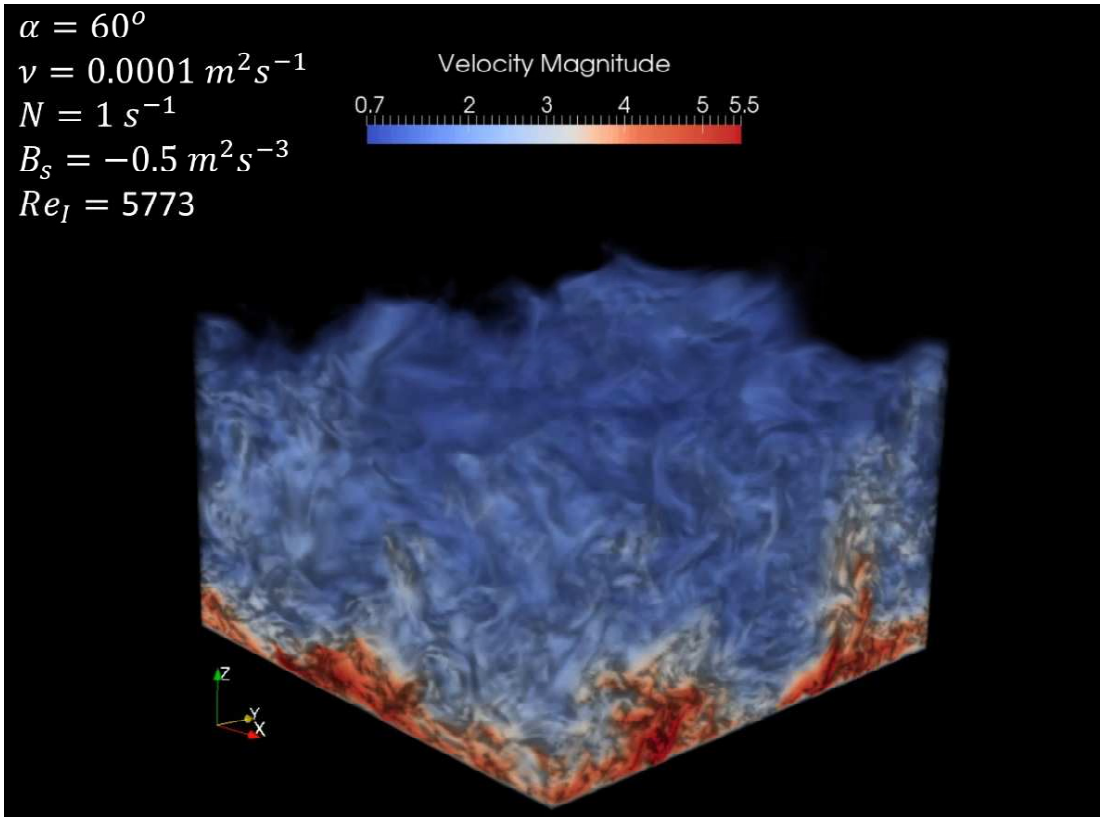


Figure 3.8: Volume rendering of instantaneous velocity magnitude from direct numerical simulation of turbulent katabatic flow. The simulation is for an infinite plane inclined at slope angle $\alpha = 60^o$, however, the slope is rotated down into an isometric view here to better show the turbulence throughout the domain.

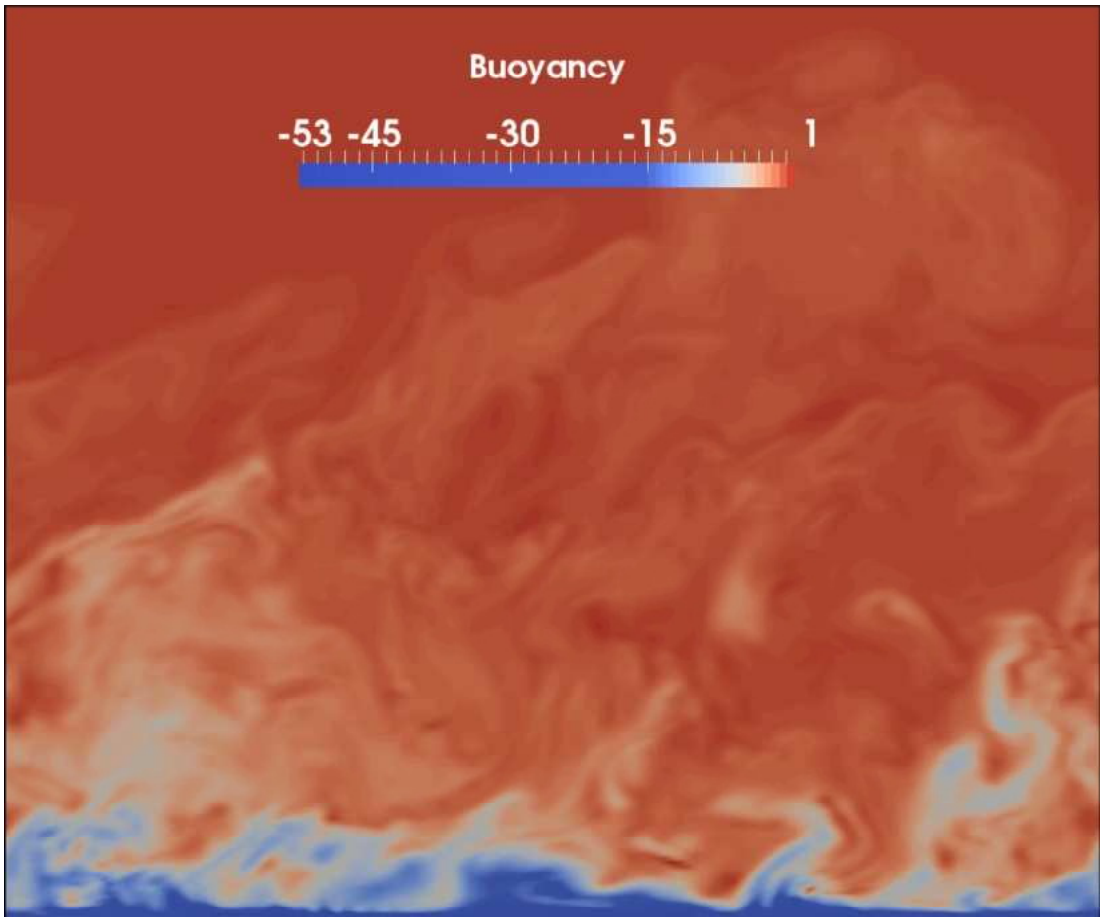


Figure 3.9: 2D slice of the instantaneous buoyancy field from direct numerical simulation of turbulent katabatic flow.

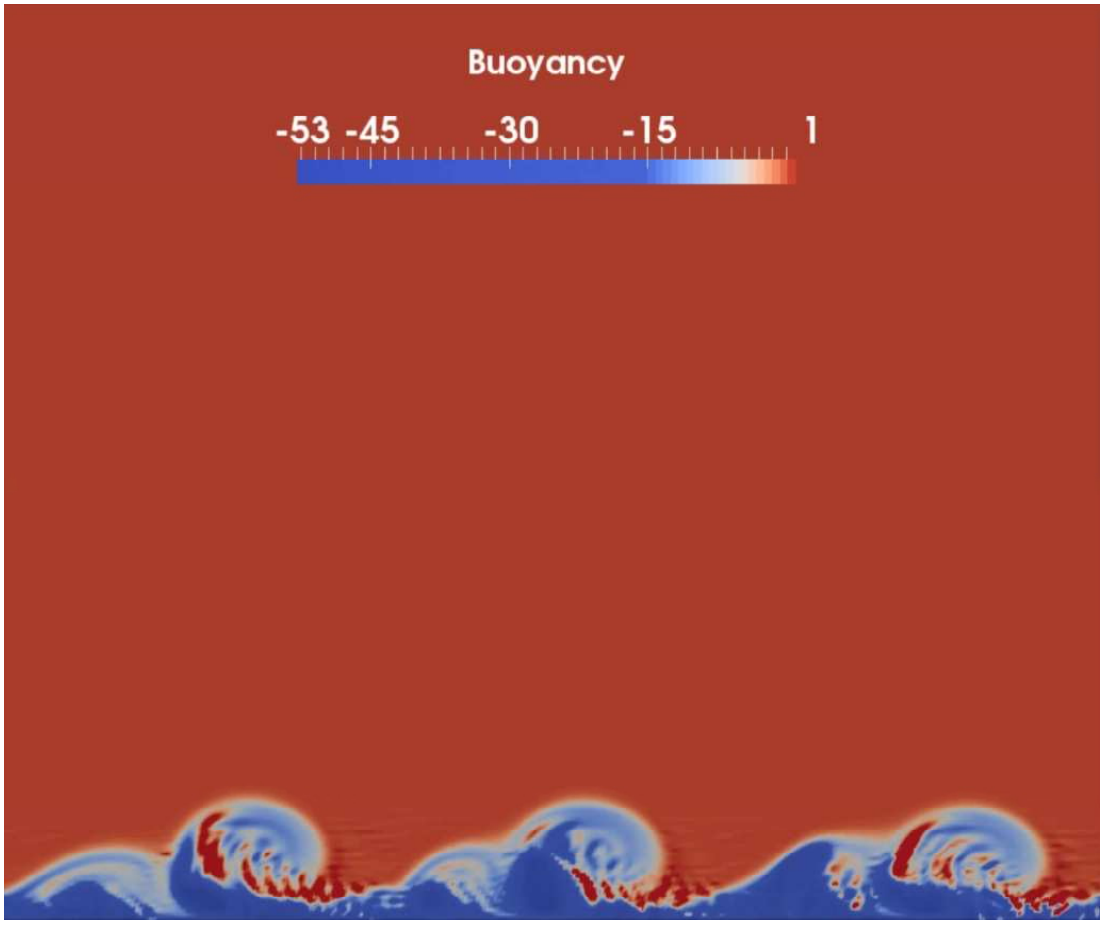
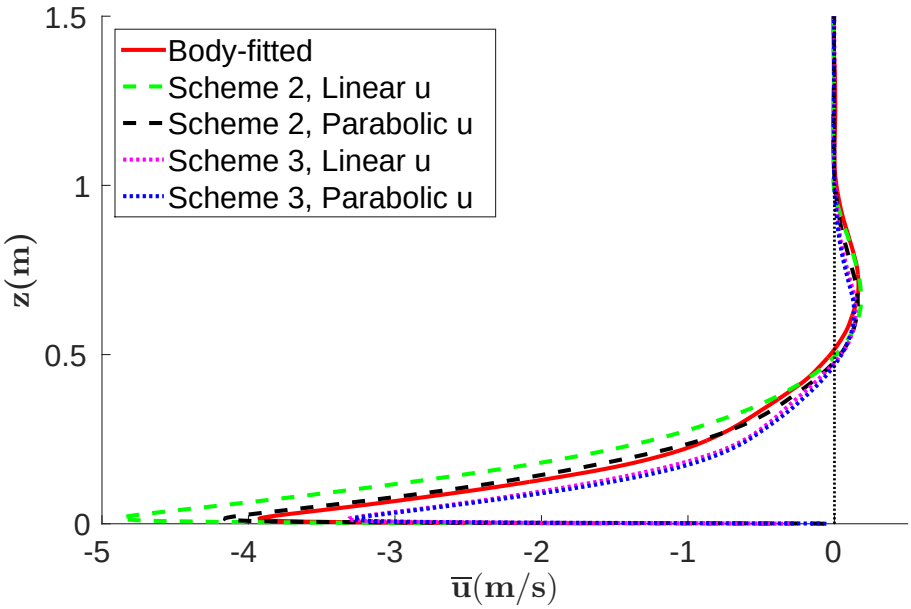
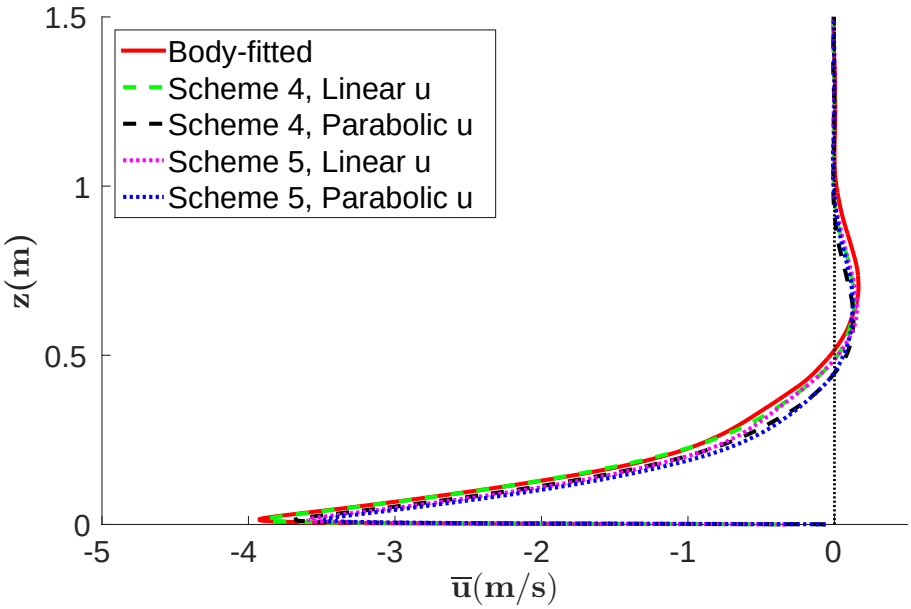


Figure 3.10: Kelvin-Helmholtz instability during transition to turbulence shown in a 2D slice of the instantaneous buoyancy field from direct numerical simulation of turbulent katabatic flow.

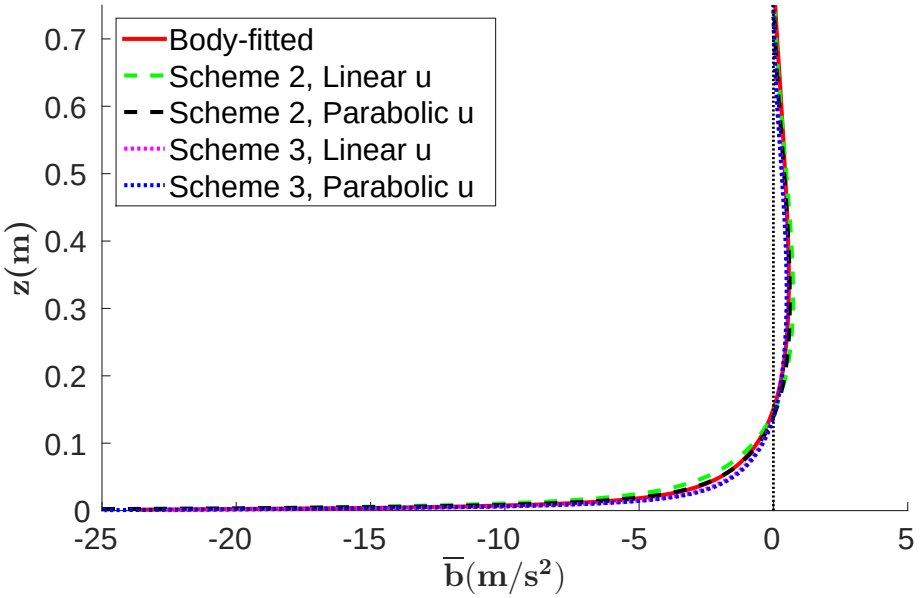


(a)

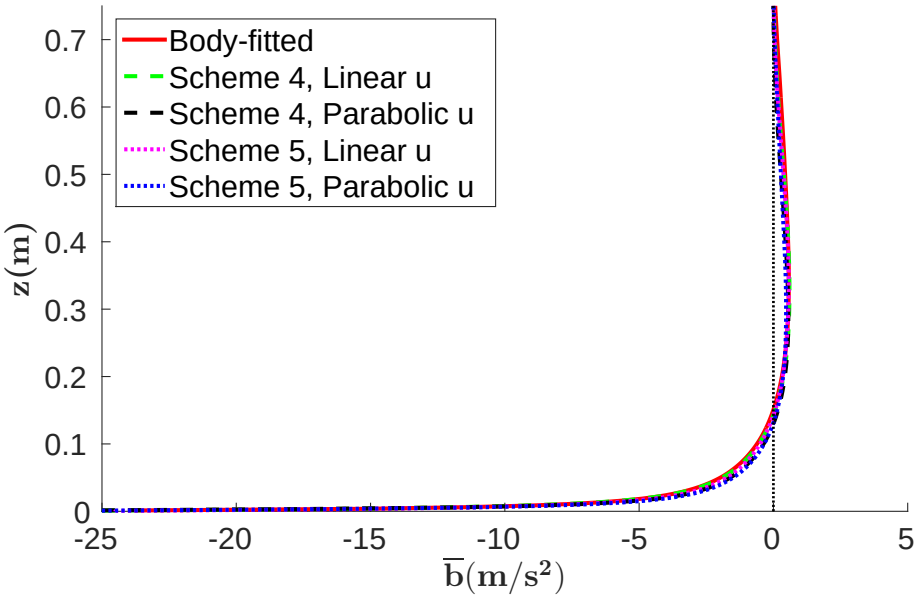


(b)

Figure 3.11: Comparison of down-slope velocity from different buoyancy reconstruction schemes with a body-fitted mesh simulation for direct numerical simulation of turbulent katabatic flow.



(a)



(b)

Figure 3.12: Comparison of buoyancy from different buoyancy reconstruction schemes with a body-fitted mesh simulation for direct numerical simulation of turbulent katabatic flow.

CHAPTER 4

TURBULENT INFLOW FOR THE LARGE-EDDY SIMULATION TECHNIQUE

Large-eddy simulation (LES) is a turbulence modeling technique in which the filtered form of the Navier-Stokes equations are solved. Unlike the Reynolds-averaged Navier-Stokes (RANS) technique where only the largest integral scale of the flow field is resolved, the goal in LES is to resolve eddies down to the filter cutoff. Turbulent eddies are commonly divided into three ranges:

- Energy-containing range—the largest eddies, which are anisotropic and are affected by the boundary conditions of the flow;
- Inertial subrange—small isotropic eddies which are still large enough for inertial effects to dominate and viscous effects to be negligible;
- Dissipation range—the smallest eddies, which are isotropic, dominated by viscous effects, and are responsible for essentially all of the energy dissipation.

Credible LES simulation aims to resolve a portion of the so-called inertial subrange of turbulence [40]. Compared to RANS, this is a computationally expensive task as eddies have to be resolved that are much smaller than the integral scale both in time and space. The fundamental assumption in LES is that eddies smaller than the filter cutoff scale, which are referred to as subgrid scale (SGS) in literature, are isotropic in

nature, and a simple eddy viscosity model would be sufficient to parameterize their effect on the flow field. Therefore, in LES, we refer to *resolved* and *modeled* stresses. In order to capture the correct stress field, resolved eddies need to simulate the *energy cascade* process in which larger eddies break into smaller eddies and the smallest eddies are dissipated by the action of the SGS eddy viscosity model. In a direct numerical simulation (DNS) of turbulence, all scales are resolved by the simulation, and the eddies with the smallest physical scale, known as the Kolmogorov scale, are dissipated by the action of the physical viscosity. Therefore, DNS is computationally very expensive and only feasible for a few fundamental problems. To this end, there is a great interest to broaden the application of LES to practical problems. However, there are several scientific challenges to achieving this goal. Proper formulation of inlet conditions is one of them. Unlike the RANS technique, one cannot simply define a mean profile at the inlet as it does not initiate the energy cascade in the simulation. A turbulent inflow boundary condition is needed to sustain the energy cascade within the simulation domain. A plethora of research has been done in this particular area.

Perhaps the simplest method to generate realistic turbulence in LES is to run a *periodic standalone* simulation in which flow is forced through a periodic (or cyclic) domain until turbulence becomes fully developed [47]. The solution at the outlet is imposed as the inlet condition. This method avoids prescribing an inlet and does not require additional computation once the flow is developed; however, the inlet is essentially the wake of an infinite series of identical domains in front of the domain of interest. *Pre-cursor* methods avoid a false wake as an inlet by running a periodic standalone simulation in a domain that represents the correct inflow condition, and then using the solution in a vertical plane parallel to the flow in this auxiliary domain as the inlet condition for the domain of interest. Issues of the pre-cursor

method include periodicity in the inflow (based on the auxiliary domain period), storage requirements for each inflow database of interest, and subsequent slowdown of computations during file reading.

Recycling methods avoid running auxiliary simulations and database storage by reinserting the solution from a vertical plane inside the domain of interest back onto the inlet at each time step [26]. In this method, turbulence becomes fully developed between the inlet and cyclic planes, yet periodicity remains an issue. Avoiding any kind of periodicity or recycling, *Synthetic turbulence* methods directly prescribe an inlet condition based on expected turbulence statistics [24]. While synthetic turbulence methods approximate certain aspects of turbulent flow, they can require long fetches to break down into realistic turbulence. Equally important, the previous three approaches do not readily extend to complex geometry flows.

The *buoyancy perturbation* method has been proposed in recent work by Muñoz-Esparza et al. [34, 36] in simulations of the atmospheric Ekman flow. This method triggers the development of realistic turbulence by applying perturbations to the buoyancy field near the domain inlet. This allows a mean velocity profile to be prescribed at the inlet, with turbulence naturally emerging in the flow after a short fetch. Details of this method are discussed in the next section.

The buoyancy perturbation method by Muñoz-Esparza et al. was not applied to LES with near-wall resolution, in which the filter and grid are sufficiently fine to resolve 80% of the energy in the velocity field throughout the entire domain [40], which is done often in LES of engineering flows. Furthermore, the method has not been applied to a well-studied turbulence modeling validation case, for which profiles of Reynolds stresses from DNS can be used to investigate the turbulence quality. This thesis modifies and extends the work of Muñoz-Esparza et al. to generic flow

problems. Specifically, the well-known turbulent channel flow with a mean logarithmic profile is considered using near-wall resolution, and the effect of globally neutral buoyancy perturbations on the Reynolds stress turbulence statistics is studied.

4.1 Buoyancy Perturbation Method

Muñoz-Esparza et al. [34, 36] have shown that perturbations to the buoyancy field near the inlet boundary can trigger the development of turbulence within a relatively short fetch for a neutrally stratified atmospheric boundary layer (ABL) flow. The application of interest was nesting a LES domain within a mesoscale atmospheric flow simulation using the Weather Research and Forecasting (WRF) Model framework. For reference in judging the turbulence quality, comparison of velocity spectra in each study was made with a periodic standalone simulation of the LES domain. Random perturbations of specified amplitude bounds were added directly to the potential temperature field. Of the perturbation methods explored in [34], the *cell perturbation* method was found to perform best in regard to turbulence development within the shortest fetch, and ease of implementation with a low computational cost. Stacked square cells of 8×8 grid points in the horizontal plane were specified in the LES domain near the inlet, in which pseudo-random potential temperature perturbations uniformly distributed in the interval $[-0.5, \ +0.5]$ K were applied, as depicted in Figure 4.1. Turbulence wavelengths lower than approximately $6\Delta x$ are rapidly dissipated in WRF, therefore $8\Delta x$ was selected as the perturbation cell size.

Upon further study of the cell perturbation method, Muñoz-Esparza et al. [35, 36] found the optimum Eckert number, E_c, and dimensionless perturbation time scale, Γ, to be $E_c = 0.2$ and $\Gamma = 1$ in equations

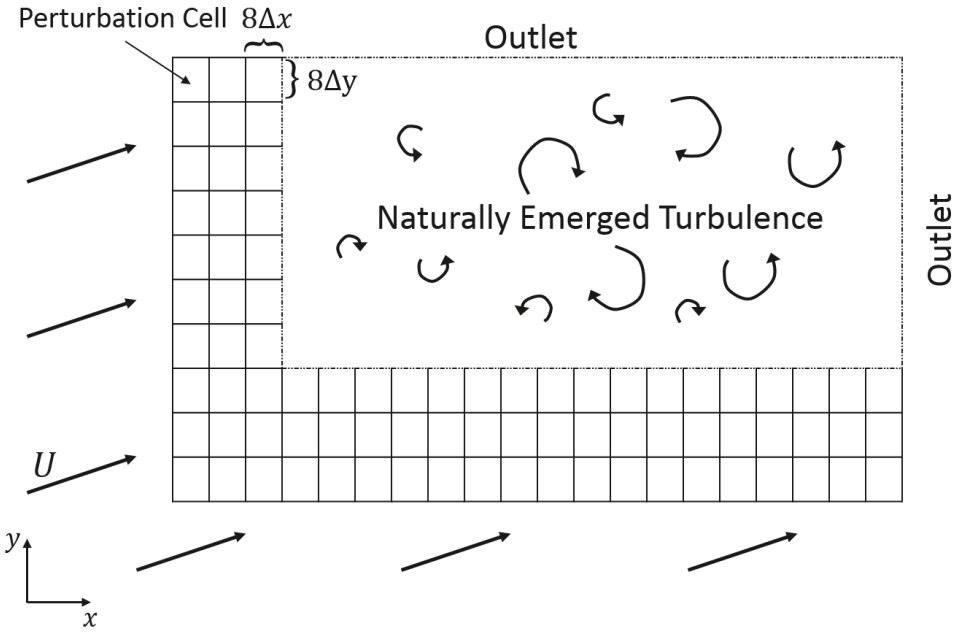


Figure 4.1: Top view sketch of the cell perturbation method used by Muñoz-Esparza et al. [34] in which pseudo-random perturbations uniformly distributed in the interval $[-0.5,\ +0.5]$ K were applied to the potential temperature field in 8×8 grid-point horizontal planes (perturbation cells) stacked near the inlet to trigger the development of natural turbulence.

$$E_c = \frac{U_g^2}{c_p \tilde{\theta}_{pm}}, \tag{4.1}$$

$$\Gamma = \frac{t_p U_1}{d_c}, \tag{4.2}$$

where U_g is the geostrophic wind, c_p is the specific heat capacity at constant pressure, $\tilde{\theta}_{pm}$ is the maximum perturbation amplitude, t_p is the perturbation time at which the next potential temperature perturbation is applied, U_1 is the velocity magnitude at the first vertical grid point imposed at the boundary of the LES domain, and d_c is the diagonal of the cell. Optimum results were obtained when perturbation cell size corresponded to the fully resolved inertial subrange wavelengths of the turbulence, rather than the energy-containing range. The cell perturbation method was found

to develop realistic turbulence within a shorter fetch compared to a synthetic turbulence method, and is attractive because it is simple to implement, computationally inexpensive, and requires minimal turbulence information.

4.2 Buoyancy Box Perturbation Method for General Turbulent Inlet Boundaries

In this thesis, an implementation analogous to the cell perturbation method by Muñoz-Esparza et al. [34, 36] has been developed within the multi-GPU parallel incompressible wind solver GIN3D. While Muñoz-Esparza et al. primarily made comparison with velocity spectra of turbulence at a point above the ground in a periodic standalone simulation, the interest of this thesis is a more rigorous investigation of turbulence statistics by observing profiles of mean velocity and Reynolds stresses compared to DNS. To this end, the turbulent channel flow case has been chosen for study, as it is a fundamental test case for turbulence models and has extensive DNS data on turbulence statistics readily available.

To generalize the turbulent inlet for engineering and atmospheric flows, there are several differences in the present buoyancy perturbation implementation compared to that of Muñoz-Esparza et al. in WRF. For the fine meshes used to resolve the viscous sublayer in turbulent channel flow, cell regions with vertical height, rather than the horizontal plane cell regions used in WRF, are better suited to perturb the buoyancy field. The goal is to perturb a volume of fluid with dimensions that relate to certain length scales of the turbulence (to be discussed in the next paragraph). As a volume of fluid is resolved by an increasingly fine mesh, the net effect of uniformly distributed pseudo-random perturbations in horizontal planes within the volume will

increasingly approach zero. Therefore, the method implemented in this thesis will maintain a perturbation cell height and be referred to as the *box perturbation* method. For coarse meshes without near-wall resolution, a box height of one vertical grid cell Δz may be appropriate, as was used in WRF.

As mentioned previously, Muñoz-Esparza et al. obtained optimum results when perturbation cell size was below the energy-containing range and corresponded to the fully resolved inertial subrange wavelengths of the turbulence. In order to prescribe perturbation box dimensions that fulfill this condition, an estimate of the inertial subrange for each case is required, as well as the threshold at which turbulence wavelengths are rapidly dissipated for the employed numerical schemes and models. Similar to WRF, turbulence wavelengths lower than approximately 6Δ are rapidly dissipated in GIN3D, therefore 8Δ was considered the minimum box dimension in the horizontal directions with respect to each Δx and Δy grid spacing. In order to estimate the inertial subrange for turbulent channel flow, the following turbulence length scales were consider, listed from largest to smallest:

- L_o—outer scale of the turbulence characterizing the largest eddies;
- L_{DI}—demarcation between the inertial subrange and the dissipation range;
- η—Kolmogorov microscale describing the smallest turbulent eddies.

The Kolmogorov microscale can be estimated from [25]

$$\frac{\eta}{L_o} \approx Re_L^{-3/4}, \tag{4.3}$$

and

$$Re_L = \frac{\Delta U L_o}{\nu}, \tag{4.4}$$

where L_o is taken as the channel half height, δ, ΔU is the velocity difference taken as the peak mean velocity at δ, and ν is the kinematic viscosity. Then, the demarcation between the inertial subrange and the dissipation range can be found as $L_{DI} = 60\eta$ from Pope [40]. In the present work, perturbation box dimensions in the horizontal direction were chosen to be near the average between L_o and L_{DI}, which was above the 8Δ minimum and assumed to be in the inertial subrange. The box height was chosen to be near $2L_{DI}$.

Continuing with key differences, the update time, t_p, was constant with height in WRF, but is a function of the inlet velocity, $U_{in}(z)$, here:

$$t_p = \frac{X_p}{U_{in}(z)}, \tag{4.5}$$

where X_p is the box length in the streamwise direction. This approach results in an update time that is consistent with the local time needed to advect the flow across the wavelength perturbed by a box. Perturbations to the temperature field are applied as a source term in the heat balance equation, rather than directly adding to the temperature variable, as in WRF. This is seen as a more stable approach. Maximum perturbation source amplitudes, $\tilde{\Phi}_{pm}$, have been prescribed by considering the simplified heat balance equation that ignores transport and diffusion of temperature for a fluid element traversing a perturbation box during perturbation time, t_p, as follows:

$$\tilde{\Phi}_{pm} = \frac{\Delta T}{t_p} \tag{4.6}$$

where ΔT is the change in temperature. Dividing both sides of Eq. 4.6 by the uniform inlet temperature T_∞ to normalize, and defining a temperature intensity, $T_i \equiv \Delta T/T_\infty$, yields

$$\tilde{\Phi}_{pm} = \frac{T_i T_\infty}{t_p}. \tag{4.7}$$

Substuting Eq. 4.5 into Eq. 4.7 gives the final form:

$$\tilde{\Phi}_{pm} = \frac{T_i T_\infty U_{in}(z)}{X_p}. \tag{4.8}$$

Pseudo-random temperature source perturbations uniformly distributed in the interval $[-\tilde{\Phi}_{pm},\ +\tilde{\Phi}_{pm}]$ are then applied to perturbation boxes near the inflow boundary. Equation 4.8 is proposed as a consistent way to prescribe maximum perturbation source amplitudes based on presumed temperature intensity, considering that the temperature source must be greater for fluid elements that traverse a perturbation box within a smaller perturbation time if the same buoyancy forcing is desired. In this way, the inlet boundary condition is prescribed by providing a T_i and U_{in} profile, as well as perturbation box dimensions and the number of boxes in the stream-wise direction from the inlet.

4.3 Buoyancy Box Perturbation Method Validation

Flow visualization of results for LES of turbulent channel flow with friction Reynolds number 395 (defined by $Re_\tau = u_\tau \delta/\nu$, where u_τ is the friction velocity) is shown in Fig. 4.2 for the box perturbation method. Laminar incoming flow is seen to be tripped by the random perturbations to the temperature field, leading to a fully developed

turbulent flow. The effect of perturbation boxes on temperature near the inlet can be seen in Fig. 4.3. Simulation settings were:

- Friction velocity: $u_\tau = 1.194 \times 10^{-2}\ m/s$;
- Kinematic viscosity: $\nu = 1.511 \times 10^{-5}\ m^2/s$;
- Thermal diffusivity: $\gamma = 2.119 \times 10^{-5}\ m^2/s$;
- Thermal expansion coefficient: $\beta = 3.430 \times 10^{-3}\ K^{-1}$;
- Reference density: $\rho_\infty = 1.205\ kg/m^3$;
- Reference temperature: $T_\infty = 293.0\ K$
- Gravitational acceleration: $g = 9.810\ m/s^2$;
- Channel half height: $\delta = 0.5\ m, 831.1\eta$;
- Domain size: $(X \times Y \times Z\times) = 6\pi\delta \times \pi\delta \times 2\delta$;
- Numerical grid dimensions: $(NX \times NY \times NZ) = 769 \times 129 \times 257$;
- Numerical grid spacing: $(\Delta x \times \Delta y \times \Delta z) = 0.01228\,m\times0.01236\,m\times0.003922\,m$;
- Span-wise boundary condition: periodic;
- Top & bottom boundary condition: no-slip velocity, zero normal gradient temperature;
- Inlet: logarithmic law of the wall, $u^+ = 1/\kappa \ln y^+ + B$, for velocity (where $u^+ = U(z)/u_\tau$, $y^+ = u_\tau z/\nu$, $\kappa = 0.41$, and $B = 5.2$) [40], T_∞ for temperature;
- Outlet: Convective outlet for velocity, zero normal gradient temperature

- Lagrangian dynamic subgrid-scale eddy viscosity model;
- Perturbation boxes in the x-direction from the inlet: 6;
- Perturbation box dimensions: $(X_p \times Y_p \times Z_p) = 449.1\eta \times 452.1\eta \times 143.4\eta$.

Buoyancy effects are included by the Boussinesq approximation with buoyancy forcing, F_B, in the z-component of the momentum equations as [12, 25]:

$$F_B = g\rho_\infty \beta \left(T - T_\infty\right), \tag{4.9}$$

where T is the local temperature. To maintain stability, a weighted average between upwind and central difference was used in solving for temperature [20], with 25% upwinding. Muñoz-Esparza et al. found 3 perturbation cells to give optimum results; however, the number of perturbation boxes in the x-direction from the inlet was chosen to be 6 in the present work because this showed more uniform turbulence generation with the box perturbation method.

Figure 4.4 shows the mean velocity profiles using three different T_i values ($5 \times 10^{-4},\ 7 \times 10^{-4},\ 9 \times 10^{-4}$) that were uniform in time and space in each simulation. The first $2\pi\delta$ of the domain in the stream-wise direction was ignored in all statistics. Comparison is made with the theoretical logarithmic law of the wall and DNS performed by Moser et al. [33]. $T_i = 9 \times 10^{-4}$ gives the best agreement from the wall into the log region, where it then falls the farthest below the DNS in the channel center. Figure 4.5 zooms in on the channel center where it can be seen that $T_i = 5 \times 10^{-4}$ has the best agreement with the DNS in the log-law region.

Figure 4.6 shows the Reynolds shear stress, and clearly depicts a relationship of increasing magnitude with increasing T_i. This direct relationship is also seen in the

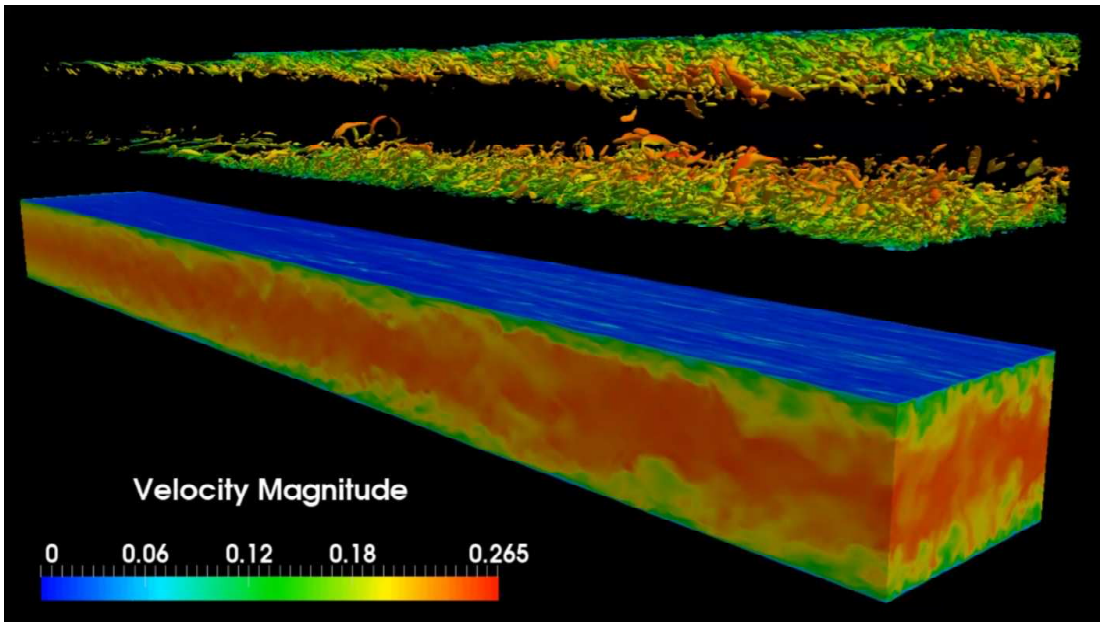


Figure 4.2: Flow visualization of the box perturbation method applied to channel flow with $Re_\tau = 395$. Top shows iso-contours of Q criterion colored by velocity magnitude. Bottom shows velocity magnitude.

normal Reynolds stresses, indicating that T_i may be tuned to better match the correct turbulence statistics. In this case, $T_i = 5 \times 10^{-4}$ gave a close fit with the DNS for Reynolds shear stress and the normal stress in the y-direction (Fig. 4.8). All values of T_i gave an overshoot for the maximum normal Reynolds stress in the x-direction, shown in Fig. 4.7. The normal Reynolds stress in the z-direction can be seen in Figure 4.9, where each value of T_i produced high stresses in the center of the channel. Earlier simulations were conducted with a constant maximum perturbation source amplitude, $\tilde{\Phi}_{pm}$, which effectively gives T_i a profile with the lowest value in the center (this relationship can be seen in Eq. 4.8, considering that $U_{in}(z)$ increases away from the wall). The resulting normal Reynolds stress in the z-direction, shown in Figure 4.10, was lower in the channel center, indicating that these stresses may be controlled by a variable T_i profile to improve agreement with the DNS.

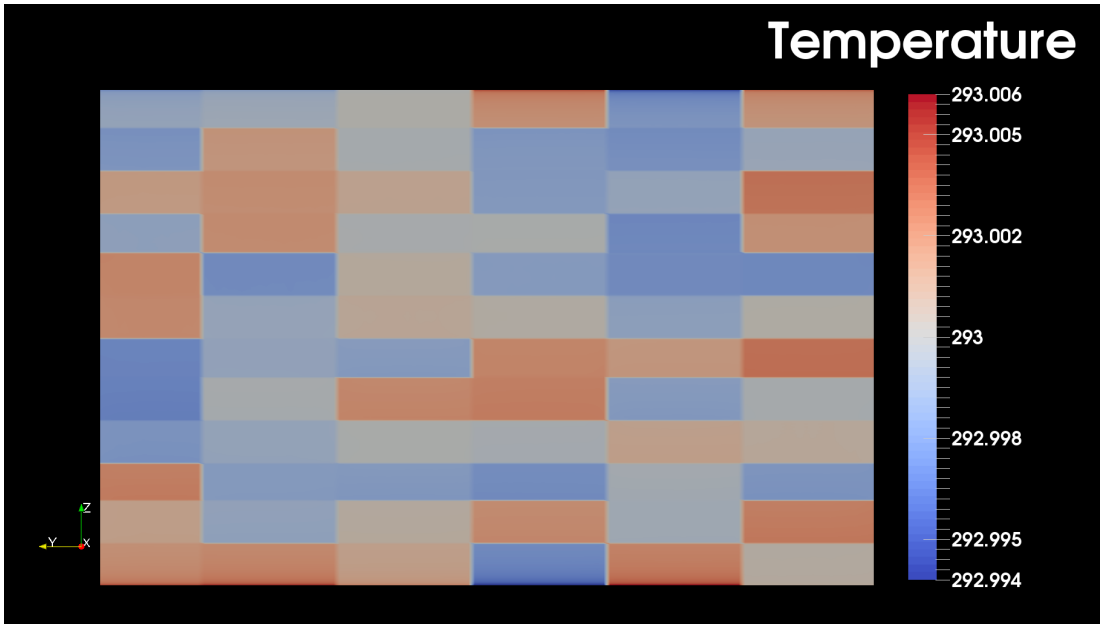


Figure 4.3: Visualization of the effect of perturbation boxes on temperature near the inlet from a slice with a stream-wise normal. The box perturbation method applied to channel flow with $Re_\tau = 395$.

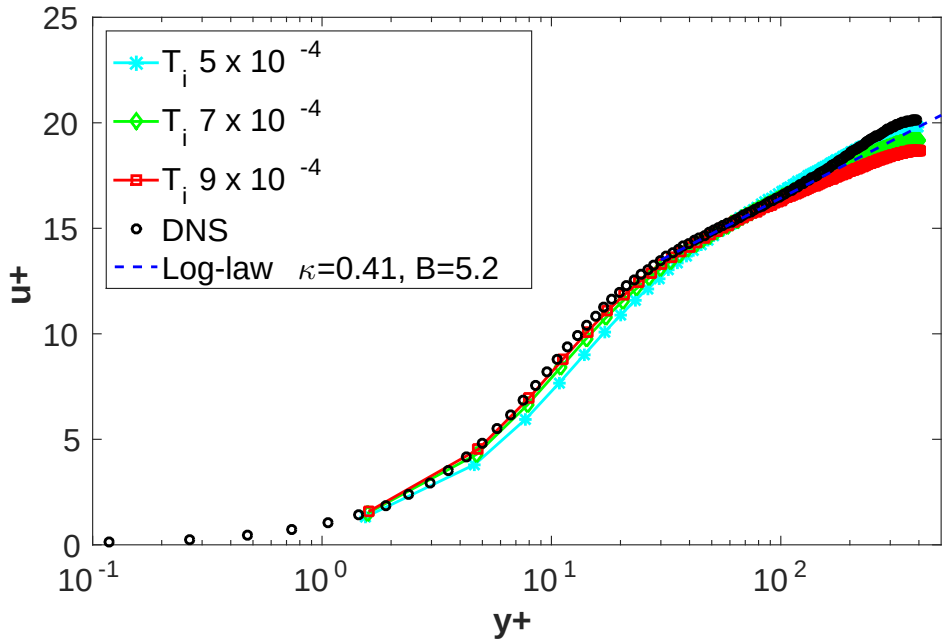


Figure 4.4: Mean velocity profiles from the box perturbation method applied to channel flow with $Re_\tau = 395$. DNS performed by Moser et al. [33].

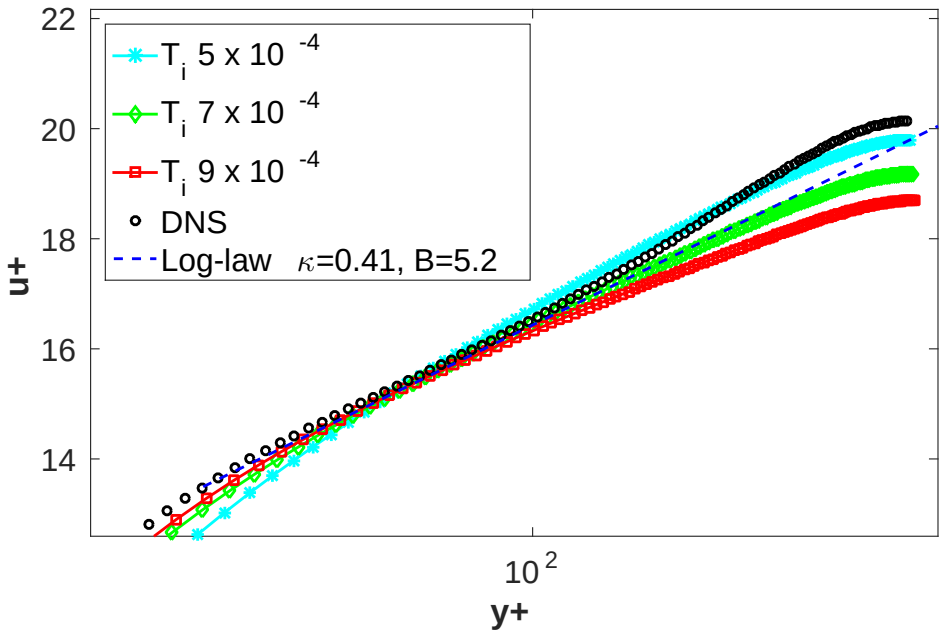


Figure 4.5: Close-up of the channel center for the mean velocity profiles in Fig. 4.4. The box perturbation method applied to channel flow with $Re_\tau = 395$. DNS performed by Moser et al. [33].

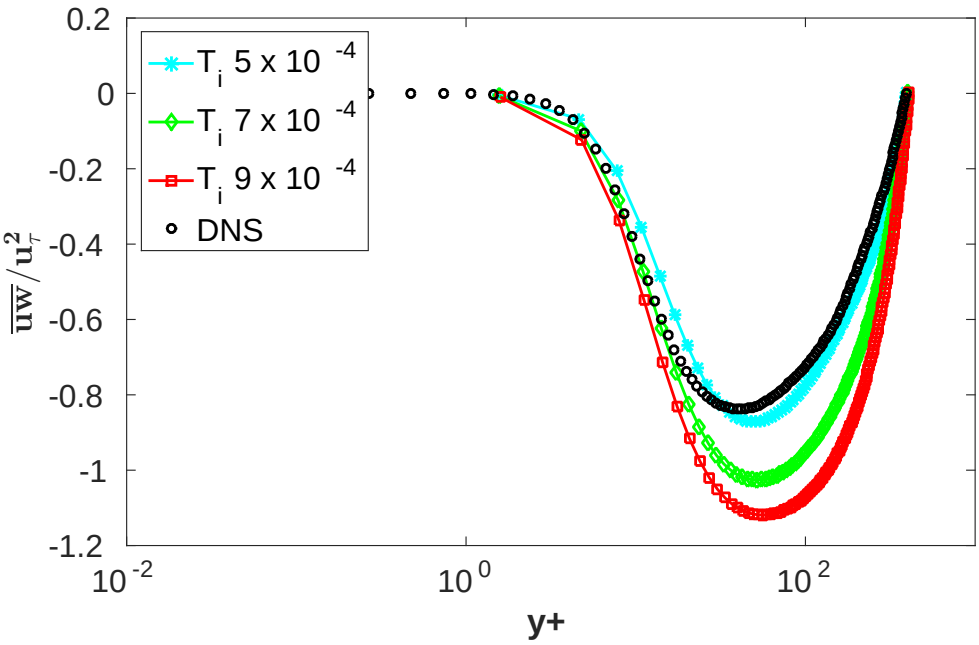


Figure 4.6: Comparison of the normalized τ_{13} component of the Reynolds stress tensor from the box perturbation method applied to channel flow with $Re_\tau = 395$. DNS performed by Moser et al. [33].

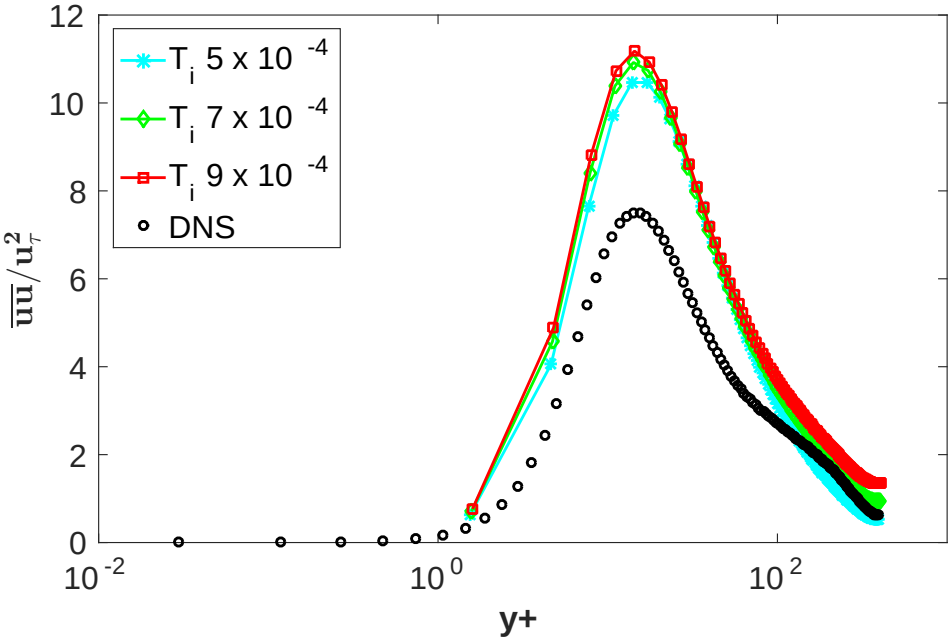


Figure 4.7: Comparison of the normalized τ_{11} component of the Reynolds stress tensor from the box perturbation method applied to channel flow with $Re_\tau = 395$. DNS performed by Moser et al. [33].

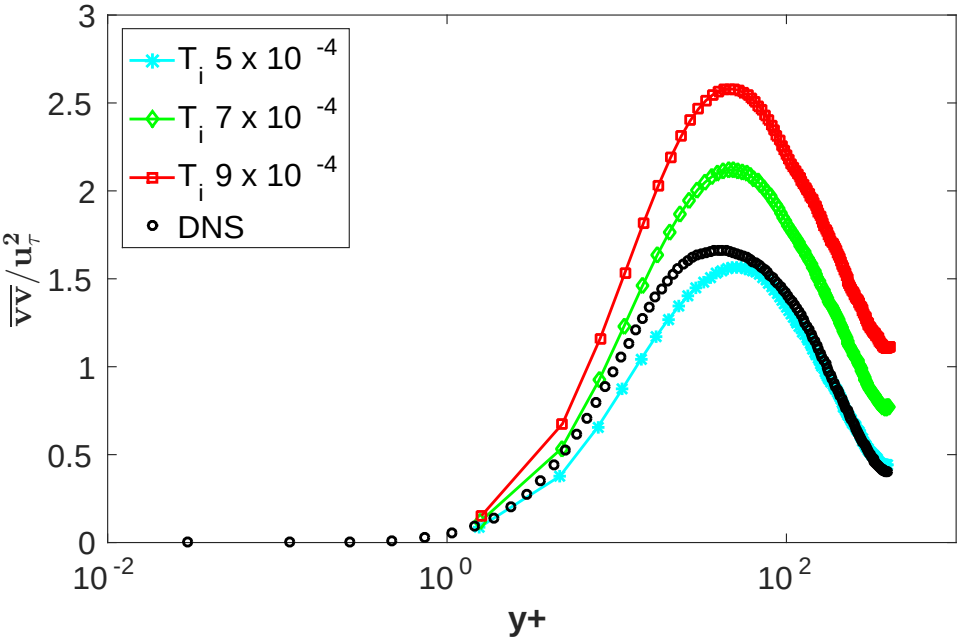


Figure 4.8: Comparison of the normalized τ_{22} component of the Reynolds stress tensor from the box perturbation method applied to channel flow with $Re_\tau = 395$. DNS performed by Moser et al. [33].

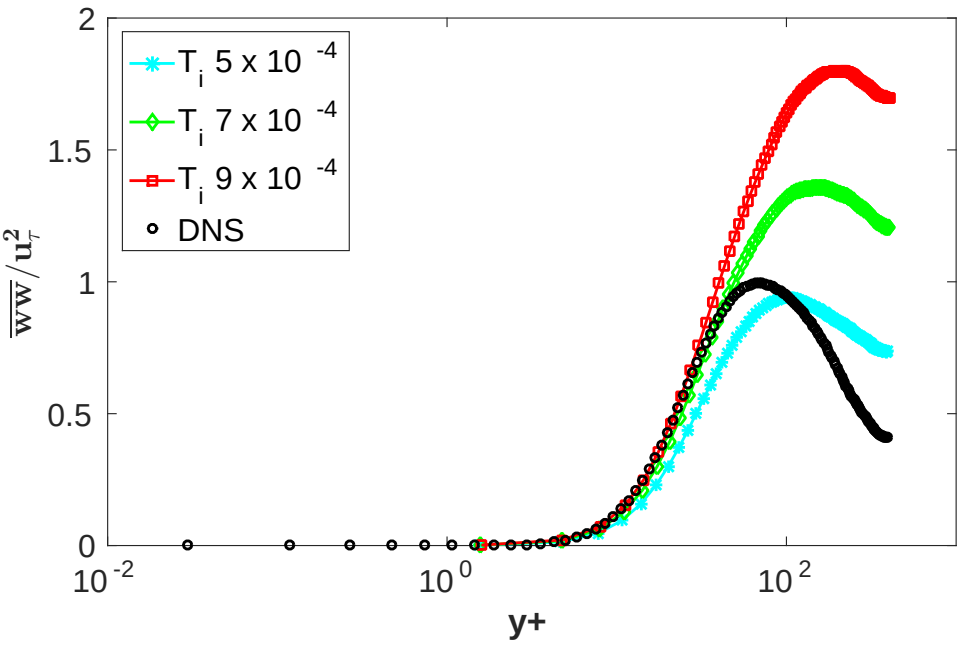


Figure 4.9: Comparison of the normalized τ_{33} component of the Reynolds stress tensor from the box perturbation method applied to channel flow with $Re_\tau = 395$. DNS performed by Moser et al. [33].

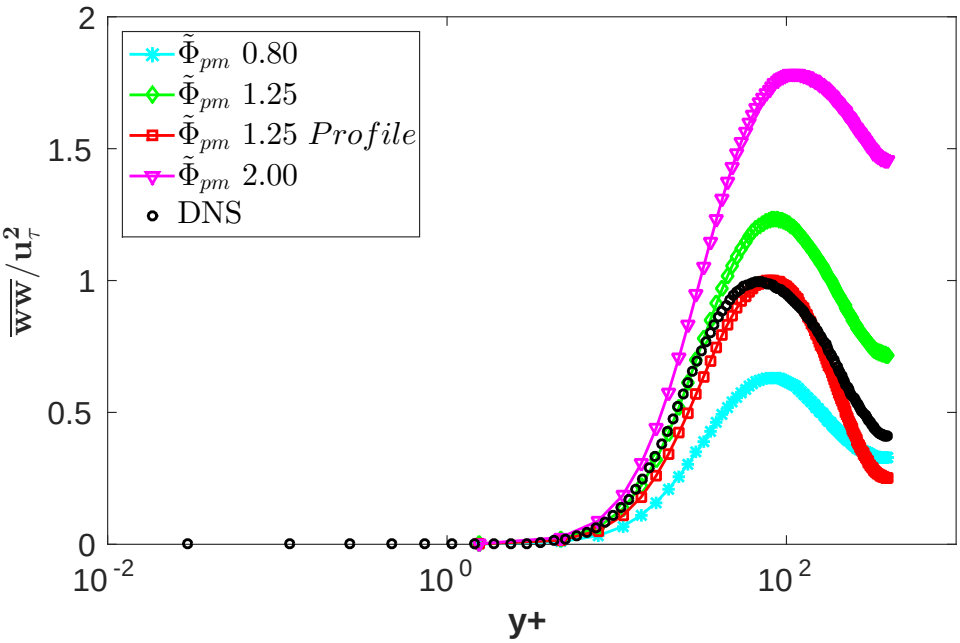


Figure 4.10: Comparison of the normalized τ_{33} component of the Reynolds stress tensor from the box perturbation method applied to channel flow with $Re_\tau = 395$. Uniform $\tilde{\Phi}_{pm}$ was used in these simulations, except for the legend entry $\tilde{\Phi}_{pm}$ 1.25 $Profile$, which used the DNS profile scaled by 1.25. DNS performed by Moser et al. [33].

The box perturbation method developed in this thesis shows promise as a simple method with low computational cost for generating turbulent inflow in LES of engineering and atmospheric flows. It has been shown that Reynolds stresses are sensitive to temperature intensity, T_i, and agreement with DNS can likely be improved to an acceptable level by varying T_i to have lower values at heights with lower Reynolds stresses. Guidance on choosing such a T_i profile has not been thoroughly explored. While sensitivity of the mean velocity profile and Reynolds stresses to T_i has been investigated, the sensitivity to changes in perturbation box dimensions and number of boxes from the inlet remains a topic for future work.

CHAPTER 5

SUMMARY

5.1 Buoyancy Reconstruction Schemes

Immersed boundary (IB) method on a structured Cartesian grid is well suited for including complex terrain geometry in wind simulations because it is computationally efficient and avoids errors possible in body-fitted meshes from skewed cells. Much of the work done with IB methods has focused on reconstruction of the velocity field, while treatment of boundary conditions for heat transfer has received less coverage despite their significance in many engineering flows. In this thesis, five schemes have been introduced to reconstruct buoyancy in the near surface IB node based on the buoyancy gradient boundary condition in fundamental katabatic flows. These schemes were implemented and studied within the multi-GPU parallel wind solver GIN3D.

Scheme 1, by Gilmanov et al. [19], uses the buoyancy gradient boundary condition at the surface and known buoyancy gradient in the fluid together with a central difference along the IB line to calculate the buoyancy at the IB node. Approximation of the buoyancy gradient in the fluid was done using a central difference that included the IB node, which in turn polluted the solution such that laminar katabatic flow could not be accurately simulated. The remaining schemes are unique to this thesis, as far as the author is aware. Scheme 2 avoids approximation of the buoyancy gradient in the fluid, and assumes a constant gradient, equal to the boundary condition, in a

central difference along the IB line to calculate the buoyancy gradient at the IB node. Scheme 3 is identical to Scheme 1, except approximation of the buoyancy gradient in the fluid is done using a second order accurate one-sided difference that excludes the IB node. Scheme 4 also avoids approximation of the buoyancy gradient in the fluid. The surface buoyancy gradient is moved to the IB node and used in a second order accurate one-sided difference with two cell face intersection points from the fluid to calculate the buoyancy at the IB node. Scheme 5 is identical to Scheme 3, except it interpolates the buoyancy gradient to the IB node and then proceeds as in Scheme 4 with a second order accurate one-sided difference to calculate the buoyancy at the IB node.

Schemes 2 and 3 had good agreement with the analytical solution for laminar katabatic flow, and showed first order accuracy locally and second order globally. Results for Schemes 4 and 5 were shown for turbulent katabatic flow only. Success in laminar flows does not readily extend to the turbulent regime. Therefore, direct numerical simulation (DNS) of turbulent katabatic flow was performed to assess the true performance of these schemes. Scheme 4 combined with linear velocity reconstruction agreed well with body-fitted mesh results in DNS of turbulent katabatic flow. Future work would apply Scheme 4 in cases with heat transfer on more complex geometries, such as a cylinder or sphere, and in simulating wind in complex terrain with thermally active surfaces.

5.2 Buoyancy Box Perturbation Method

Large-eddy simulation (LES) of turbulent flow resolves the large eddies, which extract energy from the mean flow and are highly dependent on the geometry and

boundary conditions of the domain, while modeling smaller eddies that exhibit a more universal, isotropic behavior. An ongoing challenge in extending the LES technique to practical engineering flows is the need to generate turbulent inflow conditions that can trigger and sustain the expected energy cascade of turbulence in the flow domain. Several methods for generating realistic turbulent inflow conditions have been proposed, each with strengths and weaknesses, yet the topic of turbulent inflow boundary conditions remains an active research area.

Muñoz-Esparza et al. [34, 36] proposed the cell perturbation method—random potential temperature perturbations in square horizontal planes stacked near the inlet—to give variable buoyancy forcings that trip the natural evolution of turbulence within a short fetch for the atmospheric Ekman flow. This thesis followed the perturbation idea of Muñoz-Esparza et al., and proposed a new turbulent inlet boundary condition based on perturbing the flow field through random temperature sources in small boxes that divide up the inlet region. This new procedure is called the box perturbation method. The inlet boundary condition is prescribed by providing a temperature intensity and incoming velocity profile, as well as perturbation box dimensions and the number of boxes in the stream-wise direction from the inlet. Validation of this method was done using turbulent channel flow, a case fundamental to studies on turbulent inlet methods. Results show that agreement with turbulence statistics from DNS of channel flow can likely be improved to an acceptable level by controlling the temperature intensity profile in the box perturbation method. This new method, unique to this thesis, shows promise as a simple method with low computational cost for generating turbulent inflow in LES of engineering as well as atmospheric flows. Future work would involve determining a guide in choosing a temperature intensity profile for a particular case, studying the sensitivity to changes

in perturbation box dimensions and number of boxes from the inlet, and extension of the box perturbation method to simulation of complex terrain winds.

REFERENCES

[1] E. Balaras. Modeling complex boundaries using an external force field on fixed cartesian grids in large-eddy simulations. *Computers & Fluids*, 33(3):375–404, 2004.

[2] Boise State University. The kestrel CPU/GPU cluster — boise state university wiki. `http://wiki.boisestate.edu/wiki/the-kestral-cpugpu-cluster/`, 2015. Accessed: 2015-07-02.

[3] Jung-Il Choi, Roshan C Oberoi, Jack R Edwards, and Jacky A Rosati. An immersed boundary method for complex incompressible flows. *Journal of Computational Physics*, 224(2):757–784, 2007.

[4] A. J. Chorin. Numerical solution of the navier-stokes equations. *Mathematics of computation*, 22(104):745–762, 1968.

[5] A. de la Torre, V. Daniel, R. Tailleux, and H. Teitelbaum. A deep convection event above the Tunuyan valley near the Andes Mountains. *Monthly Weather Review*, 132:2259–2267, 2004.

[6] J. W. Deardorff. A numerical study of three-dimensional turbulent channel flow at large reynolds numbers. *Journal of Fluid Mechanics*, 41(02):453–480, 1970.

[7] R. DeLeon, K. Felzien, and I. Senocak. Toward a gpu-accelerated immersed boundary method for wind forecasting over complex terrain. In *ASME 2012 Fluids Engineering Division Summer Meeting*, volume 1, pages 8–12, 2012.

[8] R. DeLeon, D. Jacobsen, and I. Senocak. Large-eddy simulations of turbulent incompressible flows on gpu clusters. *Computing in Science & Engineering*, 15(1):26–33, 2013.

[9] E. A. Fadlun, R. Verzicco, P. Orlandi, and J. Mohd-Yusof. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics*, 161(1):35–60, 2000.

[10] E. Fedorovich and A. Shapiro. Structure of numerically simulated katabatic and anabatic flows along steep slopes. *Acta Geophysica*, 57(4):981–1010, 2009.

[11] H. J. S. Fernando. Fluid dynamics of urban atmospheres in complex terrain. *Annual Review of Fluid Mechanics*, 42:365–389, 2010.

[12] J. H. Ferziger and M. Perić. *Computational methods for fluid dynamics*. Springer Science & Business Media, 2012.

[13] T. Gao, Y. Tseng, and X. Lu. An improved hybrid cartesian/immersed boundary method for fluid–solid flows. *International Journal for Numerical Methods in Fluids*, 55(12):1189–1211, 2007.

[14] Gavrichenkov, I. Nvidia: Graphics chips with 20TFLOPS dp performance needed for ExaFLOPS supercomputers — x-bit labs. `http://www.xbitlabs.com/news/graphics/display/20101124175100_Nvidia_Graphics_Chips_with_20TFLOPS_DP_Performance_Needed_for_ExaFLOPS_Supercomputers.html`, 2010. Accessed: 2013-02-04.

[15] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, 1991.

[16] R. Ghias, R. Mittal, and H. Dong. A sharp interface immersed boundary method for compressible viscous flows. *Journal of Computational Physics*, 225(1):528–553, 2007.

[17] A. Gilmanov and S. Acharya. A hybrid immersed boundary and material point method for simulating 3D fluidstructure interaction problems. *International Journal for Numerical Methods in Fluids*, 56(12):2151–2177, 2008.

[18] A. Gilmanov and F. Sotiropoulos. A hybrid Cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies. *Journal of Computational Physics*, 207(2):457 – 492, 2005.

[19] A. Gilmanov, F. Sotiropoulos, and E. Balaras. A general reconstruction algorithm for simulating flows with complex 3D immersed boundaries on Cartesian grids. *Journal of Computational Physics*, 191(2):660 – 669, 2003.

[20] M. Griebel, T. Dornsheifer, and T. Neunhoeffer. *Numerical Simulation in Fluid Dynamics: A Practical Introduction*. Monographs on Mathematical Modeling and Computation. Society for Industrial and Applied Mathematics, 1998.

[21] D. A. Jacobsen and I. Senocak. A full-depth amalgamated parallel 3d geometric multigrid solver for gpu clusters. In *49th AIAA Aerospace Science Meeting*, 2011.

[22] D. A. Jacobsen and I. Senocak. Multi-level parallelism for incompressible flow computations on gpu clusters. *Parallel Computing*, 39(1):1–20, 2013.

[23] D. A. Jacobsen, J. C. Thibault, and I. Senocak. An MPI-CUDA implementation for massively parallel incompressible flow computations on multi-GPU clusters. In *49th AIAA Aerospace Science Meeting*, 2010.

[24] A. Keating, U. Piomelli, E. Balaras, and H. Kaltenbach. A priori and a posteriori tests of inflow conditions for large-eddy simulation. *Physics of Fluids (1994-present)*, 16(12):4696–4712, 2004.

[25] P. K. Kundu, I. M. Cohen, and D. R. Dowling. *Fluid Mechanics*. Elsevier Science, 2011.

[26] T. S. Lund, X. Wu, and K. D. Squires. Generation of turbulent inflow data for spatially-developing boundary layer simulations. *Journal of Computational Physics*, 140(2):233–258, 1998.

[27] K. A. Lundquist, F. K. Chow, and J. K. Lundquist. An immersed boundary method for the weather research and forecasting model. *Monthly Weather Review*, 138(3):796–817, 2010.

[28] M. Marquis, J. Wilczak, M. Ahlstrom, J. Sharp, A. Stern, J. C. Smith, and S. Calvert. Forecasting the wind to reach significant penetration levels of wind energy. *Bulletin of the American Meteorological Society*, 92(9):1159–1171, 2011.

[29] R. Mittal, H. Dong, M. Bozkurttas, F. M. Najjar, A. Vargas, and A. von Loebbecke. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. *Journal of computational physics*, 227(10):4825–4852, 2008.

[30] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37:239–261, 2005.

[31] J. Mohd-Yusof. Combined immersed-boundary/b-spline methods for simulations of flow in complex geometries. *CTR Annual Research Briefs, Center for Turbulence Research, NASA Ames Research Center/ Stanford University*, pages 317–327, 1997.

[32] J. M. Moran and M. D. Morgan. *Meteorology: the Atmosphere and the Science of Weather*. Macmillan College Publishing Company, 4th ed. edition, 1994.

[33] R. D. Moser, J. Kim, and N. N. Mansour. Direct numerical simulation of turbulent channel flow up to $\mathrm{Re}_\tau = 590$. *Physics of Fluids*, 11(4):943–945, 1999.

[34] D. Muñoz-Esparza, B. Kosović, J. Mirocha, and J. van Beeck. Bridging the transition from mesoscale to microscale turbulence in numerical weather prediction models. *Boundary-Layer Meteorology*, 153(3):409–440, 2014.

[35] D. Muñoz-Esparza, B. Kosović, J. van Beeck, and J. Mirocha. Erratum:a stochastic perturbation method to generate inflow turbulence in large-eddy simulation models: Application to neutrally stratified atmospheric boundary layers[phys. fluids 27, 035102 (2015)]. *Physics of Fluids (1994-present)*, 27(3):039901, 2015.

[36] D. Muñoz-Esparza, B. Kosović, J. van Beeck, and J. Mirocha. A stochastic perturbation method to generate inflow turbulence in large-eddy simulation models: Application to neutrally stratified atmospheric boundary layers. *Physics of Fluids (1994-present)*, 27(3):035102, 2015.

[37] OpenFOAM Foundation. OpenFOAM — the OpenFOAM foundation. `http://www.openfoam.org/`, 2015. Accessed: 2015-07-02.

[38] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips. GPU computing. *Proceedings of the IEEE*, 96(5):879 –899, May 2008.

[39] C. S. Peskin. Flow patterns around heart valves: a numerical method. *Journal of computational physics*, 10(2):252–271, 1972.

[40] S. B. Pope. *Turbulent flows*. Cambridge university press, 2000.

[41] L. Prandtl. *Führer durch die Strömungslehre*. Vieweg und Sohn, 1942.

[42] P. Queney. The problem of the airflow over mountains: a summary of theoretical studies. *Bulletin of the American Meteorological Society*, 29:16–26, 1948.

[43] J. Sanders and E. Kandrot. *CUDA by example: an introduction to general-purpose GPU programming*. Addison-Wesley Professional, 2010.

[44] I. Senocak, A. S. Ackerman, D. E. Stevens, and N. N. Mansour. Topography modeling in atmospheric flows using the immersed boundary method. Annual Research Briefs, Center for Turbulence Research, NASA-Ames/Stanford University, 2004.

[45] I. Senocak, M. Sandusky, R. DeLeon, D. Wade, K. Felzien, and M. Budnikova. An immersed boundary geometric preprocessor for arbitrarily complex terrain and geometry. *Journal of Atmospheric and Oceanic Technology*, (2015), 2015.

[46] A. Shapiro and E. Fedorovich. Unsteady convectively driven flow along a vertical plate immersed in a stably stratified fluid. *Journal of Fluid Mechanics*, 498:333–352, 2004.

[47] G. R. Tabor and M. H. Baba-Ahmadi. Inlet conditions for large eddy simulation: a review. *Computers & Fluids*, 39(4):553–567, 2010.

[48] J. C. Tannehill, D. Anderson, and R. H. Pletcher. *Computational Fluid Mechanics and Heat Transfer, Second Edition.* Series in Computational and Physical Processes in Mechanics and Thermal Sciences. Taylor & Francis, 1997.

[49] J. C. Thibault and I. Senocak. Accelerating incompressible flow computations with a Pthreads-CUDA implementation on small-footprint multi-GPU platforms. *The Journal of Supercomputing*, 59:693–719, 2012.

[50] TOP500 Supercomputing Site. Top 10 sites for November 2014. `http://www.top500.org`, 2015. Accessed: 2015-03-16.

[51] U. S. DoE. Funding opportunity announcement for wind forecasting improvement project in complex terrain — Department of Energy. `http://energy.gov`, 2014. Accessed: 2014-05-01.

[52] U. S. DoE. Wind vision: a new era for wind power in the United States. `http://www.energy.gov/windvision`, 2015. Accessed: 2015-06-08.

[53] H. K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method.* Pearson Education, 2007.

[54] N. Wood. Wind flow over complex terrain: A historical perspective and the prospect for large-eddy modelling. *Boundary-Layer Meteorology*, 96:11–32, 2000.