Multi-Scale Localized Perturbation Method for Geophysical Fluid Flows

Erik T. Higgins

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Masters of Science
in
Aerospace Engineering

Eric G. Paterson, Chair
Jonathan Pitt, Co-Chair
Heng Xiao

August 7, 2020
Blacksburg, Virginia

Keywords: computational fluid dynamics, non-linear interaction, multi-scale modeling, OpenFOAM, ocean physics

Multi-Scale Localized Perturbation Method for Geophysical Fluid Flows

Erik T. Higgins

(ABSTRACT)

An alternative formulation of the governing equations of a dynamical system, called the multi-scale localized perturbation method, is introduced and derived for the purpose of solving complex geophysical flow problems. Simulation variables are decomposed into background and perturbation components, then assumptions are made about the evolution of these components within the context of an environmental flow in order to close the system. Once closed, the original governing equations become a set of one-way coupled governing equations called the "delta form" of the governing equations for short, with one equation describing the evolution of the background component and the other describing the evolution of the perturbation component. One-way interaction which arises due to non-linearity in the original differential equations appears in this second equation, allowing the background fields to influence the evolution of a perturbation. Several solution methods for this system of equations are then proposed. Advantages of the delta form include the ability to specify a complex, temporally- and spatially-varying background field separate from a perturbation introduced into the system, including those created by natural or man-made sources, which enhances visualization of the perturbation as it evolves in time and space. The delta form is also shown to be a tool which can be used to simplify simulation setup. Implementation of the delta form of the incompressible URANS equations with turbulence model and scalar transport within OpenFOAM is then documented, followed by verification cases. A stratified wake collapse case in a domain containing a background shear layer is then presented, showing how complex internal gravity wave-shear layer interactions are retained and easily observed in spite of the variable decomposition. The multi-scale localized perturbation method shows promise for geophysical flow problems, particularly multi-scale simulation involving the interaction of large-scale natural flows with small-scale flows generated by man-made structures.

Multi-Scale Localized Perturbation Method for Geophysical Fluid Flows

Erik T. Higgins

(GENERAL AUDIENCE ABSTRACT)

Natural flows, such as those in our oceans and atmosphere, are seen everywhere and affect human life and structures to an amazing degree. Study of these complex flows requires special care be taken to ensure that mathematical equations correctly approximate them and that computers are programmed to correctly solve these equations. This is no different for researchers and engineers interested in studying how man-made flows, such as one generated by the wake of a plane, wind turbine, cruise ship, or sewage outflow pipe, interact with natural flows found around the world. These interactions may yield complex phenomena that may not otherwise be observed in the natural flows alone. The natural and artificial flows may also mix together, rendering it difficult to study just one of them. The multi-scale localized perturbation method is devised to aid in the simulation and study of the interactions between these natural and man-made flows. Well-known equations of fluid dynamics are modified so that the natural and man-made flows are separated and tracked independently, which gives researchers a clear view of the current state of a region of air or water all while retaining most, if not all, of the complex physics which may be of interest.

Once the multi-scale localized perturbation method is derived, its mathematical equations are then translated into code for OpenFOAM, an open-source software toolkit designed to simulate fluid flows. This code is then tested by running simulations to provide a sanity check and verify that the new form of the equations of fluid dynamics have been programmed correctly, then another, more complicated simulation is run to showcase the benefits of the multi-scale localized perturbation method. This simulation shows some of the complex fluid phenomena that may be seen in nature, yet through the multi-scale localized perturbation method, it is easy to view where the man-made flows end and where the natural flows begin. The complex interactions between the natural flow and the artificial flow are retained in spite of separating the flow into two parts, and setting up the simulation is simplified by this separation. Potential uses of the multi-scale localized perturbation method include multi-scale simulations, where researchers simulate natural flow over a large area of land or ocean, then use this simulation data for a second, small-scale simulation which covers an area within the large-scale simulation. An example of this would be simulating wind currents across a continent to find a potential location for a wind turbine farm, then zooming in on that location and finding the optimal spacing for wind turbines at this location while using the large-scale simulation data to provide realistic wind conditions at many different heights above the ground. Overall, the multi-scale localized perturbation method has the potential to be a powerful tool for researchers whose interest is flows in the ocean and atmosphere, and how these natural flows interact with flows created by artificial means.

Acknowledgments

I would like to acknowledge my advisors, Dr. Eric Paterson, Dr. Jonathan Pitt, and Dr. Heng Xiao for their insights into the fluid dynamics as a whole and their time, energy, and guidance to help turn this work into a reality. I would also like to thank Dr. John Gilbert, Dr. Matt Jones, Dylan Wall, Christian Martin, and Ryan Somero for their helpful discussions on OpenFOAM's ins-and-outs. I would also like to thank the attendees of the 15th OpenFOAM Workshop for their thought-provoking questions and comments on an early form of the material presented in this thesis. I would like to acknowledge Advanced Research Computing at Virginia Tech for providing computational resources and technical support that have contributed to the results reported within this thesis. URL: http://www.arc.vt.edu

Contents

List of Figures

List of Tables

List of Symbols

List of symbols test

$(\cdot)_b$	Background component of an arbitrary simulation field $(\cdot)$
α	Haline contraction coefficient of the fluid
β	Thermal expansion coefficient of the fluid
$\delta(\cdot)$	Perturbation ("delta") component of an arbitrary simulation field $(\cdot)$
$\dot{\sigma}_{(\cdot)}$	Net generation (source) of an arbitrary simulation scalar field $(\cdot)$
ϵ	Specific dissipation of turbulent kinetic energy
$\kappa_{(\cdot)}$	Molecular diffusivity of a scalar field $(\cdot)$
$\kappa_{t,(\cdot)}$	Eddy (turbulent) diffusivity of an arbitrary simulation scalar field $(\cdot)$
ν	Molecular kinematic viscosity
ν_t	Eddy (turbulent) viscosity
ω	Intenal gravity wave angular frequency
ρ	Density
ρ_0	Constant reference density
$\sigma_k, \sigma_\epsilon$	Model constants for the $k-\epsilon$ model
$\underline{\underline{D}}$	Fluid deformation tensor
$\vec{g}$	Gravity vector
$\vec{U}$	Velocity vector
C	Arbitrary scalar field

$C_{1\epsilon}, C_{2\epsilon}, C_{3\epsilon}, C_\mu$ Coefficient for the $k-\epsilon$ model

D Initial wake diameter

F IGW forcing amplitude

Fr Froude number

G Buoyancy production term for the buoyant $k-\epsilon$ model

g Gravitational acceleration

k Specific turbulent kinetic energy

k_0 Initial wake centerline turbulent kinetic energy

k_x Intenal gravity wave horizontal wavenumber

Ke Keulegan number

$kRHS$ Right-hand side of the k equation from the buoyant $k-\epsilon$ model

N Brunt-Väisälä frequency

P Shear production term for the $k-\epsilon$ model

p Pressure

Re Reynolds number

Ri Richardson number

RS Reynolds Stress term, nominally written as $\nabla \cdot \left(\overline{u_i' u_j'}\right)$

S Salinity

S_{ref} Constant reference salinity

T Temperature

T_{ref} Constant reference temperature

U_0 Initial wake speed

U_D Streamwise velocity defect

U_{D0} Initial wake centerline velocity defect

ϵRHS Right-hand side of the ϵ equation from the buoyant $k-\epsilon$ model

Chapter 1

Introduction

As computing power grows, use of supercomputing clusters within science grows with it. What previously could only be determined through empirical or analytic expressions derived from the governing equations of a physical system can now be simulated on larger and larger scales. This is very apparent in fluid dynamics, as turbulence models enable prediction of turbulent flows in pipes and around vehicles, and beyond even to global-scale flow predictions such as those studied using global circulation models. Similarly, higher computing power allows fundamental phenomena such as turbulence microstructures to be studied and compared to experimental data. Simulations resolving different spatial and temporal scales may often be coupled together in what are called multi-scale simulations. These simulations allow information that would not otherwise be resolved in a given simulation to be interpolated or extrapolated from a simulation able to resolve the necessary scales. An example of this would be a small-scale simulation focusing on a region of wetlands which is used to study heat transfer between the ground and atmosphere under diurnal heating. Information from this small-scale simulation may be used to calibrate a temperature boundary condition in a larger-scale simulation, such as one used to predict weather conditions across an entire continent.

External influences in a limited-scale domain require additional considerations. In the simulation of land-based wind turbine farm, one may need to consider the upstream velocity, temperature, and turbulence time history and spatial variations to better estimate power output and component wear as a function of time of year. A simple approach would be to determine a curve fit of these quantities and impose this curve fit upon the upstream boundary of the domain as the inlet or freestream conditions. These upstream data would be sourced from either field measurements or a larger-scale simulation, the latter being an example of a potential multi-scale simulation use in the case where field measurements are sparse, difficult to measure, or expensive to obtain.

A limitation of this approach is that once this upstream data enters the domain, it evolves only with consideration to the conditions within the domain; wind in a simulation of a

wind turbine farm only evolves in the presence of the wind turbines, localized topography, and boundary conditions, and not of any topography or developments occurring outside of the domain unless there is a mathematical forcing function which models the effects of these phenomena. In other words, physical phenomena which occur over a larger spatial and temporal scale than the simulation can resolve is ignored, and in some cases this may be important to consider. Once a simulation is completed, again using the wind turbine farm as an example, it may be necessary to isolate the wind turbine wakes to refine and optimize wind turbine placement for the purpose of maximizing power generation. Isolating a wind turbine wake in otherwise-uniform wind currents is trivial, but with considerations for flow variations due to topography and any aforementioned large-scale phenomena, this can become difficult, potentially requiring multiple simulations to determine.

The multi-scale localized perturbation method has been developed as a tool to ease simulation setup and processing of geophysical simulations, particularly those which require initial or inlet condition from larger-scale simulations. This is achieved through a decomposition of simulation variables into background and perturbation components, then key assumptions about the characteristics of these components allow the system to be closed. Careful derivation of these modified governing equations is documented, and several solution methods are then introduced to give the user flexibility in the solution of these separate components, then verification and demonstration cases are documented. These cases show how large-scale features that may not be resolved with a small-scale simulation domain can be imposed upon this domain such that flow within the domain may interact with these features. This allows a user to simplify problem setup with minimal loss of complex physics which the user may be interested in. The perturbation component, including the influence of any non-linear interaction between the perturbation and background, remains separated from the background field at all times, allowing enhanced visualization of the evolution of the perturbation which may aid in the post-processing of the simulation. Implementation of the multi-scale localized perturbation equations into OpenFOAM, an open-source computational fluid dynamics software package, is then shown, allowing users to adapt this method for other problems and equations of interest. Overall, this method is shown to be useful for geophysical problems, particularly those involving disparate spatial and temporal scales. Application and extension of this method to multi-scale simulations, including those where a small-scale simulation is one-way coupled to a large-scale simulation, is possible. Further modifications to the multi-scale localized perturbation method could allow for the background and perturbation components to have a greater degree of scale similarity than that which is assumed in the present derivation. This may better capture the interaction of background and perturbation turbulence, as significant scale overlap exists between these fields. One possible phenomenon whose study may be enhanced under the multi-scale localized perturbation method is wave-wave interactions between background internal gravity waves in a fluid, perhaps those generated by meso-scale forcing such as tidally-driven flow over an uneven topography, and internal gravity waves generated by a localized perturbation. Other potential applications of this method include the study of wave-shear layer interactions, including critical level interactions, and Kelvin-Helmholtz instabilities.

Chapter 2

Literature Review

Developments in our understanding of fluid dynamics have occurred since ancient times, but important steps in the development of the formulae used in this thesis trace their lineage back to the analytic solutions of fluid dynamics. With the aid of computers, the field of computational fluid dynamics (CFD) began to flourish and with it researchers gained the ability to numerically solve the governing equations of fluid flows and process enormous amounts of data from both experiments, field studies, and numerical simulations. These technologies together have provided a basis for modern science in fluid dynamics and environmental modeling and simulation.

2.1 Stratified Fluid Dynamics

The presence of density stratification introduces non-negligible, spatially-varying gravitational forcing which greatly impact fluid dynamics. Under a statically-stable stratification, that is, one with a density which increases as you descend with gravity, gravity acts as a restoring force, maintaining the stratification outside of any vigorous mixing or otherwise vertical motion of the fluid. Newton's second law can be used to approximate the impact of this restoring force on a packet of fluid neglecting any other forces. The natural frequency of this motion is termed the Brunt-Väisälä frequency, or BV frequency, and is typically represented in text as N. The equation for the BV frequency is given in eq. (2.1) below.

$$N^2 = \frac{g}{\rho_0}\frac{\partial \rho}{\partial z} \tag{2.1}$$

The value of N^2 is positive for any stable stratification and negative for an unstable one, giving a packet of fluid in a statically-stable fluid a real natural frequency, corresponding to oscillatory motion, and a packet of fluid in a statically-unstable fluid an imaginary nat-

ural frequency, corresponding to run-away motion. The oscillatory motion of a fluid in a statically-stable stratification creates a wave-like motion which can be described using dispersion relations which relates oscillations in space to oscillations in time. These wave are called internal gravity waves if they exist within a continuous density stratification or interfacial waves if they exist along a density discontinuity such as the air-water interface at the surface of a body of water.

Beyond this simple restoration force, density stratification results in non-linear instabilities. The presence of a statically-unstable fluid column will immediately result in mixing as currents that result from convective instabilities form, easily seen when a pot of water heats up on a stove. Even in the presence of a statically-stable stratification, density differences can yield shear instabilities, where fluid "rolls up" then breaks into turbulence. Internal gravity waves and shear currents also show unique interactions. Galmiche, Thual, and Bonneton (1997) simulates direct numerical simulation (DNS) of internal gravity waves in a shear current and found that the internal gravity waves enhance the kinetic energy of the shear current, which is also seen by Javam, Imberger, and Armfield (2000a).

Several important non-dimensional numbers in stratified fluid dynamics includes Reynolds numbers Re, Richardson number Ri, Froude number Fr, and buoyancy time Nt. The first is common in non-geophysical flow analyses, including industrial pipe flow and vehicle aerodynamics, and is the ratio of inertial forces to viscous forces. At high Reynolds numbers, flow is typically turbulent throughout, and as a Reynolds number increases, the range of scales of motion in the fluid generally increases. This is the result of the presence of more kinetic energy in the system which cascades down to smaller and smaller scales before being dissipated by viscous forces.

Richardson number is the ratio of buoyancy to shear and is defined as in eq. (2.3). A region of flow with a Richardson number less than 1/4 will form Kelvin-Helmholtz instabilities (shear instabilities), which remain until Richardson number returns about 1; a region of fluid where the Richardson number is less than 0 is statically unstable (Woods 1969; Abarbanel et al. 1984). Froude number is the ratio of inertial forces to gravitational forces, usually achieved by comparing velocity to a velocity derived from gravity. There are several definitions of Froude number depending on whether the user is interested in ocean surface vehicle dynamics, ocean surface wave dynamics, or internal geophysical flows far from fluid interfaces, with the internal geophysical flow variant given in eq. (2.4). Finally, the buoyancy time Nt is the scaling of a dimensional time by the BV frequency, and is a common metric in stratified fluid dynamics. Far away from any external forcing or highly turbulence patches, gravitational forcing becomes the dominant force and so a time period on the order of $1/N$ becomes a reasonable scaling for time.

$$Re \equiv \frac{velocity \times lengthscale}{kinematicviscosity} \tag{2.2}$$

$$Ri \equiv \frac{N^2}{(rate\ of\ vertical\ shear)^2} = \left(\frac{shear\ time\ scale}{buoyancy\ time\ scale}\right)^2 \tag{2.3}$$

$$Fr \equiv \frac{velocity}{length\ scale \times N} \tag{2.4}$$

2.1.1 Stratified Turbulence

Just as the presence of density stratification creates complexity in macroscopic fluid dynamics, microscopic turbulent motions also see the influence of gravity's restorative forcing under a large range of conditions. Kinetic energy from the largest characteristic scales "cascades" downwards to smaller and smaller scales until it is dissipated into heat by viscosity. A highly turbulent patch will see little influence of buoyancy, and has characteristics similar to that of isotropic turbulence (Gargett, Osborn, and Nasmyth 1984). In this case, buoyancy only affects the scales far larger than the patch, but as turbulence decays buoyancy begins to affect smaller scales. Turbulent fluctuations in the vertical direction are suppressed, increasing turbulence anisotropy, and rather than causing overturning, vertical velocity fluctuations in regimes affected by buoyancy will produce internal gravity waves. Further turbulence decay sees buoyancy affecting increasingly-small scales until vertical turbulent motions are fully suppressed (Hopfinger 1987; Stillinger, Helland, and Van Atta 1983). One of the largest consequences of this buoyancy-induced anisotropy is that many popular RANS turbulence models will not work for the majority of the flow as they assume isotropic turbulence. This includes the $k-\epsilon$ and the $k-\omega$ turbulence models. Reynolds stress models, including algebraic variants, where anisotropic Reynolds stress components can be accounted for are a potential alternative, however they are more computationally expensive with more equations to solve numerically.

2.2 Ocean Modeling

Stratified and dynamic environments, such as the ocean and atmosphere, are rich with complex physics. These processes have been studied for millennia, but only in recent centuries have scientists attempted to quantify them using mathematical models which allows for greater understanding, and potentially simulation and predictive capabilities. While models for processes such as diffusion are well-known and can be easily developed on their own, the ocean as a whole sees many overlapping and interacting processes. Indeed it is the interaction of all these processes that leads to the dynamic ocean that covers the majority of our planet. The influence of the ocean on the atmosphere is extensive, seeing gaseous species being

exchanged between the ocean surface and the atmosphere including greenhouse gases (Bigg et al. 2003). Vellinga and Wood (2002) simulate the loss of the thermohaline circulation in the Atlantic ocean and found global-scale changes to atmospheric surface temperature and precipitation. Overall, improvements to ocean modeling can aid in modeling and simulation of global climates, weather, and biological processes.

Internal gravity waves (IGWs) are a common phenomenon in the ocean as well as the atmosphere. Small vertical perturbations to a density stratification lead to motion due to the restoring force of stratification, leading to the propagation of this density perturbation in the form of a wave. The strength of the stratification and the characteristics of the wave itself drives the dynamics and trajectory of these waves. Origins of IGWs are numerous and include tidally-driven flows over topography as well as wave-wave interactions and turbulence decay processes (Sarkar and Scotti 2017; Müller et al. 1986; Stillinger, Helland, and Van Atta 1983). Their impact is widespread so proper simulation is important. Given that they arise from a perturbation to density, IGWs are a non-hydrostatic phenomenon so hydrostatic codes are unable to resolve them which may limit the codes available to a researcher who would like to study IGWs and their impact on flows at large (Marshall et al. 1997b). Similar concessions need to be made for a grid resolution which will resolve the IGW wavelengths of importance for the given flow and conditions. Non-linear wave-wave interaction is a major consequence of the ubiquitousness of IGWs, and modeling and simulation of this is a major test for any high-fidelity oceanographic or atmospheric CFD code.

Wave-wave interactions and internal wave breaking have been studied not just for their inherent scientific value but also for their role in oceanic mixing and the global energy balance. Breaking waves in particular are prone to turbulence production and transferring energy from the wave to smaller scales (Staquet and Sommeria 2002). Successful direct numerical simulations (DNS) of wave-wave interactions include the reflection of IGWs off of a reflecting surface, whether the surface normal to gravity or sloped (Chalamalla et al. 2013; Zhou and Diamessis 2013; Slinn and Riley 1996). Breaking IGWs, often spurned by wave-wave interactions, leads to an increase in turbulence and mixing which can alter the energy balance in a localize region of the ocean, but on large-scale may see the transfer of energy between processes including tidally-forced flow. DNS, and to an extent large eddy simulations (LES), can resolve the turbulent motions that result from IGW breaking, as seen in Dörnbrack, Gerz, and Schumann (1995), but further work is required to model breaking IGWs in RANS simulations. In spite of the inherent lack of turbulent eddy resolution by RANS simulations, researchers have devised models which attempt to improve turbulent mixing effects. Gent and McWilliams (1990) derived a formulation of isopycnal mixing for non-eddy resolving codes. Klymak and Legg (2010) attempted to remedy this in an oceanographic global circulation model by modifying eddy viscosity parameterization to better model turbulence and mixing due to overturning.

Table 2.1: Overview of GCMs reviewed in this thesis.

Model	Non-hydrostatic	Unstructured mesh	Citation
POM	No	No	(Blumberg and Mellor 1987)
ROMS	No	No	(Shchepetkin and McWilliams 2005)
SUNTANS	Yes	Yes	(Fringer, Gerritsen, and Street 2006)
MITgcm	Yes	No	(Marshall et al. 1997a)

2.2.1 General Circulation Model Codes

Many fluid dynamic codes, called general circulation models (GCM), have been created to simulate atmospheric and oceanographic flows and processes on a variety of scales, from basin-scale to global-scale. A common feature of these codes is the ability to include real bathymetry in the simulation to capture submerged ridges and mounts. Turbulence can be treated through either an explicit turbulence or subgrid-scale model, or implicitly through the discretization of the Laplacian operator. In addition, many codes use a finite volume formulation, and many have extension which allow them to perform the simultaneous or coupled simulations of the atmosphere and free surface. Models can have either a hydrostatic or non-hydrostatic formulation, which affects which flows the models can simulate but in either case large-scale currents may be resolved (Marshall et al. 1997b). Several well-known GCMs are introduced below, but this list is far from exhaustive. An overview of these GCMs is given in table 2.1.

Princeton Ocean Model

Princeton Ocean Model (POM) is a hydrostatic coastal circulation model from the 1980's, featuring a free surface boundary condition and terrain-following gridding that allows it to accurately simulate coastal flows using a finite difference formulation (Blumberg and Mellor 1987). Temperature and salinity transport equations and an equation of state aid in simulating a realistic ocean. A variant of the Mellor-Yamada turbulence model, which is designed for geophysical flows, brings the computational efficiency of RANS to the GCM (Mellor and Yamada 1982). Several derivative models have been developed, including variants which use finite volume formulations and allow for the use of unstructured grids (Chen, Liu, and Beardsley 2003).

Regional Ocean Modeling System

Regional Ocean Modeling System (ROMS) is another general circulation model with a focus on regional and coastal circulation, similar to POM. The model uses a finite volume,

hydrostatic formulation with mathematical and discretization details presented in papers by Shchepetkin and McWilliams (2003) and Shchepetkin and McWilliams (2005). The model is used to simulate coastal flow including upwelling, and favorable comparisons to data are made (Marchesiello, McWilliams, and Shchepetkin 2003).

Stanford Unstructured Nonhydrostatic Terrain-following Adaptive Navier–Stokes Simulator

Stanford Unstructured Nonhydrostatic Terrain-following Adaptive Navier–Stokes Simulator (SUNTANS) is a non-hydrostatic GCM which, unlike the aforementioned models, uses an unstructured grid to discretize the domain, allowing greater flexibility in bathymetry and the ability to simulate smaller-scale phenomena including internal gravity waves. Scalar transport equations and separate horizontal and vertical eddy viscosities and diffusivities allow careful consideration for mixing and double diffusion (Fringer, Gerritsen, and Street 2006).

MIT General Circulation Model

MIT General Circulation Model (MITgcm) is another ocean circulation model, but with greater emphasis on generality to treat flow over a variety of terrain and scale, as well as being able to resolve non-hydrostatic motion (Marshall et al. 1997b; Marshall et al. 1997a; Adcroft et al. 2004). The finite volume approach, hydrostatic and non-hydrostatic formulation options, and general height-based vertical coordinate makes it attractive to simulating not just coastal flows but also global-scale flows.

Code Comparisons in Literature

Comprehensive comparisons of codes are present in the literature. Dutay et al. (2002) sees the cooperation of many teams from around the world to compare simulation results of chlorofluorocarbon concentration in the ocean. Later studies compared GCM models for global mixed layer depths, thermohaline transport, and sea-ice over a variety of periods ranging from months to decades (Doney et al. 2004; Danabasoglu et al. 2014; Danabasoglu et al. 2016). Such code comparisons require extensive coordination and experience to organize, but they are important to examine how each model performs as well as the advantages and disadvantages for each model. Atmospheric and oceanographic data is sparse, but is necessary for model validation.

2.2.2 Multi-Scale Fluid Simulations

The ability to couple or otherwise link meso-scale simulations with small-scale simulations provides an interesting capability to researchers and analysts. Multi-scale simulations are often used for weather and climate simulations, but are also seen in many other branches of science and engineering as a survey by Groen, Zasada, and Coveney (2014) shows. Multi-scale simulations allow for information to propagate not just from the large-scale to the small-scale, but also in the inverse direction as well as the simultaneous motion of information across all scales.

Multi-scale simulations often take the form of a simulation whose goal is to resolve multiple scales of motion where the researcher is primarily interested in the propagation of information from the large-scales downward. This approach is common in atmospheric simulations where a large-scale simulation provides initial and boundary conditions for a smaller scale simulation whose domain is located entirely within the large-scale simulation domain. This approach is commonly referred to as nesting, and allows for researchers to resolve small scales only where it is necessary which improves computational efficiency of the system. Nesting can be seen in the multi-physics simulation of atmospheric transport of chemical species by Tang et al. (2004). Wiersema, Lundquist, and Chow (2020) use a similar approach to model realistic flow around urban buildings to track tracers and compare to field measurements.

Multi-scale modeling and simulations do not solely concern the movement of data from a meso-scale domain to a small-scale or micro-scale domain, but also from the small-scale upwards to the large-scale. Sumner, Watters, and Masson (2010) reviewed approaches for multi-scale modeling for CFD of wind turbine farms, primarily looking into the use of small-scale simulations focusing on the turbulent wakes of individual turbine blades, then analyzing how multiple wind turbine wakes interact with each other in a realistic flow over a desired topography in order to better position wind turbines within a farm as well as predicting wear on blades.

Other approaches to multi-scale modeling includes more active coupling between simulations or two-way nesting. Colella et al. (2011) use multi-scale modeling for the simulation of a fire in a ventilation system, one which is large enough that a full-scale 3D CFD simulation is impractical and where a 1D simulation is used for the large-scale pipe system simulation. In this manner, the small-scale simulation of the fire is conducted with fully-3D CFD and data from both the 1D model and 3D simulation are fed back into each other to complete the multi-scale system. This runs into an additional challenge of ensuring that data exchanged between solvers of different physics is consistent with the recipient simulation's governing equations. A dedicated multi-scale code called SOMAR-LES which uses adaptive mesh refinement to selectively refine regions of the domain where finer details such as highly turbulent structures may need to be resolved (Chalamalla et al. 2017). The finest regions are treated using an LES approach with a subgrid-scale model to improve turbulence resolution, and the authors show that this approach works well compared to benchmark cases with an advantage of saving computational time compared to a uniformly fine mesh. In this approach, multiple

scales are resolved efficiently while still enabling information transfer across scales.

2.3 Near-Wakes of Towed and Self-Propelled Bodies

As a solid body moves through a different-speed fluid, or as a fluid flows around a stationary solid body, momentum is transferred between the flow and body. This transfer manifests itself on the body as a drag force and on the fluid as a wake, where the flow immediately downstream of the body moves slower relative to the undisturbed fluid relative to the body. Wakes, and in particular turbulent wakes, have been of interests to fluid dynamicists and aerodynamic and hydrodynamic engineers for decades. Wind tunnel experiments by Naudascher (1965) looked at the wake of a self-propelled body in a flow represented by an axisymmetric plate with an axial jet in the center of it pointing downstream. This momentumless or net-zero momentum (NZM) wake was created by adjusting the thrust of the jet until the thrust produced equaled the drag on the body. Velocity and turbulent kinetic energy (TKE) distributions across the axisymmetric wake were measured, and a doubly-inflected velocity profile was found as a result of the combination of drag wake and thrust wake.

Similar wind tunnel experiments by Schetz and Jakubowski (1975) on an axisymmetric, unpropelled body and a similar axisymmetric, self-propelled body powered with an axial jet. These experiments show a similar qualitative wake structure to Naudascher, and that the propulsor increases TKE in the downstream wake increases with the inclusion of the jet.

Tennekes and Lumley completed a theoretical derivation of turbulent NZM wake characteristics. Their final scaling laws showed that the centerline velocity defect, the difference between the wake centerline velocity and the freestream velocity, of self-propelled wakes decay faster than that of pure drag wakes. Furthermore, these authors state that non-NZM wakes as defined by their velocity defect $U_D \equiv U - U_0$, including those of self-propelled vehicles undergoing a net acceleration, will decay slower than NZM wakes for a given length downstream of the source x, as seen in eqs. (2.5) and (2.6) (Tennekes and Lumley 1972). Hassid looked at these same NZM wakes through the eyes of a RANS formulation and found that the model was able to reproduce self-similarity parameters for the NZM wake as seen by several independent experiments (Hassid 1980b).

$$U_D \propto x^{-4/5} \qquad \text{(NZM (self-propelled) wake)} \tag{2.5}$$

$$U_D \propto x^{-2/3} \qquad \text{(drag wake)} \tag{2.6}$$

2.4 Life Cycle of a Stratified Wake

Stratified wakes are home to complex physics and dynamics which have made them an interest to scientists and researchers investigating fundamental physics. Lin and Pao conducted a review of early research into the evolution of stratified and unstratified wakes, as well as the general decay of stratified turbulence, from articles published up to 1979 (Lin and Pao 1979). These authors found that stratified wakes undergo a collapse after a period of time post-transit, characterized by a decrease in wake height and an increase in wake width. Internal gravity wave production is associated with this collapse, and the authors note that turbulence quantities decrease monotonically. Later authors have examined the wake life cycle more closely and have found that it is made up of several regimes based on general velocity, density, turbulence, and wake dimension patterns (Spedding 1997; Bonnier and Eiff 2002; Meunier, Diamessis, and Spedding 2006). This life cycle proceeds most generally as a 3D turbulence wake which undergoes collapse due to non-equilibrium forces, and eventual transformation into horizontal, quasi-2D motions, but others have found further sub-regimes in this sequence using tow-tank experiments. These life cycles are commonly defined in terms of a non-dimensional time Nt, which is physically significant far away from a solid body or extremely turbulent patch as buoyancy forcing will be the dominant force. An example of this lifecycle for a drag wake is parameterized by Spedding (1997) and is shown in fig. 2.1. Taking eq. (2.6) and assuming that distance downstream is normalized by the diameter D then taking $x/D \propto Fr(Nt)$, we find a stratified drag wake initially decays at the same rate as an unstratified one, yet at the first onset of buoyancy forces it begins to slow, which Spedding (1997) conjectures is due to potential energy generated by the object mixing up stratified fluid beginning to be converted to kinetic energy in the collapse. The final stage in the life of a stratified wake, the quasi-2D regime, sees the formation of large horizontal "pancake" vortices with no vertical motion. This is seen experimentally and numerically by many, and once again shows a departure from unstratified theory (Lin and Pao 1979; Spedding 1997; Diamessis, Spedding, and Domaradzki 2011).

Direct numerical simulation (DNS) of turbulent wake models capture this general evolution (Gourlay et al. 2001; Brucker and Sarkar 2010). Abdilghanie and Diamessis (2013) showed using DNS that IGW generation is different between high Reynolds number, here 100,000, and lower Reynolds number, showing the importance of Reynolds number for wake dynamics. LES by Dommermuth et al. (2002) provided similar results but for a higher Reynolds number than DNS. The importance of Reynolds number effects was corroborated by Diamessis, Spedding, and Domaradzki (2011) who found that higher Reynolds number affects late wake development in such a way that might make low Reynolds number scaling laws difficult to use for higher Reynolds numbers.

RANS simulations are much less common, perhaps owing to their greater emphasis on mathematical models to describe the effects of stratified turbulence. Hassid (1980a) provides an early NZM wake collapse simulation using a RANS turbulence model and replicates experimental data, and (Chernykh et al. 2012) were able to accomplish a similar goal using both

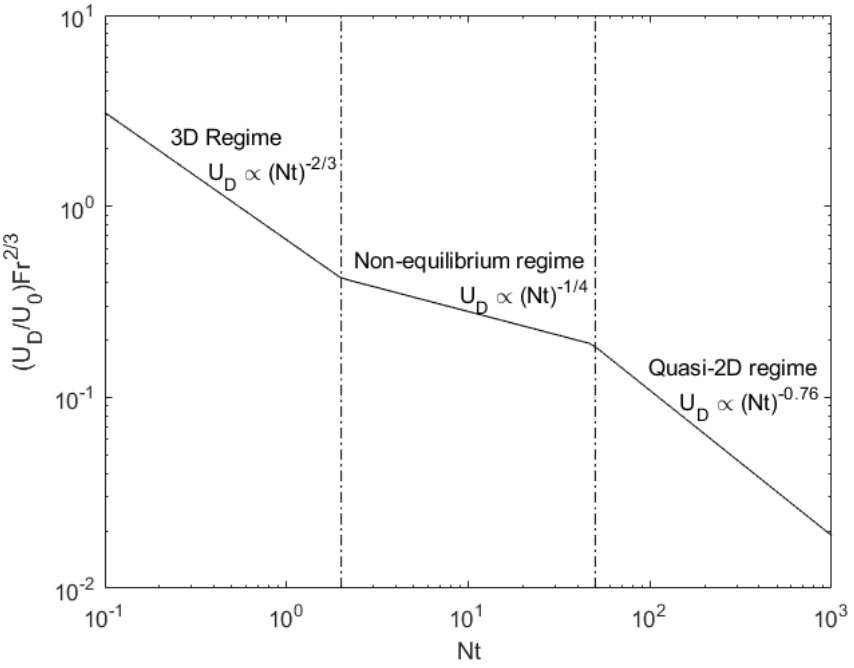

Figure 2.1: Life cycle of a stratified wake as described by Spedding (1997). The transition points are approximate.

RANS and DNS. Non-net zero momentum wakes were also examined by Chernykh, Moshkin, and Fomina (2009) as well as de Stadler and Sarkar (2012), with both groups finding that wake evolution was qualitatively similar to that of a NZM wake. Chernykh and Voropaeva (2015) conducted RANS simulations of a turbulent wake in a shear flow and found that the addition of ambient shear aided in preserving the wake far longer than a wake in a quiescent background, which showcases the importance of the inclusion of realistic background conditions in the study of stratified wakes in an active ocean or atmospheric environment.

While most of the aforementioned studies examine wakes in a simple density stratification, recent wake studies have utilized more complex and realistic equations of state and scalar transport equations to better simulate the ocean environment. Moody et al conducted a complex simulation, field study, and tow tank study of a stratified wake. Their computational simulations were aided by the use of MITgcm to simulate the fluid which includes a non-linear temperature and salinity stratification (Moody et al. 2017). Martin performs a complex parameter study of stratified wakes using MITgcm to study the effects of stratification, mixed layer depth, and initial wake conditions on late-wake characteristics (Martin 2016).

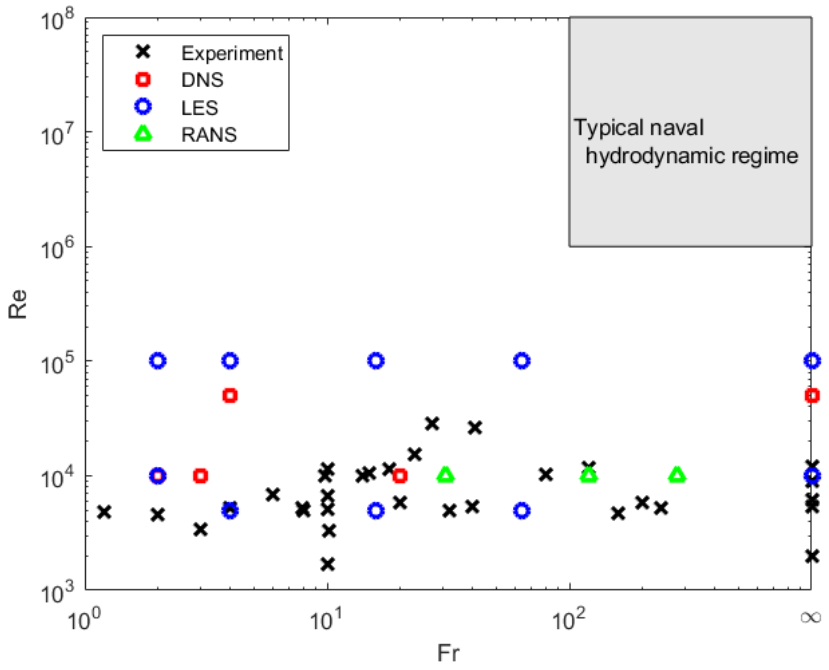

Figure 2.2: Froude number, Reynolds number combinations of select stratified wake cases.

2.5 High Re Limitations of Experiments and DNS

Turbulent wakes, as with many other fluid flows, shows a strong dependence on Reynolds number. Stratification leads to Froude number being another important quantity in describing and predicting stratified turbulent wakes characteristics. Meunier et al, for example, derive theoretical scaling laws for stratified wakes, most of which are functions of Reynolds number or Froude number (Meunier, Diamessis, and Spedding 2006). Figure 2.2 shows a non-exhaustive list of Reynolds number-Froude number pairs for tow-tank experiments and numerical simulations of stratified wakes, with sources and specific Reynolds number-Froude number pairs tabulated in table 2.2.

Experimental studies of stratified wakes typically on the order of 10^3 to 10^4 whereas engineering and environmental flows of interest may be on the order of 10^6 and higher (Spedding 2014). This constraint comes from inherent size limits on tow-tank facilities where controlled conditions must be maintained for a propagating wake such that it does not come under significant interference from the sides of the tank for its entire evolution. Diamessis et al found that for high Reynolds number wakes around $Re = 100,000$, wake collapse and the resulting turbulent motions may persist for thousands of buoyancy periods, so any experimental studies on the late-wake of these types of wakes may be extremely limited, if not impossible (Diamessis, Spedding, and Domaradzki 2011). Similarly, high Reynolds number turbulence requires a sufficiently grid resolution to capture all scales of motion which ties

Table 2.2: Experiments and numerical simulations shown in fig. 2.2.

Study	Method	Re range ($\times 10^3$)	Fr range
Abdilghanie and Diamessis (2013)	DNS	5, 100	4, 16, 64
Bonnier and Eiff (2002)	Exp.	2 - 11.5	3 - 10; ∞
Brucker and Sarkar (2010)	DNS	50	4, ∞
Chernykh et al. (2012)	DNS, RANS	10	31, 120, 280
Diamessis, Spedding, and Domaradzki (2011)	LES	5, 100	4, 16, 64
Dommermuth et al. (2002)	LES	10, 100	2, ∞
Meunier and Spedding (2004)	Exp.	5	8, 32
Meunier and Spedding (2006)	Exp.	5 - 32	6 - 40
Spedding, Browand, and Fincham (1996)	Exp.	1.7 - 10	1.2 - 10.2
Spedding (1997)	Exp.	5 - 12	10 - 240
de Stadler and Sarkar (2012)	DNS	10	3

the upper limit of Reynolds number to the computing power of the largest supercomputers of the day. The study of high Reynolds number turbulent wakes is possible using different approaches; LES resolves the largest energy-containing eddies and models the smallest eddies using a sub-grid model while RANS simulations model all scales of turbulent motion. RANS is popular with full-scale engineering flows making it a desirable method for the simulation of full-scale stratified wakes.

2.6 Summary of Contribution

The majority of studies in the literature focus on wakes in an unstratified or linearly-stratified fluid with limited background or ambient flows. There have been simulations conducted with simple ambient shear flows, but real ocean and atmospheric environments contain much more complicated density, velocity, and turbulence profiles, and the ability to prescribe non-trivial profiles, perhaps obtained from in-situ measurements, may aid a researcher in determining the cause of observed phenomena and calculating the impact of the stratified wake on the surrounding fluid at large. Interactions between the wake and the ambient may provide mechanisms for large-scale energy transfer and influence the development of the late-wake.

Many simulations are performed with a density transport equation or a simple linear equation of state for density. Lack of a more realistic equation of state which takes in further parameters such as temperature, salinity, and other scalars may inhibit the ability of a simulation to capture other phenomena of interest including double-diffusive phenomena. The

stratified experiments observed in this literature review featured either a linear temperature or salinity stratification as more complex profiles may be more difficult to produce and maintain.

This paper seeks to provide a formulation of the governing equations of stratified, incompressible flow to allow for the rapid study of engineering and environmental flows including stratified turbulent wakes. Attention is paid to turbulence models which will allow high Reynolds number flows to be simulated, giving applicability to previously-intractable simulations or cost-ineffective experiments. The ability to prescribe realistic background conditions allows researchers to consider far more factors than previously possible and pushes the envelope in terms of the range of environments and flows that may be considered. Clean separation of the ambient fields from the perturbation fields allow for easy comparison of simulations even in the case of dissimilar background fields, greatly improving the capabilities to perform case studies.

Chapter 3

Derivation of the Multi-scale Localized Perturbation Method

The multi-scale localized perturbation method was originally to simulate complex environmental flows, which are governed by non-linear partial differential equations. These environments may also have anthropogenic flows present as well. For example, a stream or lake with an industrial outflow pipe might generate complex flows where the turbulence inherent to the natural water features may want to be tracked and simulated separate from the turbulence contained in the outflow pipe as well as all turbulence generation that comes from the interaction of the outflow pipe flow and the ambient water flow. While the ultimate application of this thesis are for the interaction of ambient flows and human-generated flows, the fundamental principles of the multi-scale localized perturbation method may be applied to other natural phenomena which are governed by partial differential equations, particularly ones where non-linear interactions between the background and perturbation fields may be of interest.

3.1 Key Principles

The key principle of the multi-scale localized perturbation method is the decomposition of a simulation field, such as temperature or velocity, into two components: a background component, symbolized here by the subscript b, and a perturbation or "delta" field, preceded by the symbol δ. Given an arbitrary simulation variable ϕ which is a function of 3D space represented by the position vector $\vec{x}$ and the time variable t, we could write the decomposition of ϕ as

$$\phi\left(\vec{x},t\right)=\phi_b\left(\vec{x},t\right)+\delta\phi\left(\vec{x},t\right) \tag{3.1}$$

The specific quantity encapsulated by the background and perturbation components may be vary from problem to problem, but it is intended that the background component refers to quantities which result from processes outside of the simulation. In the case of an oceanographic simulation, the background velocity field $\vec{U}_b$ may refer to velocity components which originate due to natural processes including wind, tides, and Coriolis forces while the velocity perturbation components $\delta\vec{U}$ may arise from the passage of a vessel as well as the non-linear interaction between background components and ship-induced perturbations to this background.

While this decomposition is a linear one, when performed on non-linear differential equations, non-linearities may be produced when substituting the decomposed form into the differential equation. These non-linearities between background and perturbation components may allow a researcher to identify and examine how background and perturbation components interact to produce further perturbation components.

Let the function $PDE(\phi)$ represent the governing differential equation for an arbitrary simulation variables $\phi(\vec{x}, t)$ including any time or position derivatives of ϕ, then one may write the governing differential equation as

$$PDE(\phi) = 0 \tag{3.2}$$

By performing the decomposition, we rewrite the above equation as

$$PDE(\phi_b + \delta\phi) = 0 \tag{3.3}$$

However, doubling the number of variables while leaving the number of equations the same results in a need to close the system by introducing further equations. For this, we must introduce an assumption that the background fields and the perturbation fields enjoy a large degree of scale separation, both temporally and spatially. For many environmental flows in the ocean and atmosphere, we may assume that the most energetic components of the background internal wave field are at the largest of lengthscales and timescales, as reviewed by Garrett and Munk (1979). This may be up to the order of kilometers for the lengthscale, and up to tidal or inertial periods for the timescales. Similarly, we assume that the perturbation features are much more compact spatially and evolve more quickly in time. The wake of an airplane or ship, for example, may be assumed to be on the order of the vehicle size in width and with timescales on the order of the buoyancy period.

The introduction of this assumption has the following implication mathematically: over the timescale and lengthscales of the background quantities, we can assume that the perturbation evolves much faster and over a smaller area, and the influence of the perturbation on the background itself is negligible. Mathematically, it may be given that the governing differential equation may be written as a function of the background components only, and that this background-only equation is identical in form to the original governing differential

equation. That is,

$$PDE(\phi_b) = 0 \tag{3.4}$$

in addition to eq. (3.3). Equations (3.3) and (3.4) are general and always true. However, we may rewrite eq. (3.3) into a more revealing form. In general, governing differential equations may be non-linear, and non-linear components of eq. (3.2) may be represented by the function $NL(\phi_b, \delta\phi)$ as

$$PDE(\phi_b + \delta\phi) = PDE(\phi_b) + PDE(\delta\phi) + NL(\phi_b, \delta\phi) = 0 \tag{3.5}$$

where terms are grouped into a linear PDE based on ϕ_b only, a linear PDE which is a function of $\delta\phi$ only, then finally all non-linear components as a function of ϕ_b and $\delta\phi$. In the case that $PDE(\phi_b + \delta\phi)$ is linear, then $NL(\phi_b, \delta\phi) \equiv 0$. Likewise, if the non-linear interaction between ϕ_b and $\delta\phi$ is captured by $NL(\phi_b, \delta\phi)$ then this term is also 0 if at least one of its arguments is identically 0. Thus, we can rewrite eq. (3.3) as

$$PDE(\phi_b) + PDE(\delta\phi) + NL(\phi_b, \delta\phi) = 0 \tag{3.6}$$

Now, the governing equations of the decomposed system is given as

$$PDE(\phi_b) = 0 \tag{3.7}$$
$$PDE(\phi_b) + PDE(\delta\phi) + NL(\phi_b, \delta\phi) = 0 \tag{3.8}$$

Since there is redundant information in the second line, we may simplify this set of equations into a final form shown in eqs. (3.9) and (3.10). This is referred to as the "delta form" of the governing equations.

$$PDE(\phi_b) = 0 \tag{3.9}$$
$$PDE(\delta\phi) + NL(\phi_b, \delta\phi) = 0 \tag{3.10}$$

Alternatively, one may think of the decomposition in a slightly different formulation, one which is less revealing of the non-linear interactions between background and perturbation variables, but one that is closer to the actual process of the decomposition itself. This approach is closer to how the multi-scale localized perturbation method is implemented in chapter 4.

$$PDE(\phi_b) = 0 \tag{3.11}$$
$$PDE(\phi_b + \delta\phi) - PDE(\phi_b) = 0 \tag{3.12}$$

In either case, the first equation represents the evolution as defined by the governing partial differential equation of the background ϕ quantity in time and space independent of the perturbation components of ϕ. The second equation contains the dynamics of the perturbation components of ϕ by themselves as well as any non-linear interaction between the background components and perturbation components acting solely on the perturbation field itself. This may be thought of as one-way coupling between the background dynamics and perturbation dynamics. Since the number of equations was doubled in response to the number of unknowns being doubled, the system is closed and can be numerically solved so long as the initial un-decomposed system was closed and could be solved. Since the coupling is one-way, the background-only dynamics can be solved first before the perturbation and non-linear interaction dynamics, and any conventional method for solving partial differential equations, such as the PISO algorithm for computational fluid dynamics, can be used in solving these systems of equations.

The above process can be extended for a PDE with more than one variable. Take for instance, a set of governing PDEs which is a function of two variables ϕ and ψ.

$$PDE(\phi, \psi) = 0 \tag{3.13}$$

The same approach is taken with decomposing ϕ and ψ into background and perturbation components.

$$\phi = \phi_b + \delta\phi \tag{3.14}$$
$$\psi = \psi_b + \delta\psi \tag{3.15}$$

Substituting this decomposition yields the governing PDE

$$PDE(\phi_b + \delta\phi, \psi_b + \delta\psi) = 0 \tag{3.16}$$

Again, we can express this PDE in a different form, rather as a sum of a PDE which only depends on background variables, a PDE which only depends on perturbation variables, then finally a series of non-linear interaction terms NL which encompasses all of the non-linearity between all combinations of independent variables.

$$\begin{aligned} PDE(\phi_b + \delta\phi, \psi_b + \delta\psi) = PDE(\phi_b, \psi_b) + PDE(\delta\phi, \delta\psi) \\ + NL(\phi_b, \delta\phi) + NL(\phi_b, \delta\phi, \psi_b) + NL(\phi_b, \delta\phi, \psi_b, \delta\psi) \\ + NL(\psi_b, \delta\psi) + NL(\delta\phi, \psi_b) + NL(\phi_b, \delta\psi) + ... = 0 \end{aligned} \tag{3.17}$$

Following the same procedure as with the single variable decomposition, we assume that the background processes are scale-separated from those of the perturbations, allowing us to assume that the effect of the perturbation on the background is negligible. Therefore, we can derive an expression for the background-only processes in the form of a family of PDEs.

$$PDE(\phi_b, \psi_b) = 0 \tag{3.18}$$

This equation and eq. (3.17) form a set of 4 PDEs which, assuming that the original set of PDEs $PDE(\phi, \psi) = 0$ is closed, is also closed. Since there is a redundancy of information, we can subtract the former from the latter to retrieve the set of governing PDEs for the perturbation variables, plus any non-linear interaction between background and perturbation variables.

$$\begin{aligned} PDE(\delta\phi, \delta\psi) + NL(\phi_b, \delta\phi) + NL(\phi_b, \delta\phi, \psi_b) + NL(\phi_b, \delta\phi, \psi_b, \delta\psi) \\ + NL(\psi_b, \delta\psi) + NL(\delta\phi, \psi_b) + NL(\phi_b, \delta\psi) + ... = 0 \end{aligned} \tag{3.19}$$

3.1.1 Variations of Numerical Solutions

The ultimate goal of this decomposition is the ability for a researcher or engineer to rapidly facilitate parameter studies related to the transit of disturbances or vehicles within a complex, possibly time-resolved background. The researcher can modify the system as needed by the problem as hand. For example, a researcher looking into how atmospheric internal gravity waves affect the development and evolution of an airplane wake may want to fully simulate both the background fields and perturbation fields. In this case, the researcher would fully solve both sets of governing differential equations, first the background-only equations then the combined perturbation/non-linear interaction equations.

If a precursor simulation to determine the background simulation fields has already been performed, then only the perturbation/non-linear interaction equations may need to be solved, and the background fields may be spatially and temporally interpolated from the precursor simulation data. This has an advantage as it allows for the precursor and small-scale simulations to be of different scales. For example, the precursor simulation may encompass an entire mountain chain and the lee waves which form off of these mountains. As the internal gravity waves propagate and interact with each other and other ambient internal gravity

waves to produce turbulence. A large-scale simulation on the order of hundreds of kilometers in size may be used to simulate this turbulence development. Far downstream of the mountains and within the domain of this large-scale simulation, an airplane may be simulated and data from the large-scale simulation may be used as initial and boundary conditions for the small-scale simulation centered on the airplane. As the large-scale simulation data for several hours or days may be available, the ability to interpolate the large-scale simulation data onto the small-scale simulation domain allows for more accurate results, and the interaction of the background velocity, density, and turbulence quantities with those of the perturbation induced by the plane's passage may be examined within the framework of the multi-scale localized perturbation method. Data interpolated from the precursor simulation would need to be consistent with the governing differential equations of the small-scale simulation which adds an additional challenge, particularly if the precursor simulation is, for example, 2D or 1D while the small-scale simulation is not.

Furthermore, the simulation duration may be very short compared to the characteristic timescale of the background fields. In this case, the engineer or researcher may be able to hold the background fields constant in time to greatly speed up the simulation. In this case, only an instant of the precursor simulation is needed, and consistent mathematical expressions may then be used for the initial conditions in the background fields. The flexibility of the multi-scale localized perturbation method allows for the simulation to be adapted to the needs of the user, potentially allowing for a small sacrifice in time-resolution of the physics for a savings in computational time.

While not explored in this derivation, two-way coupling of background and perturbation components may be performed to improve results in cases where scale separation between the background and perturbation components is not sufficient to assume one-way coupling. This may be particularly useful for turbulence quantities, as scale overlap in background and perturbation turbulence may be extensive for geophysical or engineering flows. One potential implementation would include proportionally splitting the impact of the non-linear interaction terms between the background dynamical equations and the perturbation dynamical equations. For example, instead of the non-linear interaction terms $NL(\phi_b, \delta\phi)$ only appearing in the expansion of the perturbation dynamics, as they do in eq. (3.8), these terms may appear in both eqs. (3.7) and (3.8), but scaled with an empirical coefficient which potentially differs between the two equations. This would allow the perturbation to influence the evolution of the background.

3.2 Example Decompositions

Several example decompositions will be performed here to demonstrate the decomposition in a more concrete manner. These examples will mainly focus on partial differential equations related to fluid dynamics of stratified fluids, such as in the atmosphere or ocean.

3.2.1 Divergence-Free Fluid

Incompressible flows are common in environmental flows in nature. Incompressible flows are defined such that the fluid density does not change as a result of the fluid flow behavior, namely pressure fluctuations associated with the bulk motion of the fluid. Mathematically, incompressible flows are defined by the kinematic constraint that the divergence of velocity of the fluid is identically zero everywhere in the flow.

$$\nabla \cdot \vec{U} = 0 \tag{3.20}$$

If one assumes that the fluid flow in question may be decomposed into a background component and a perturbation component, the multi-scale localized perturbation method is applied to its fields. Here,

$$\vec{U} = \vec{U}_b + \delta\vec{U} \tag{3.21}$$

Making this substitution into eq. (3.20), and applying an elementary vector calculus identity analogous to distribution for the divergence term, yields

$$\nabla \cdot \vec{U}_b + \nabla \cdot \delta\vec{U} = 0 \tag{3.22}$$

Next, one assumes that the spatial and temporal scales of the background components are well-separated from those of the perturbation components, and that the perturbation components are small in scale such that they have a negligible impact on the evolution of the background dynamics. The background dynamics can then be modeled as separate of any perturbation, and the background-only dynamics can be determined by taking the combined equation in eq. (3.22) and setting all "delta" fields to zero. Therefore, the background-only equation is given as

$$\nabla \cdot \vec{U}_b = 0 \tag{3.23}$$

This implies that the background component of an incompressible fluid must itself be incompressible. Taking the two equations above and writing them together allows one to quantify the system completely.

$$\nabla \cdot \vec{U}_b = 0 \tag{3.24}$$

$$\nabla \cdot \vec{U}_b + \nabla \cdot \delta\vec{U} = 0 \tag{3.25}$$

Since the second equation contains redundant information already contained in the first equation, one may simplify the set of equations to its final form

$$\nabla \cdot \vec{U}_b = 0 \tag{3.26}$$

$$\nabla \cdot \delta\vec{U} = 0 \tag{3.27}$$

This implies that for an incompressible fluid whose variable fields are being decomposed under the multi-scale localized perturbation method, the background component and the perturbation component must individually be incompressible. Since divergence is a linear operator, no non-linear interaction between the background and perturbation fields is created by the multi-scale localized perturbation.

3.2.2 Turbulent Scalar Transport Equation

A more complex example can be performed on a scalar transport equation with turbulent diffusion for an incompressible. This governing equation, seen in eq. (3.28), contains a time derivative, a non-linear advection term, and a non-linear diffusion term as well as a general source term for a general passive scalar C. Total diffusion is given by a molecular diffusion κ_C and a turbulent diffusivity given by $\kappa_{t,C}$. The total source/sink term is given by a general term $\dot{\sigma}_C$.

$$\frac{\partial C}{\partial t} + \vec{U} \cdot \nabla C - \nabla \cdot ((\kappa_C + \kappa_{t,C})\nabla C) = \dot{\sigma}_C \tag{3.28}$$

Here, $\kappa_{t,C}$ is given by a constitutive relation from a turbulence model. For a model based on the eddy diffusivity model, this may be proportional to the turbulent eddy viscosity predicted by the flow turbulence model. This example allows for a further exploration of the multi-scale localized perturbation method through the inclusion of more variables as well as non-linearity in the governing equation. As with the example in the above section, one first begins by assuming that the fluid can be decomposed into background and perturbation components, given below.

$$\vec{U} = \vec{U}_b + \delta\vec{U} \tag{3.29}$$

$$C = C_b + \delta C \tag{3.30}$$

$$\kappa_{t,C} = \kappa_{tb,C} + \delta\kappa_{t,C} \tag{3.31}$$

$$\dot{\sigma}_C = \dot{\sigma}_{b,C} + \delta\dot{\sigma}_C \tag{3.32}$$

Making these substitutions into eq. (3.28) yields

$$\frac{\partial C_b}{\partial t} + \frac{\partial \delta C}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right) \cdot \nabla \left(C_b + \delta C\right) - \nabla \cdot \left((\kappa_C + \kappa_{tb,C} + \delta\kappa_{t,C}) \nabla \left(C_b + \delta C\right)\right) = \dot{\sigma}_{b,C} + \delta\dot{\sigma}_C \tag{3.33}$$

Again, one assumes the scale separation between the background and perturbation components, enough that the background fields can be assumed to evolve independently of the perturbation components over the length of the simulation. Thus, one can derive the governing equation for the background-only components by setting all "delta" fields to zero in the above equation.

$$\frac{\partial C_b}{\partial t} + \vec{U}_b \cdot \nabla C_b - \nabla \cdot \left((\kappa_C + \kappa_{tb,C}) \nabla C_b\right) = \dot{\sigma}_{b,C} \tag{3.34}$$

Equations (3.33) and (3.34) form the governing differential equations for the combined background-perturbation dynamics and the background-only dynamics, respectively. There is some redundant information in this combined system in that eq. (3.33) contains eq. (3.34), so one may subtract the latter from the former to eliminate this redundancy. This result yields

$$\frac{\partial \delta C}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right) \cdot \nabla \delta C - \nabla \cdot \left((\kappa_C + \kappa_{tb,C} + \delta\kappa_{t,C}) \nabla \delta C\right) = \delta\dot{\sigma}_C - \delta\vec{U} \cdot \nabla C_b + \nabla \cdot \left(\delta\kappa_{t,C} \nabla C_b\right) \tag{3.35}$$

One can see the appearance of two new terms on the right-hand side of the equation. These terms represent the non-linear interaction between the background fields and the perturbation fields, and how these interactions affect the dynamics of the "delta" fields. The first non-linear interaction term represents the advection of the background scalar field by the perturbation velocity field. Any background scalar advected by this perturbation velocity would move scalar from one area of the domain to another, and the change in the perturbation scalar field would reflect this change. Similarly, there may be a turbulent eddy diffusivity associated with the perturbation flow field which in turn would lead to an increase in diffusion of the background scalar field. The scalar perturbation field would reflect this increase in diffusion and the redistribution of C from one region of the domain to another.

A problem arises with this example, however. If the background scalar field is advected or diffused by perturbation processes, then one should expect the background scalar field quantity to decrease by an equal amount in order to maintain conservation of mass. Since the coupling between the background and perturbation dynamics is only one-way, changes in the perturbation fields are not necessarily reflected in the background field clearly violating the conservation of mass. In the separation of scales, one should expect however that perturbations are small in magnitude, thus the volume-integrated perturbation scalar should be very small relative to the volume-integrated background scalar. In this, one assumes that

any non-conformity to conservation of mass is small and that this small discrepancy is a trade-off due to the time benefits saved in the multi-scale localized perturbation method.

One may also notice that the molecular diffusion of the scalar C, given by κ_C in the example, is not decomposed. The multi-scale localized perturbation method does not handle the decomposition of physical constants such as this and others such as gravitational acceleration, since this would imply that the background and perturbation fields are completely separate substances rather than two idealistically-separated quantities of fluid or mass.

As outlined in section 3.1, the one-way coupled nature of the decomposition results in a natural solution order to the system of equations outlined by eqs. (3.34) and (3.35). The background-only dynamics only require information from the background fields while the perturbation field dynamics require information from both background and perturbation fields. Naturally, one may solve for the background fields first before solving for the perturbation fields at each timestep. If the fluid flow and scalar transport are solved in a segregated manner, such as in the PISO algorithm, one may solve for flow velocity and pressure for the perturbation fields before solving eq. (3.35) for C. In this case, the non-linear terms can be evaluated explicitly and treated as an explicit source term that may be easily handled by the linear equation solver of a computational fluid dynamics program.

3.3 Delta Form of Navier-Stokes Equations

The decomposition of the incompressible RANS equations begins *in medias res* from eqs. (3.36) and (3.37), where the Reynolds stress term is replaced with the form RS for the sake of space.

$$\nabla \cdot \vec{U} = 0 \tag{3.36}$$

$$\frac{\partial \vec{U}}{\partial t} + \vec{U} \cdot \nabla \vec{U} = -\nabla \left(\frac{p}{\rho_0} \right) + \nabla \cdot (2\nu \underline{D}) + RS + \frac{\rho}{\rho_0} \vec{g} \tag{3.37}$$

The decomposition of the divergence of velocity has already been shown, so here we will focus on the decomposition of the conservation of momentum. We begin the decomposition of the conservation of momentum in the same way, decomposing the simulation variables into background components and perturbation components.

$$\vec{U} = \vec{U}_b + \delta \vec{U} \tag{3.38}$$

$$p = p_b + \delta p \tag{3.39}$$

$$\underline{D} = \underline{D}_b + \delta \underline{D} \tag{3.40}$$

$$\rho = \rho_b + \delta \rho \tag{3.41}$$

$$RS = RS_b + \delta RS \tag{3.42}$$

Making these substitutions yields

$$\frac{\partial \vec{U}_b}{\partial t}+\frac{\partial \delta\vec{U}}{\partial t}+\left(\vec{U}_b+\delta\vec{U}\right)\cdot\nabla\left(\vec{U}_b+\delta\vec{U}\right)=-\nabla\left(\frac{p_b+\delta p}{\rho_0}\right)+\nabla\cdot\left(2\nu\left(\underline{D}_b+\delta\underline{D}\right)\right)+RS+\frac{\rho_b+\delta\rho}{\rho_0}\vec{g} \tag{3.43}$$

We assume that there is scale separation between the background and perturbation components, which allows us to assume that the perturbation components have little effect on the evolution of the background components. In this, the background is independent of the perturbation, and we can write the governing differential equation for the above equation simply by setting all "delta" fields to zero.

$$\frac{\partial \vec{U}_b}{\partial t}+\vec{U}_b\cdot\nabla\left(\vec{U}_b\right)=-\nabla\left(\frac{p_b}{\rho_0}\right)+\nabla\cdot\left(2\nu\underline{D}_b\right)+RS_b+\frac{\rho_b}{\rho_0}\vec{g} \tag{3.44}$$

Similarly, we can subtract eq. (3.44) from eq. (3.43) to remove redundant information to yield the governing differential equation for the perturbation velocity and pressure

$$\frac{\partial \delta\vec{U}}{\partial t}+\left(\vec{U}_b+\delta\vec{U}\right)\cdot\nabla\left(\delta\vec{U}\right)+\left(\delta\vec{U}\right)\cdot\nabla\left(\vec{U}_b\right)=-\nabla\left(\frac{\delta p}{\rho_0}\right)+\nabla\cdot\left(2\nu\delta\underline{D}\right)+\left(RS-RS_b\right)+\frac{\delta\rho}{\rho_0}\vec{g} \tag{3.45}$$

Since the simulations are assumed to encompass incompressible fluids, the divergence of velocity must be equal to zero identically. Following the procedure outlined in section 3.2.1, we can see that two more equations are added to the system:

$$\nabla\cdot\vec{U}_b=0 \tag{3.46}$$

$$\nabla\cdot\delta\vec{U}=0 \tag{3.47}$$

In atmospheric and oceanographic simulations, density is modeled through an equation of state which may be a function of many variables, including location on earth, temperature, salinity, humidity, and others. In functional form, the equation of state might appear as

$$\rho=\rho(T,S,...) \tag{3.48}$$

where T is temperature and S is salinity. A simple linear equation of state for seawater as a function of temperature and salinity may be written in the below form

$$\rho\left(T,S\right)=\rho_0+\rho_0\beta(T-T_{ref})+\rho_0\alpha(S-S_{ref}) \tag{3.49}$$

where T_{ref} and S_{ref} are reference temperatures and salinities, respectively, and where

$$\beta \equiv \frac{1}{\rho_0}\frac{\partial \rho}{\partial T}\bigg|_S \tag{3.50}$$

$$\alpha \equiv \frac{1}{\rho_0}\frac{\partial \rho}{\partial S}\bigg|_T \tag{3.51}$$

This model is simple enough that an explicit formula for $\delta\rho$ can be produced since it is assume the constants $\alpha, \beta, T_{ref}, S_{ref}$, and ρ_0 are constant between the background and perturbation fields

$$\delta\rho(\delta T, \delta S) = \rho(T_b + \delta T, S_b + \delta S) - \rho(T_b, S_b) = \rho_0\beta(\delta T) + \rho_0\alpha(\delta S) \tag{3.52}$$

but for more complex equations of state with more variables and lengthier expressions, potentially including curve-fit data, a more general formulation may be preferred

$$\delta\rho(T_b, \delta T, S_b, \delta S, ...) = \rho(T_b + \delta T, S_b + \delta S, ...) - \rho(T_b, S_b, ...) \tag{3.53}$$

The key equations for the decomposed system derived in this section include eqs. (3.44), (3.45), (3.46), (3.47), (3.48), and (3.53).

3.4 Delta Form of Scalar Transport Equations

Scalars such as temperature, salinity, humidity, and other chemical tracers may also have their transport and diffusion simulated using a generic scalar transport equation. This equation largely takes the form of that included in section 3.2.2. The model scalar transport equation from this section is adapted for use for temperature and salinity here, and the general source/sink terms are removed. Turbulence is closed by an eddy diffusivity formulation.

$$\frac{\partial T}{\partial t} + \vec{U}\cdot\nabla T - \nabla\cdot\left(\left(\kappa_T + \kappa_{t,T}\right)\nabla T\right) = 0 \tag{3.54}$$

$$\frac{\partial S}{\partial t} + \vec{U}\cdot\nabla S - \nabla\cdot\left(\left(\kappa_S + \kappa_{t,S}\right)\nabla S\right) = 0 \tag{3.55}$$

Eddy diffusivities here will be written as in functional form

$$\kappa_{t,T} = \kappa_{t,T}(\nu_t) \tag{3.56}$$

$$\kappa_{t,S} = \kappa_{t,S}(\nu_t) \tag{3.57}$$

Again, velocity $\vec{U}$, temperature T, salinity S, and the eddy diffusivities $\kappa_{t,T}$ and $\kappa_{t,S}$ are decomposed into background and perturbation components.

$$\frac{\partial T_b}{\partial t} + \frac{\partial \delta T}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right) \cdot \nabla \left(T_b + \delta T\right) - \nabla \cdot \left(\left(\kappa_T + \kappa_{tb,T} + \delta\kappa_{t,T}\right) \nabla \left(T_b + \delta T\right)\right) = 0 \quad (3.58)$$

$$\frac{\partial S_b}{\partial t} + \frac{\partial \delta S}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right) \cdot \nabla \left(S_b + \delta S\right) - \nabla \cdot \left(\left(\kappa_S + \kappa_{tb,S} + \delta\kappa_{t,S}\right) \nabla \left(S_b + \delta S\right)\right) = 0 \quad (3.59)$$

Next, a large degree of scale separation is assumed to exist between the background and perturbation fields, and it is assumed that the perturbation is small such that its impact on the evolution of the background in negligible. In this case, the background can be assumed to evolve independent of the perturbation, so we can derive a set of governing differential equations for this background by setting all "delta" fields in the above set of equations to zero.

$$\frac{\partial T_b}{\partial t} + \vec{U}_b \cdot \nabla T_b - \nabla \cdot \left(\left(\kappa_T + \kappa_{tb,T}\right) \nabla T_b\right) = 0 \quad (3.60)$$

$$\frac{\partial S_b}{\partial t} + \vec{U}_b \cdot \nabla S_b - \nabla \cdot \left(\left(\kappa_S + \kappa_{tb,S}\right) \nabla S_b\right) = 0 \quad (3.61)$$

The equations for the perturbation dynamics plus any one-way, non-linear interaction between the background and perturbation fields which acts on the perturbation fields can be determined by subtracting the above background-only equations from the combined background and perturbation equations. The result as well as the background-only equations and constitutive relations for the eddy diffusivities form the complete system of governing equations for scalar transport, and are given below

$$\frac{\partial T_b}{\partial t} + \vec{U}_b \cdot \nabla T_b - \nabla \cdot \left(\left(\kappa_T + \kappa_{tb,T} \right) \nabla T_b \right) = 0 \tag{3.62}$$

$$\frac{\partial S_b}{\partial t} + \vec{U}_b \cdot \nabla S_b - \nabla \cdot \left(\left(\kappa_S + \kappa_{tb,S} \right) \nabla S_b \right) = 0 \tag{3.63}$$

$$\begin{aligned} \frac{\partial \delta T}{\partial t} + \left(\vec{U}_b + \delta \vec{U} \right) \cdot \nabla \left(\delta T \right) - \nabla \cdot \left(\left(\kappa_T + \kappa_{tb,T} + \delta \kappa_{t,T} \right) \nabla \left(\delta T \right) \right) \\ = -\delta \vec{U} \cdot \nabla \left(T_b \right) + \nabla \cdot \left(\delta \kappa_{t,T} \nabla T_b \right) \end{aligned} \tag{3.64}$$

$$\begin{aligned} \frac{\partial \delta S}{\partial t} + \left(\vec{U}_b + \delta \vec{U} \right) \cdot \nabla \left(\delta S \right) - \nabla \cdot \left(\left(\kappa_S + \kappa_{tb,S} + \delta \kappa_{t,S} \right) \nabla \left(\delta S \right) \right) \\ = -\delta \vec{U} \cdot \nabla \left(S_b \right) + \nabla \cdot \left(\delta \kappa_{t,S} \nabla S_b \right) \end{aligned} \tag{3.65}$$

$$\kappa_{tb,T} = \kappa_{t,T}(\nu_{tb}) \tag{3.66}$$

$$\kappa_{tb,S} = \kappa_{t,S}(\nu_{tb}) \tag{3.67}$$

$$\delta \kappa_{t,T} = \kappa_{t,T}(\nu_{tb} + \delta \nu_t) - \kappa_{t,T}(\nu_{tb}) \tag{3.68}$$

$$\delta \kappa_{t,S} = \kappa_{t,S}(\nu_{tb} + \delta \nu_t) - \kappa_{t,S}(\nu_{tb}) \tag{3.69}$$

3.5 Delta Form of $k - \epsilon$ Turbulence Model

With the RANS equations, equation of state, and scalar transport equations rewritten into their delta form equivalents, it remains now to decompose any turbulence model used in the simulations. Many popular turbulence models are used in computational fluid dynamics, including algebraic models, 1-equation, 2-equation, and up to Reynolds stress (6-equation) models. Presented here is the decomposition of a $k - \epsilon$ model with a buoyant production term as derived by Rodi (1987). This is a modification of a 2-equation turbulence model commonly used in engineering and environmental flow simulations (Launder and Spalding 1974). Three quantities are introduced by this model: turbulent kinetic energy (TKE) k, turbulent kinetic energy dissipation rate ϵ, and eddy viscosity ν_t. The two governing partial differential equation for k and ϵ, and the constitutive relation for eddy viscosity are given below

$$\frac{\partial k}{\partial t} + \vec{U} \cdot \nabla k - \nabla \cdot \left(\frac{\nu_t}{\sigma_k} \nabla k \right) = P + G - \epsilon \tag{3.70}$$

$$P = \nu_t \left(\nabla \vec{U} + \left(\nabla \vec{U} \right)^T \right) : \nabla \vec{U} \tag{3.71}$$

$$G = \beta \frac{\nu_t}{\sigma_{t,T}} \nabla T \cdot \vec{g} + \alpha \frac{\nu_t}{\sigma_{t,S}} \nabla S \cdot \vec{g} \tag{3.72}$$

$$\frac{\partial \epsilon}{\partial t} + \vec{U} \cdot \nabla \epsilon - \nabla \cdot \left(\frac{\nu_t}{\sigma_\epsilon} \nabla \epsilon \right) = C_{1\epsilon} \frac{\epsilon}{k} \left(P + C_{3\epsilon} G \right) - C_{2\epsilon} \frac{\epsilon^2}{k} \tag{3.73}$$

$$\nu_t = C_\mu \frac{k^2}{\epsilon} \tag{3.74}$$

where $\sigma_k, \sigma_\epsilon, C_{1\epsilon}, C_{2\epsilon}, C_{3\epsilon}, C_\mu$ are model constants; α and β are volumetric expansion coefficients of salinity and temperature, respectively; $\sigma_{t,T}$ and $\sigma_{t,S}$ is the turbulent Prandtl number and turbulent Schmidt number for salinity, respectively. Physically, P is the shear production term of turbulent kinetic energy and G is the buoyant production of turbulent kinetic energy. The colon operator refers to the double contraction of the two tensor operands into a single scalar. This total production of k is balanced, in part or in full, by the TKE dissipation ϵ as well as the turbulent diffusion and the advection of k. Similarly, TKE production will lead to the increase of ϵ, and production of ϵ is balanced by dissipation of ϵ as well as diffusion and advection. Model constants are tuned from experimental data; Launder and Spalding's original constants are given below with the additional $C_{3\epsilon}$ constant given by Rodi in table 3.1 below (Launder and Spalding 1974; Rodi 1987).

Table 3.1: Buoyant $k - \epsilon$ Model Coefficient (Launder and Spalding 1974; Rodi 1987).

Variable	Value
σ_k	1
σ_ϵ	1.3
$C_{1\epsilon}$	1.44
$C_{2\epsilon}$	1.92
$C_{3\epsilon}$	1 if $\partial\rho/\partial z > 0$; 0-0.2 if $\partial\rho/\partial z \leq 0$
C_μ	0.09

$C_{3\epsilon}$ has potentially 2 different values depending on the stratification of the fluid. If the fluid is stably stratified, buoyancy forces should damp out turbulence thus the contribution of G to growth of ϵ should be small, while for unstable stratification G should have a larger contribution to ϵ as the unstable stratification will lead to an increase in turbulence (Rodi

1987). For the decomposition of the buoyant $k-\epsilon$ model, shorthand is introduced, seen below.

$$kRHS \equiv P + G \tag{3.75}$$

$$\epsilon RHS \equiv C_{1\epsilon}\frac{\epsilon}{k}\left(P + C_{3\epsilon}G\right) - C_{2\epsilon}\frac{\epsilon^2}{k} \tag{3.76}$$

Then one may decompose these terms as normal in a slightly different manner. Here, $kRHS$ and ϵRHS are evaluated in with the un-decomposed variables, e.g. $\vec{U}$ instead of $\vec{U}_b$ or $\delta\vec{U}$, while $kRHS_b$ and ϵRHS_b are evaluated only with the background variables.

$$\delta kRHS = kRHS - kRHS_b \tag{3.77}$$

$$\delta\epsilon RHS = \epsilon RHS - \epsilon RHS_b \tag{3.78}$$

Decomposing the k and ϵ equations gives

$$\frac{\partial k_b}{\partial t} + \frac{\partial \delta k}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right)\cdot\nabla\left(k_b + \delta k\right) - \nabla\cdot\left(\frac{\nu_{tb} + \delta\nu_t}{\sigma_k}\nabla\left(k_b + \delta k\right)\right) = kRHS - \epsilon_b - \delta\epsilon \tag{3.79}$$

$$\frac{\partial \epsilon_b}{\partial t} + \frac{\partial \delta \epsilon}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right)\cdot\nabla\left(\epsilon_b + \delta\epsilon\right) - \nabla\cdot\left(\frac{\nu_{tb} + \delta\nu_t}{\sigma_\epsilon}\nabla\left(\epsilon_b + \delta\epsilon\right)\right) = \epsilon RHS \tag{3.80}$$

$$\nu_{tb} + \delta\nu_t = C_\mu\frac{\left(k_b + \delta k\right)^2}{\epsilon_b + \delta\epsilon} \tag{3.81}$$

Due to separation of scales, the background is assumed to evolve independent of the perturbation, rendering the governing equations of the background turbulence as

$$\frac{\partial k_b}{\partial t} + \vec{U}_b\cdot\nabla k_b - \nabla\cdot\left(\frac{\nu_{tb}}{\sigma_k}\nabla k_b\right) = kRHS_b - \epsilon_b \tag{3.82}$$

$$\frac{\partial \epsilon_b}{\partial t} + \vec{U}_b\cdot\nabla \epsilon_b - \nabla\cdot\left(\frac{\nu_{tb}}{\sigma_\epsilon}\nabla \epsilon_b\right) = \epsilon RHS_b \tag{3.83}$$

$$\nu_{tb} = C_\mu\frac{k_b^2}{\epsilon_b} \tag{3.84}$$

These equations are then subtracted from the above equations to determine the governing equations of the perturbation turbulence plus any non-linear interaction which occurs between the background and perturbation

$$\frac{\partial \delta k}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right) \cdot \nabla \delta k - \nabla \cdot \left(\frac{\nu_{tb} + \delta\nu_t}{\sigma_k} \nabla \delta k\right) = \delta k RHS$$
$$- \delta\epsilon - \delta\vec{U} \cdot \nabla k_b + \nabla \cdot \left(\frac{\delta\nu_t}{\sigma_k} \nabla k_b\right) \quad (3.85)$$

$$\frac{\partial \delta \epsilon}{\partial t} + \left(\vec{U}_b + \delta\vec{U}\right) \cdot \nabla \delta \epsilon - \nabla \cdot \left(\frac{\nu_{tb} + \delta\nu_t}{\sigma_\epsilon} \nabla \delta \epsilon\right) = \delta \epsilon RHS$$
$$- \delta\vec{U} \cdot \nabla \epsilon_b + \nabla \cdot \left(\frac{\delta\nu_t}{\sigma_\epsilon} \nabla \epsilon_b\right) \quad (3.86)$$

$$\delta\nu_t = C_\mu \frac{(k_b + \delta k)^2}{\epsilon_b + \delta\epsilon} - C_\mu \frac{k_b^2}{\epsilon_b} \quad (3.87)$$

It should be noted that a physically-realizable k or ϵ value is non-negative. Since the background k and ϵ values must also be physically-realizable by themselves and therefore non-negative, this implies that the perturbations to k and ϵ must be greater than the negative of their respective background values, e.g. $\delta k >= -k_b$. For the case where $\delta k = -k_b$ and $\delta\epsilon = -\epsilon_b$, $\delta\nu_t$ would equal $-\nu_{tb}$. This has interesting mathematical implications assuming that perturbation to turbulent diffusivity is proportional to perturbation to eddy viscosity, as this implies that the diffusive non-linear interaction term in eq. (3.35) is actually non-diffusive and a non-physical source of a perturbation to a passive scalar. To prevent this, the perturbation to turbulence quantities is taken to be non-negative.

3.6 PISO Algorithm for Multi-scale Localized Perturbation Method

The coupled nature of the RANS equations call for careful numerical solution in order for numerical stability to be achieved. Among the most popular methods is the pressure-implicit splitting of operators (PISO) algorithm developed by Issa (1986). This method takes careful considerations for stability of incompressible flows, and, when adapted to a finite volume method such as in OpenFOAM, can feature grid-staggering to avoid odd-even decoupling. The PISO algorithm is not limited to RANS equations, as Issa shows that it can also be adapted for other coupled equations, namely the $k - \epsilon$ turbulence model. Issa's original implementation focused on low-speed flows while a later modification by Oliveira and Issa allowed special consideration for flows where buoyancy forces are important and the further coupling of an active scalar transport equation within the algorithm (Oliveira and Issa 2001).

The authors include several variants of the PISO with buoyancy algorithm in which the order of steps changes and also provide a brief comparison of the variants, but here the fourth variant will be used for this paper. This variant features the solution of the temperature transport equation in a segregated manner after the solution of the pressure field and

divergence-free velocity field, allowing the transport equations to use a physically-correct velocity field. The coupled nature of the temperature scalar, however, is less-emphasized than in other variants where the temperature field is updated within the solution loop. In addition to solving for the temperature field, the salinity field can also be treated as an active scalar whose transport is solved at the same time as the temperature field, particularly in oceanographic simulations. Further scalars, active and passive, can also be solved in this same manner (Oliveira and Issa 2001).

Algorithm 1 below shows an example PISO-type algorithm for solving the coupled equations of motion, turbulence equations, and scalar transport equations required for the numerical solution of environmental flows.

Algorithm 1 Modified PISO Algorithm for Buoyancy-Driven Flows (based on variant 4 of Oliveira and Issa (2001))

```
while simulation not at final timestep do
  solve momentum predictor using previous timestep pressure data
  for number of pressure-corrector loops < desired number do
    correct velocity
    solve pressure Poisson equation to update pressure
  end for
  solve turbulence model equations
  solve scalar transport equations
end while
```

For the solution of the multi-scale localized perturbation method flows, the above algorithm would need to be adapted to include the solution of the background fields as well as the perturbation fields including any non-linear interaction between the background and perturbation which affects the perturbation fields. In this case, the perturbation fields would be treated as the nominal simulation fields in the PISO algorithm, and any interaction can be treated either implicitly or explicitly depending on the nature of the interaction. Algorithm 2 outlines how one could numerically solve multi-scale localized perturbation method simulations using the aforementioned PISO algorithm.

Algorithm 2 Proposed algorithm for the solution of multi-scale localized perturbation method flows.

while simulation not at final timestep **do**
  if simultaneous solution of background fields **then**
    solve governing equations for all background fields according to PISO algorithm outlined in algorithm 1
  else if using background field data from precursor simulation **then**
    interpolate background data for current timestep from precursor simulation dataset
  else
    assume background fields are "frozen" in time and take background fields from provided background conditions
  end if
  solve governing equations for all perturbation fields according to PISO algorithm outlined in algorithm 1, including effects of any interactions with the background fields
end while

Chapter 4

Implementation

This chapter concerns the implementation of the multi-scale localized perturbation method of the RANS equations including buoyant $k-\epsilon$ turbulence model and active scalar transport in OpenFOAM (*Programmer's Guide*). While three solution methods are outlined in section 3.1.1, only the last method will be implemented in this thesis. In this way, it is assumed that the background fields evolve so slowly that they can be approximated as constant in time, but non-uniform in space, and therefore no extra sets of equations are required to be solved for the background fields.

4.1 Delta Form Equations in OpenFOAM Notation

OpenFOAM is an open-source CFD software suite written in C++ with auxiliary tools written in Python (Jasak, Jemcov, and Tuković 2007). It is capable of using finite-volume methods to solve fluid dynamics problem and is readily modifiable by the user, who uses high-level language to transform the governing mathematical equations into discrete equations in the form of $A\vec{x} = \vec{b}$ which may be solved by linear solvers. Discretization schemes and numerical solvers are implemented beforehand so that researchers can focus on developing and testing models rather than implementing numerical schemes, but researchers may also implement their own schemes and solvers. Discretization and numerical solver settings are selected by OpenFOAM at runtime, making it easy for a user to test different schemes and solvers on a case.

OpenFOAM is built to be general and robust. Each simulation variable, called a field in code, can be either a scalar field, vector field, or tensor field. C++ type substitutions allow a single C++ function to operate on all of these field types without the need to create a single implementation for a scalar field, vector field, and other field types. Similarly, fields can be defined at cell centers or at cell faces. For example, the velocity field $\vec{U}(x, y, z, t)$ defined at all cell centers would be designated a `volVectorField` in code, as velocity is a vector and

Table 4.1: Mathematical operators and their OpenFOAM equivalents (*Programmer's Guide*).

Mathematical function	OpenFOAM equivalent
Dot (scalar) product	`&`
Cross (vector) product	`^`
Dyad	`&&`
$\frac{\partial T}{\partial t}$	`fvm::ddt(T)` or `fvc::ddt(T)`
$\nabla\phi$	`fvc::grad(phi)`
$\nabla \cdot \left(\alpha\vec{U}\right)$	`fvm::div(alpha,U)` or `fvc::div(alpha,U)`
$\nabla \times \vec{U}$	`fvc::curl(U)`
$\nabla \cdot (\kappa\nabla S)$ or $\kappa\nabla^2 S$	`fvm::laplacian(kappa,S)` or `fvc::laplacian(kappa,S)`

the field refers to volume-averaged values. Another case is the volumetric flux across each cell face, which would be considered a `surfaceScalarField` in code.

4.1.1 Operators

OpenFOAM uses two namespaces for its calculus operators: `fvm`, or finite volume method, for implicit operators which generate left-hand side values for the coefficient matrix A, while `fvc`, or finite volume calculus, builds the right-hand side $\vec{b}$. Table 4.1 lists a selection of mathematical operators implemented within OpenFOAM.

OpenFOAM linearizes the advection term using previous timestep velocity flux ϕ such that

$$\phi = \vec{U}_f \cdot d\vec{S}$$

for the incompressible cases, where $\vec{U}_f$ is the velocity vector at each face and $d\vec{S}$ is the outward-pointing area vector of each face (*Programmer's Guide*). This approximation stems from the finite volume method where the volume integral of the divergence of a vector field can be rewritten as a surface integral using Gauss' theorem.

$$\iiint_V \nabla \cdot (\vec{U} \otimes \vec{U}) dV = \oiint_S \vec{U}\left(\vec{U} \cdot d\vec{S}\right) \approx \sum_{i=1}^{N_{faces}} \phi_i \vec{U}_i \equiv \texttt{fvm::div(phi,U)}$$

In the above expression, the `fvm` namespace denotes that the divergence operator will discretize divergence of the velocity field `U` using the previous timestep volumetric flux `phi`. The

result is a coefficient matrix for `U` for the advection term using the user-specified divergence discretization scheme selected for the given case. The PISO algorithm ensures that this flux and the velocity field itself are divergence free.

4.1.2 Delta Turbulence Model Implementation

The addition of turbulence models, whether large-eddy simulation or Reynolds-averaged Navier-Stokes models, is straightforward within OpenFOAM. The implementation of the buoyant $k-\epsilon$ model requires a simple modification of the default $k-\epsilon$ model to allow for the loading of the scalar stratification field, additional constants, then the calculation of the buoyancy production term for each equation. The delta turbulence model, however, is a large deviation from the nominal turbulence model and it was decided to develop a new class of turbulence model called `deltaTurbulenceModels`. Most of the code for this class as well as all extending classes is reused from the nominal turbulence model class in OpenFOAM as well as the $k-\epsilon$ turbulence model, with most differences stemming from the addition of new fields representing background and delta fields, including those for turbulent quantities such as turbulent kinetic energy k, dissipation of turbulent kinetic energy ϵ, and eddy viscosity ν_t.

The code for the nominal $k-\epsilon$ turbulence model found within OpenFOAM shows how the k and ϵ equations are interpreted into OpenFOAM code. The shear production term P is replaced by a shear production term based on the anisotropic part of the Reynolds stress tensor P_k (*k-epsilon*). In incompressible flows, $P_k = P$ but for compressible flows an additional term which factors in the divergence of velocity is introduced, allowing for the same turbulence model to be implemented for both incompressible and compressible flows. Similarly, phase fractions are treated within the OpenFOAM implementation of the $k-\epsilon$ model to allow for multi-phase flow simulation. Since the cases discussed in this thesis are single-phase and incompressible, all explicit velocity divergence terms and phase fractions variables are removed to simplify implementation. In code, P_k is written as

```
volTensorField tGradU = fvc::grad(this->U_);
volScalarField Pk = this->nut_ * dev(twoSymm(tGradU)) && tGradU;
```

The `dev()` function returns the deviatoric part of the tensor supplied as the argument, while the `twoSymm()` function generate a symmetric tensor from the argument tensor by adding the transpose of the argument tensor to itself.

Furthermore, explicit sink terms on the right-hand side are moved to the left-hand side to improve diagonal dominance of the coefficient matrix. This is done using the `fvm::Sp()` function after rewriting the sink term so that it is a function of the equation variable, e.g. $-\epsilon = -\frac{\epsilon}{k}k$. Following the aforementioned simplifications, eq. (3.73) is rendered in OpenFOAM code as shown below.

```
fvScalarMatrix epsilonEqn
(
fvm::ddt(epsilon_)
+ fvm::div(phi,epsilon_)
- fvm::laplacian(DepsilonEff(),epsilon_)
==
C1_ * Pk * epsilon_ / k_
- fvm::Sp(C2_ * epsilon_ / k_, epsilon_)
+ fvOptions(epsilon_)
);
```

The k equation in eq. (3.70) is rendered a similar way (*k-epsilon*).

```
fvScalarMatrix kEqn
(
fvm::ddt(k_)
+ fvm::div(phi,k_)
- fvm::laplacian(DkEff(),k_)
==
Pk
- fvm::Sp(epsilon_ / k_, k_)
+ fvOptions(k_)
);
```

The functions `DepsilonEff()` and `DkEff()` return effective diffusivities of k and ϵ, given as

$$\texttt{DkEff} = \nu + \frac{\nu_t}{\sigma_k}$$
$$\texttt{DepsilonEff} = \nu + \frac{\nu_t}{\sigma_\epsilon}$$

Next step is to modify the above equations so that all fields are expressed in terms of their background quantity and their delta quantity. Since we have already done so in eqs. (3.85) through (3.87, we may formulate these expressions into their OpenFOAM equivalents. Firstly, a new expression for buoyancy production is implemented. Instead of being a function of two stratification scalars, one scalar, called `Theta` and its equivalent background quantity `ThetaBackground`, is used, along side volumetric expansion coefficient `beta_` and turbulent diffusivity coefficient `sigmat_`. The fields for `Theta` and `ThetaBackground` are specified at runtime in case-specific files, as are the values of `beta_` and `sigma_`. The expressions for the buoyancy production expressions are given as.

```
volScalarField Gk = beta_ * this->nut_ * (g_ & fvc::grad(Theta)) / sigmat_;
volScalarField GkBackground = beta_ * this->nutBackground_ *
(g_ & fvc::grad(ThetaBackground)) / sigmat_;
```

Shear production is decomposed in the same manner, extending the above expression for shear production:

```
volTensorField tGradUBackground = fvc::grad(this->UBackground_);
volScalarField PkBackground = this->nutBackground_ * dev(twoSymm(tGradUBackground))
&& tGradUBackground;
```

Note that one must modify the constructor of the turbulence model to allow for `UBackground` to be specified.

$\delta\epsilon$ Equation

The treatment of the sink term on the right-hand side of the ϵ equation is different on account of the `fvm::Sp()` function. This function treats a term as an implicit source, and is used to "move" a negative term from the right-hand side of $A\vec{x} = \vec{b}$ to the diagonal of the coefficient matrix A, which improves convergence of linear solvers (*Programmer's Guide*). For the term `fvm::Sp(C2_ * epsilon_ / k_, epsilon)`, the first argument is treated explicitly and is used to generate the entries to the coefficient matrix while the second is the variable for which the coefficient matrix will be generated. Decomposing the variables into background and perturbation components is done so that all terms are multiplied by $\delta\epsilon$, as shown below.

$$C_2\frac{\epsilon}{k}\epsilon = C_2\frac{\epsilon}{k}\left(\epsilon_b + \delta\epsilon\right) = C_2\frac{\epsilon_b + \delta\epsilon}{k}\epsilon_b + C_2\frac{\epsilon}{k}\delta\epsilon = C_2\frac{\epsilon_b}{k}\delta\epsilon + C_2\frac{\epsilon_b^2}{k\delta\epsilon}\delta\epsilon + C_2\frac{\epsilon}{k}\delta\epsilon$$

This enables the `fvm::Sp()` object to be used on all new terms, as shown below.

$C_2\frac{\epsilon}{k}\epsilon \equiv$ `fvm::Sp(C2_ * epsilon_ / k_, delepsilon_)`

```
    + fvm::Sp(C2_ * epsilonBackground_ / k_, delepsilon_)
    + fvm::Sp(C2_ * sqr(epsilonBackground_) / (k_ * delepsilon_), delepsilon_)
```

This does not need to be repeated for the background-only portion of destruction term, as it is a negative term and we are subtracting it from the right-hand side, making it positive. With all of this in mind, the ϵ equation right-hand side is written as

```
fvScalarMatrix epsilonRHS
```

```
(
C1_ * Pk * epsilon_ / k_
+ C1_ * C4_ * Gk * epsilon_ / k_
- fvm::Sp(C2_ * epsilon_ / k_, delepsilon_)
- fvm::Sp(C2_ * epsilonBackground_ / k_, delepsilon_)
- fvm::Sp(C2_ * sqr(epsilonBackground_) / (k_ * delepsilon_), delepsilon_)
);
```

The right-hand side considering background fields only is given as

```
fvScalarMatrix epsilonBackgroundRHS
(
C1_ * PkBackground * epsilonBackground_ / kBackground_
+ C1_ * C4_ * GkBackground * epsilonBackground_ / kBackground_
- C2_ * sqr(epsilonBackground_) / kBackground_
);
```

The final $\delta\epsilon$ equation matrix, after accounting for the non-linear advection and diffusion interaction terms created by the decomposition, becomes

```
fvScalarMatrix delepsEqn
(
fvm::ddt(delepsilon_)
+ fvm::div(phi,delepsilon_)
- fvm::laplacian(DepsilonEff,delepsilon_)
==
epsilonRHS
- epsilonBackgroundRHS
- fvc::div(delphi,epsilonBackground_)
+ fvc::laplacian(delDepsilonEff,epsilonBackground_)
+ fvOptions(delepsilon_)
);
```

This code block forms the final set of equations for the $\delta\epsilon$ field. These equations are then relaxed then solved. OpenFOAM enables a runtime-specified minimum bound to be observed on all values of the final solution to `delepsEqn`, and with slight modification this can be applied to the $\delta\epsilon$ field.

δk Equation

Once the $\delta\epsilon$ fields are solved, we may solve for the δk fields. The decomposition of the right-hand side follows the same approach to the above method. The sink term in the k equation is handled using a `fvm::Sp()` object, so the term is carefully handled to ensure that the decomposed sink terms retain this object which entails that all terms are functions of δk. This is illustrated in the below equation.

$$\epsilon = \frac{\epsilon}{k}k = \frac{\epsilon}{k}(k_b + \delta k) = \frac{k_b\epsilon}{k\delta k}\delta k + \frac{\epsilon}{k}\delta k$$

When substituted onto the right-hand side along with the other production terms, the $kRHS$ term from eq. (3.85) is rendered into OpenFOAM equation code as

```
fvScalarMatrix kRHS
(
Pk
+ Gk
- fvm::Sp(epsilon_ / delk_, delk_)
);
```

The right-hand side considering background fields only is given as

```
fvScalarMatrix kBackgroundRHS
(
PkBackground
+ GkBackground
- epsilonBackground_
);
```

The final governing equation for δk given by eq. (3.85) is then rendered in OpenFOAM as

```
fvScalarMatrix delkEqn
(
fvm::ddt(delk_)
+ fvm::div(phi,delk_)
- fvm::laplacian(DkEff,delk_)
==
kRHS
- kBackgroundRHS
```

```
- fvc::div(delphi,kBackground_)
+ fvc::laplacian(delDkEff,kBackground_)
+ fvOptions(delk_)
);
```

Two extra terms on the right-hand side account for the advection of the k_b field by the perturbation to velocity and the diffusion of k_b due to a perturbation to turbulence, respectively. Once this $A\vec{x} = \vec{b}$ system is assembled, it is relaxed then solved for the values of δk at every point. As with $\delta\epsilon$, a lower bound is applied to values of δk. The final values of δk can be added to k_b to recover a final k value for each cell for the current solver iteration.

$\delta\nu_t$ Relation

The final step to updating turbulence quantities is the calculation of ν_t and $\delta\nu_t$. This relation is algebraic in nature, and can be implemented with a few lines of C++ code. This is done following eq. (3.87).

```
this->delnut_ = Cmu_ * (sqr(k_) / epsilon_ - sqr(kBackground_) / epsilonBackground_);
```

The turbulence model routine is concluded by updating the total eddy viscosity ν_t.

```
this->nut_ = Cmu_ * sqr(k_) / epsilon_;
```

4.1.3 Solver Implementation Outline

Implementing a solver in OpenFOAM begins by navigating to the OpenFOAM solver directory in `$FOAM_SOLVERS` and locating a suitable starting solver to modify; much of the code within a solver can be reused to save time. The standard solver `buoyantBoussinesqPimpleFoam` is a close equivalent to the delta solver described in this thesis as it already implements the PISO algorithm with buoyancy force modeled under the Boussinesq approximation, so this will serve as the starting point for the new solver. Users should not directly modify the OpenFOAM source code, so this solver directory is copied to a user directory where appropriate modifications will be made and the solver compiled. While OpenFOAM uses nominal C++ code style for most of its code, OpenFOAM solvers use a non-standard code style that is closer to a high-level code outline. This makes it easier for a researcher who is familiar with mathematical models to pick up OpenFOAM and implement a new model quickly and easily. After copying the code and renaming all references of `buoyantBoussinesqPimpleFoam` to `deltaBuoyantBoussinesqPimpleFoam`, one may edit the main file which in the OpenFOAM convention is the solver name with the `.C` extension added to the end of it. This file provides an outline of the solver algorithm, as shown below. Header files which end in `.H` are included using the `#include` macro, and for the solvers, subsections of the algorithm are refactored into these files for ease of viewing and editing. This behavior is non-standard as far as C++

code style goes, but it allows a user to quickly view what the script is doing at a glance, and it transforms the code to look more like a high-level algorithm.

`deltaBouyantBoussinesqPimpleFoam.C`

A summarize view of the `deltaBuoyantBoussinesqPimpleFoam.C` file is given below. Please note that for the sake of space, several lines of code have been omitted, such as `#include` statements. This is largely adapted from the `buoyantBoussinesqPimpleFoam` solver, with one notable exception being the solution of the temperature scalar transport equation after turbulence is updated for the current timestep.

```
int main(int argc, char *argv[])
{

#include "createFields.H"

while (runTime.run())
{

while (pimple.loop())
{

#include "UEqn.H"

while (pimple.correct())
{
#include "pEqn.H"
}

if (pimple.turbCorr())
{
laminarTransport.correct();
turbulence->correct();
}

#include "TEqn.H"
}

}

}
```

One benefit of OpenFOAM's high-level solver implementation is that solver algorithms are simple to follow. Within the `main()` function, the solver first loads and initializes the solver fields using code found in the `createFields.H` file, then it performs its main loop for the user-defined time length. For each timestep, the solver performs a number of PIMPLE loops, the number of which the user specifies in each case setting files. PIMPLE is a portmanteau of SIMPLE and PISO, which uses the unsteady PISO algorithm combined with SIMPLE loop for each timestep. The code within the `while (pimple.loop())` block is effectively a SIMPLE loop. Firstly, the solver solves the momentum equation defined within `UEqn.H`, then performs another loop to solve the pressure equation for the incompressible fluid. This equation is found within `pEqn.H`. Once this is satisfied, turbulence and any transport phenomena are simulated and updated, then any passive scalars, such as temperature, are solved for. For this case, the temperature equations are found within `TEqn.H`.

`createFields.H, UEqn.H, pEqn.H,` and `TEqn.H` are found within the same directory as `deltaBuoyantBoussinesqPimpleFoam.C`, as these are files that are integral to this specific solver, but other `.H` files included within `deltaBuoyantBoussinesqPimpleFoam.C` are typically general functions that are shared between OpenFOAM solvers. The source code for these files can be found within OpenFOAM source code.

`createFields.H` File

This file contains the code to initialize simulation fields, either from input files or from a prescribed value. This is largely untouched except to add in the background and perturbation variables, which follow a similar format as the existing variables. The code for adding `UBackground` and `delU` is shown below.

```
volVectorField UBackground
(
IOobject
(
"UBackground",
runTime.timeName(),
mesh,
IOobject::MUST_READ,
IOobject::NO_WRITE
),
mesh
);

volVectorField delU
(
```

```
IOobject
(
"delU",
runTime.timeName(),
mesh,
IOobject::MUST_READ,
IOobject::AUTO_WRITE
),
mesh
);
```

Here, the keyword `IOobject::MUST_READ` in the `UBackground` block means that the solver will attempt to load a file in a time directory named `UBackground` and OpenFOAM will throw an error if it cannot load such a file. The keyword `IOobject::AUTO_WRITE` means that the `delU` field will automatically print its values for each cell every time the user specifies the solver to output the data; `IOobject::NO_WRITE` means that the `UBackground` fields will not be written, which is acceptable since we are assuming for this variant of the delta solver that all background fields are constant in time. This process is repeated for all other decomposed variables. The background and perturbation volumetric flux through each cell face is also added as shown below.

```
surfaceScalarField phiBackground("phiBackground",fvc::flux(UBackground));
surfaceScalarField delphi("delphi",fvc::flux(delU));
```

After initializing the temperature and density fields, we can calculate the initial density field and the time-constant background density fields using the equation of state.

```
volScalarField T = TBackground + delT;
rhoBackground = rho0 + rho0 * beta * (TBackground - TRef);
rho = rho0 + rho0 * beta * (T - TRef);
```

The total form of the variables, e.g. `U`, do not need to be loaded or written to disk as they can be determined by the addition of their respective background and perturbation components as seen below.

```
volVectorField U(UBackground + delU);
```

Since a new class of turbulence models is created for the multi-scale localized perturbation method, the code to initialize the turbulence model in `createFields.H` is adapted from the nominal turbulence model read code.

```
autoPtr<incompressible::deltaTurbulenceModel> turbulence
(
incompressible::deltaTurbulenceModel::New(UBackground, delU, phiBackground, delphi,
laminarTransport)
```

```
);
```

Other modifications to this file include setting the delta pressure reference point. In the stock OpenFOAM `buoyantBoussinesqPimpleFoam` solver, `p_rgh` is the dynamic pressure defined as the total pressure minus the hydrostatic pressure, so we can replace these terms with `delp`. This step does not need to be repeated for `pBackground` as this field is not solved for during the main loop. Indeed, it is not even fully necessary to initialize `pBackground` at all for this variant of the multi-scale localized perturbation method as the background pressure does not appear in any equations for the delta variables.

`UEqn.H` File

The file `UEqn.H` contains the routines for building and solving the discrete equations of conservation of momentum, completing the momentum predictor step of the PISO algorithm. The default `buoyantBoussinesqPimpleFoam` form of `UEqn.H` contains functions which accounts for mesh motion, but these will not be included in `deltaBuoyantBoussinesqPimpleFoam`. The momentum equation is formulated in `buoyantBoussinesqPimpleFoam` as

```
fvVectorMatrix UEqn
(
fvm::ddt(U)
+ fvm::div(phi,U)
+ fvm::DDt(U) /* accounts for mesh motion */
+ turbulence->divDevRhoReff(U)
==
fvOptions(U)
);
```

Turbulence is included in the momentum equation with the specific contribution being calculated in a function implemented in the turbulence model called `divDevRhoReff(U)`, which represents the divergence of the rate of shear tensor. For compressible flows, this function takes the density field and velocity field as arguments while for incompressible cases it only takes the velocity field as an argument. For linear stress models, this function returns the following function

```
 - fvm::laplacian(nuEff(),U) - fvc::div(nuEff(),dev2(T(fvc::grad(U))))
```

which represents in the code the function

$$-\nabla \cdot \left(\nu_{eff} \nabla \vec{U}\right) - \nabla \cdot \left(\nu_{eff} \left[\left(\nabla \vec{U}\right)^T - \frac{2}{3} \nabla \cdot \vec{U} \underline{I}\right]\right)$$

Here, `nuEff()` refers to the effective viscosity of the fluid which depends on the flow as well

as the turbulence model used. For eddy viscosity models, `nuEff()` will return the sum of the molecular and total eddy viscosity. Note that this whole expression is multiplied by -1 since this term is placed on the left-hand side of the momentum equation.

The delta form of buoyant $k-\epsilon$ model requires implementation of the delta form of the divergence of the rate of shear tensor. This is shown below

$$-\nabla\cdot\left(\nu_{eff}\nabla\vec{U}\right)-\nabla\cdot\left(\nu_{eff}\left[\left(\nabla\vec{U}\right)^{T}-\frac{2}{3}\nabla\cdot\vec{U}\underline{I}\right]\right)+\nabla\cdot\left(\nu_{eff,b}\nabla\vec{U}_{b}\right)+\nabla\cdot\left(\nu_{eff,b}\left[\left(\nabla\vec{U}_{b}\right)^{T}-\frac{2}{3}\nabla\cdot\vec{U}_{b}\underline{I}\right]\right) \tag{4.1}$$

This above equation can be written in OpenFOAM language as

```
 fvVectorMatrix delDivDevRhoReff = - fvm::laplacian(turbulence->nuEff(),delU)
- fvc::laplacian(turbulence->nuEff(),UBackground)
- fvc::div(turbulence->nuEff(),dev2(T(fvc::grad(U))))
+ fvc::laplacian(turbulence->nuEffBackground(),UBackground)
+ fvc::div(turbulence->nuEffBackground(),dev2(T(fvc::grad(UBackground))))
```

Because `fvm::laplacian()` generates the coefficient matrix A, the second operator must be `delU` in order to solve for $\delta\vec{U}$. Therefore, one must split the first term of eq. (4.1) must be split into two terms, one implicit and one explicit. This is done by splitting the divergence operator in two using vector calculus identities as shown above. `nuEffBackground()` represents the sum of molecular kinematic viscosity and background eddy viscosity. With this term constructed, eq. (3.45) without the pressure gradient and buoyancy force is rendered in OpenFOAM as

```
fvVectorMatrix delUEqn
(
fvm::ddt(delU)
+ fvm::div(phi,delU)
+ fvc::div(phi,UBackground)
- fvc::div(phiBackground,UBackground)
+ delDivDevRhoReff
==
+ fvOptions(delU)
);
```

This object is the code equivalent of $A\vec{x}=\vec{b}$ for $\delta\vec{U}$, and is then relaxed and any optional constraints are applied.

```
delUEqn.relax();
```

```
fvOptions.constrain(delU);
```

Next, the momentum prediction step is performed and any runtime-specified corrections are applied to the results.

```
solve(delUEqn == -fvc::grad(delp) + (rho - rhoBackground) * g / rho0);
fvOptions.correct(delU);
```

Here, `-fvc::grad(delp) + (rho - rhoBackground) * g / rho0` adds the explicit momentum source terms to the momentum equation. These fields are generated with previous timestep pressure and density field data. With the momentum prediction complete, the pressure field needs to be solved for such that the velocity field ends the timestep divergence free.

`pEqn.H` File

The results of the momentum prediction step is a velocity field for the current timestep, but this field is determined with previous timestep pressure information and is not divergence free. Issa called this step the pressure corrector, and in OpenFOAM solvers this file `pEqn.H` is called once per pressure corrector loop.

Given the system of equations $A\vec{x} = \vec{b}$, let $\tilde{A}$ represent the diagonal elements of A and let $\overline{H}$ represent all other elements of A. This renders $A\vec{x} = \vec{b}$ as $\tilde{A}\vec{x} = \vec{b} - \overline{H}\vec{x}$. For the final system of equations represented by the code object `delUEqn`, the values of $\tilde{A}$ and $\overline{H}$ depends on data from the current timestep which will be solved for in the current PISO iteration as well as data from previous timestep(s), and the components of $\vec{b}$ depends on data from the previous timestep(s). If the solution to the momentum predictor step is given as $\delta\vec{U}^*$, then the equation represented by `solve(delUEqn == - fvc::grad(delp) + rhok * g)` can be written in matrix-vector for as

$$\tilde{A}\delta\vec{U}^* = \vec{b} - \overline{H}\delta\vec{U}^* - \nabla\delta p + \rho_k\vec{g} \tag{4.2}$$

We can replace $\vec{b} - \overline{H}\delta\vec{U}^*$ with the term $\tilde{H}$, yielding

$$\tilde{A}\delta\vec{U}^* = \tilde{H} - \nabla\delta p + \rho_k\vec{g} \tag{4.3}$$

With predicted velocity $\delta\vec{U}^*$ already determined, one now needs to find the corrected velocity $\delta\vec{U}^{**}$. To do this, the current timestep pressure data is determined. From eq. (4.3), the $\delta\vec{U}^*$ vector is replaced by $\delta\vec{U}^{**}$ as shown in eq. (4.4) without rebuilding the coefficient matrix A and therefore the matrices $\tilde{A}$ and $\tilde{H}$. This step saves time since $\tilde{A}$ and $\tilde{H}$ can be stored in memory.

$$\tilde{A}\delta\vec{U}^{**} = \tilde{H} - \nabla\delta p + \rho_k\vec{g} \tag{4.4}$$

Left-multiplying both sides by $\tilde{A}^{-1}$ yields the solution to the discrete equations of $\delta\vec{U}^{**}$ as eq. (4.5).

$$\delta\vec{U}^{**} = \tilde{A}^{-1}\tilde{H} - \tilde{A}^{-1}\nabla\delta p + \tilde{A}^{-1}\rho_k\vec{g} \tag{4.5}$$

While $\delta\vec{U}^{*}$ is not guaranteed to divergence free, $\delta\vec{U}^{**}$ will be. Thus, we can take the divergence of both sides of eq. (4.5) to yield eq.(4.6).

$$0 = \nabla\cdot\delta\vec{U}^{**} = \nabla\cdot\left(\tilde{A}^{-1}\tilde{H} + \tilde{A}^{-1}\rho_k\vec{g}\right) - \nabla\cdot\left(\tilde{A}^{-1}\nabla\delta p\right) \tag{4.6}$$

Rearranging this equation yields a Poisson equation for pressure with a source term equivalent to the divergence of $\tilde{A}^{-1}\tilde{H} + \rho_k\vec{g}$. This pressure Poisson equation is solved at the cell faces as a form of grid staggering that helps prevent odd-even decoupling for central difference schemes. This procedure is mirrored closely by OpenFOAM code. Systems of equations in OpenFOAM have operators which return the $\tilde{A}$ and $\tilde{H}$ coefficient matrices. The inversion of $\tilde{A}$ followed by its interpolation to the cell faces, for example, is given by

```
volScalarField rAU("rAU", 1 / delUEqn.A());
surfaceScalarField rAUf("rAUf", fvc::interpolate(rAU));
```

Next, the object $\tilde{A}^{-1}\tilde{H}$ is created by the code

```
volVectorField HbyA(constrainHbyA(rAU * delUEqn.H(), delU, delp));
```

The function `constrainHbyA(HbyA, delU, delp)` returns a modified version of $\tilde{A}^{-1}\tilde{H}$ where prescribed velocity boundaries, such as inlet or slip/no-slip boundaries, are enforced on the $\tilde{A}^{-1}\tilde{H}$ object. Next, the divergence of the gravity force vector `phig` is calculated by Gauss' theorem.

```
surfaceScalarField phig(fvc::interpolate(rAU * rhok * g) & mesh.Sf());
```

where `mesh.Sf()` returns the outward-pointing area vector of each cell face. The flux of `HbyA` is also calculated, allowing for an additional factor `fvc::ddtCorr(delU, delphi)`.

```
surfaceScalarField phiHbyA
(
"phiHbyA",
fvc::flux(HbyA)
+ rAUf * fvc::ddtCorr(delU, delphi) );
```

If there are no Dirichlet pressure boundaries, mass flux across all boundaries is adjusted using the function

```
adjustPhi(phiHbyA, delU, delp);
```

The body force flux is appended to the flux of $\tilde{A}^{-1}\tilde{H}$ using

```
phiHbyA += phig;
```

then pressure gradient boundary conditions are updated using velocity boundary conditions.

```
constrainPressure(delp, delU, phiHbyA, rAUf);
```

This line is important if the pressure boundary condition `fixedFluxPressure` is used on a boundary where the velocity boundary condition is prescribed. At a wall, for example, wall-normal velocity is known a priori, and therefore prescribed, as $\vec{U} \cdot \hat{n} = 0$, so the wall-normal pressure gradient is adjusted so that this velocity condition is observed. This can be retrieved dotting both sides of eq. (4.5) with the outward-pointing area vector, then setting $\delta\vec{U}^{**}$ equivalent to the wall-normal velocity boundary condition. The resulting wall-normal pressure gradient is then applied to the `fixedFluxPressure` pressure boundaries.

Current timestep pressure data can now be determined, which is demonstrated below. OpenFOAM allows for non-orthogonal corrector steps to be performed within the pressure corrector loop which improves performance on some meshes, particularly those with high non-orthogonality.

```
while (pimple.correctNonOrthogonal())
{
fvScalarMatrix delpEqn
(
fvm::laplacian(rAUf, delp) == fvc::div(phiHbyA) );

delpEqn.setReference(pRefCell, getRefCellValue(delp, pRefCell));
delpEqn.solve(mesh.solver(delp.select(pimple.finalInnerIter())));

if (pimple.finalNonOrthogonalIter())
{
delphi = phiHbyA - delpEqn.flux();
delp.relax();
delU = HbyA + rAU * fvc::reconstruct((phig - delpEqn.flux()) / rAUf);
delU.correctBoundaryConditions();
fvOptions.correct(delU); }
}

U = UBackground + delU;
p = pBackground + delp;
phi = phiBackground + delphi;
```

```
#include "continuityErrs.H"
```

If no Dirichlet pressure boundary conditions are given, then a reference value of δp must be prescribed at a given location. On the final non-orthogonal iteration, the corrected velocity $\delta\vec{U}^{**}$ is determined by eq. (4.5). The function `fvc::reconstruct()` performs the opposite of `fvc::interpolate()` in that it interpolates cell face values of a field to the cell center values. This is done for the body force flux and the flux associated with the pressure field. The volumetric flux $\delta\phi$ is also updated. Boundary conditions for $\delta\vec{U}$ are also corrected and the pressure field is relaxed if necessary. Total fields for $\vec{U}, p$, and ϕ are updated and continuity errors can be calculated for the current timestep for diagnostics. With this step completed, the velocity field is nearly divergence free and the current timestep pressure field is determined.

Turbulence Model Update

With pressure and velocity information for the current timestep, the equations of the turbulence model are solved to update turbulence fields. This is done with the line

```
turbulence->correct();
```

in `deltaBuoyantBoussinesqPimpleFoam.C`. The function `correct()` is implemented in the `deltaTurbulenceModel` class.

`TEqn.H` File

Current timestep flow data is now used to solve the governing equations of scalar transport. This correlates to eq. (3.35) for any scalars in the flow, including stratifying scalars. A model implementation is illustrated in this section for a temperature T divided into a background component T_b and a delta component δT. An eddy diffusivity model is used to provide the turbulent diffusion of temperature using a Prandtl number `Pr` and a turbulent Prandtl number `Prt`, shown below.

```
volScalarField kappaEff("kappaEff", turbulence->nu() / Pr + turbulence->nut()
/ Prt);
```

Next, the $A\vec{x} = \vec{b}$ object for the scalar transport equations is built as shown below.

```
fvScalarMatrix delTEqn
(
fvm::ddt(delT)
+ fvm::div(phi, delT)
```

```
- fvm::laplacian(kappaEff, delT)
==
fvOptions(delT)
);
```

This is then relaxed then solved, with non-linear interaction between the background and perturbation fields included.

```
delTEqn.relax();
fvOptions.constrain(delT);
solve(delTEqn
==
- fvc::div(delphi, TBackground)
+ fvc::laplacian(turbulence->delnut() / Prt, TBackground)
);
fvOptions.correct(delT);

T = TBackground + delT;
rho = rho0 + rho0 * beta * (T - TRef); /* update density */
```

This process can be extended to any number of scalars. Additional source terms may also be including assuming that any non-linear interaction terms are also handled.

Chapter 5

Case Studies and Discussions

With the implementation of the multi-scale localized perturbation method in OpenFOAM, the task now remains to compare the performance of the delta solver to the full solver. Three test cases are selected for the purpose of verification and demonstration. The first one is a stratified wake collapse simulation where the delta form of the incompressible URANS equations is tested against a nominal URANS formulation, but without any background fields except for a linear background temperature stratification. Results are also compared against experimental data. The second case is an internal gravity wave interaction case, where internal gravity waves in the background field interact with internal gravity waves in the delta fields. This case also serves as a demonstration case for the worst case scenario with respect to scale separation; the background and perturbation scales are identical, which means that this case should see the most extreme differences between the delta form and an identical, non-delta form simulation where both sets of internal gravity waves are simulated at the same time. A third case is a demonstrative one where a background shear current is placed within the domain, then internal gravity waves from a collapsing stratified wake interact with this shear current. The internal gravity waves are shown to interact with the current in non-trivial ways. This case serves to demonstrate how the multi-scale localized perturbation method can be used to simplify case setup and visualization.

5.1 Stratified Wake Collapse Case

The first case run with the delta solver is a simulation of a temperature-stratified net-zero momentum wake based on the those done by Hassid (1980a), who in turn compares to experimental data from Lin and Pao. This simulation serves as verification that the total solver and the delta solver produce nearly identical results in the absence of a background velocity field as well as show that both solvers are able to replicate experimental results. The paper by Hassid shows data for an unstratified wake as well as a wake with a Froude

number of 31, but only the stratified case is replicated.

5.1.1 Case Setup

The case is run in a 2D+t method where it is assumed that streamwise variations in fields are small. A Galilean transform can be used to convert the time dimension into a streamwise dimension based on the moving disturbance velocity U_0.

$$x = U_0 t$$

Domain and Boundary Conditions

A 2D rectangular domain is used for this case, shown in fig. 5.1. The domain is periodic in the streamwise direction, outlet boundary conditions are set in the y direction, and free-slip walls are used in the z direction. A beach is applied in the domain, intensifying towards the y and z direction boundaries.

The domain has a total of 320,677 cells: there are 751 cells in the vertical direction throughout the domain with the center portion of the domain having 375 cells horizontally. The cells in the beaches on the $\pm y$ side of the domain stretch as they approach the boundaries, and from $y = 125m$ to $y = 250m$, there are 26 cells within the beach in the y direction; the same is done for the beach that extends from $y = -125m$ to $y = -250m$. This gradient is shown is fig. 5.2 below. No cell stretching is performed in the vertical direction in any of the beaches.

Initial Conditions

A net-zero momentum wake is assumed to be generated by a moving disturbance within the water. This disturbance is traveling in the $+x$ direction at a velocity of U_0, leaving a NZM wake with an initial centerline wake velocity of U_{D0} directed in the $-x$ direction. The disturbance velocity is non-dimensionalized the Brunt-Väisälä frequency N and the initial wake diameter D to form a Froude number Fr shown below.

$$Fr \equiv \frac{U_0}{ND} \tag{5.1}$$

where

$$N^2 = \frac{g}{\rho_0}\frac{\partial \rho}{\partial z}$$

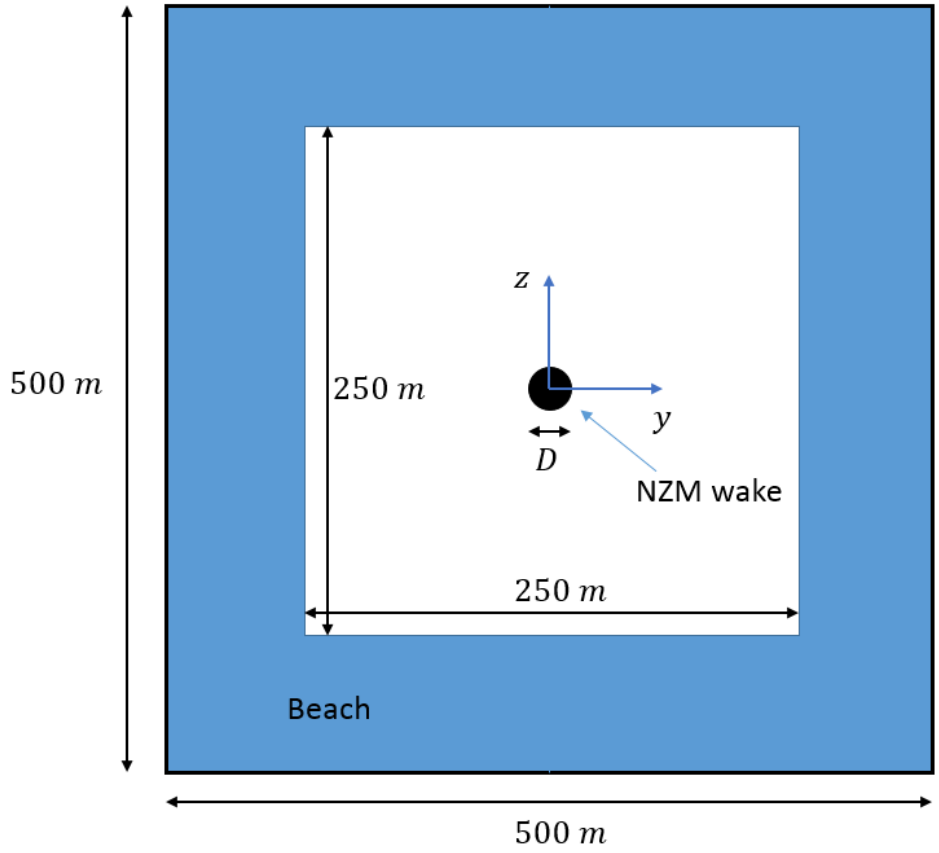

Figure 5.1: Domain used for the stratified wake collapse case

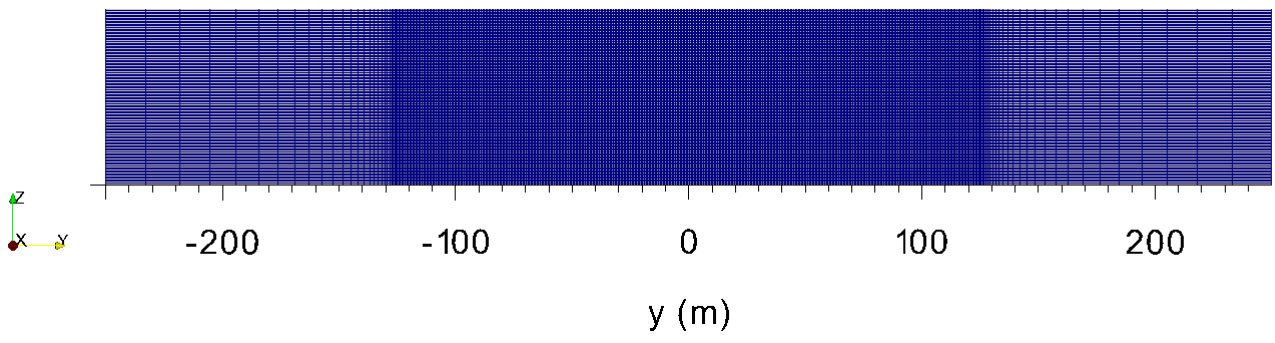

Figure 5.2: Mesh grading in the y direction.

Density is modeled using a linear equation of state

$$\rho(T) = \rho_0 - \rho_0 \beta \left(T - T_{ref}\right) \tag{5.2}$$

where the thermal expansion coefficient $\beta = 2.1 \times 10^{-4}\ 1/^{\circ}C$. It is assumed that the domain is isohaline. Prandtl number is taken as 7.2. This allows the Brunt-Väisälä frequency to be rewritten in terms of the temperature gradient.

$$N^2 = g\beta \frac{\partial T}{\partial z} \tag{5.3}$$

Assuming that $U_0 = 1\ m/s$, $D = 10\ m$, and $Fr = 31$, and using eqs. (5.1) and (5.3), the temperature gradient of the domain is $\partial T/\partial z = 0.00505\ ^{\circ}C/m$. Additionally, Reynolds number based on the wake diameter $Re = U_0 D/\nu$ is 10^7.

In addition to providing a source of mean kinetic energy and turbulent kinetic energy, the wake provides an initial source of potential energy in the form of a mixed patch. Vigorous turbulent mixing in the wake is assumed to bring cooler water upwards and send warmer water downwards, creating a disturbance to the otherwise linear temperature stratification.

NZM wake centered $y = z = 0\ m$, the equations below define the initial conditions for the case:

$$\begin{aligned}
&\delta\vec{U}(y,z) = \left(-U_{D0}\left(1 - 8(y^2+z^2)/D^2\right)\exp\left(-8(y^2+z^2)/D^2\right), 0, 0\right)^T \\
&k_b = 2.4 \times 10^{-9}\ m^2/s^2 \\
&\delta k(y,z) = k_0 \exp\left(-4(y^2+z^2)/D^2\right) \\
&k = k_b + \delta k \\
&\epsilon_b = 1.9 \times 10^{-11}\ m^2/s^2 \\
&\delta\epsilon(y,z) = \sqrt{12}\left(k_0^{3/2}/D\right)\exp\left(-6(y^2+z^2)/D^2\right) \\
&\epsilon = \epsilon_b + \delta\epsilon \\
&T_b(z) = T_0 + \frac{dT}{dz}z \\
&\delta T(y,z) = -\frac{dT}{dz}z\exp(-4(y^2+z^2)/D^2)
\end{aligned}$$

U_{D0} and k_0 are set from data provided by Hassid (1980a) and assumed to be $U_{D0} = 0.16U_0$ and $k_0 = 0.0036U_0^2$. Background k and ϵ values are taken assuming the background fluid has weak turbulence. For the total solver, the minimum possible k and ϵ values are k_b and ϵ_b, respectively. This forms a floor for those values. In the delta solver, the floor values for δk and $\delta\epsilon$ are set to 10^{-16} while the background k and ϵ values remain fixed. Effectively, the minimum k and ϵ values are nearly the same between the two simulations.

Solver Settings

The buoyant form of the $k-\epsilon$ turbulence model is used for both cases, the model being in total form for the total solver simulation and the model being in delta form for the delta solver simulation. Second order schemes are used for time and space discretization, with the `fvSchemes` file for both cases shown below.

```
ddtSchemes
{
default backward;
}

gradSchemes
{
default Gauss linear;
}

divSchemes
{
default Gauss linearUpwind Gauss linear 0.1;
div(phi,U) Gauss filteredLinear2V 0.2 0;
div(phi,delU) Gauss filteredLinear2V 0.2 0;
div(phi,UBackground) Gauss filteredLinear2V 0.2 0;
div(phiBackground,UBackground) Gauss filteredLinear2V 0.2 0;
div((nuEff*dev2(T(grad(U))))) Gauss linear;
div((nuEffBackground*dev2(grad(UBackground).T()))) Gauss linear;
div((nuEff*dev2(grad((UBackground+delU)).T()))) Gauss linear;
div(phi,delk) Gauss linearUpwind Gauss linear 0.1;
div(delphi,kBackground) Gauss linearUpwind Gauss linear 0.1;
div(phi,delepsilon) Gauss linearUpwind Gauss linear 0.1;
div(delphi,epsilonBackground) Gauss linearUpwind Gauss linear 0.1;
div(phi,delT) Gauss linearUpwind Gauss linear 0.1;
div(delphi,TBackground) Gauss linear;
}

laplacianSchemes
{
default Gauss linear uncorrected;
}

interpolationSchemes
{
default linear;
```

```
}

snGradSchemes
{
default uncorrected;
}

fluxRequired
{
default no;
p;
delp;
phi;
}
```

The settings for the `fvSolution` file for total solver case is shown below, with the linear equation solvers, matrix preconditions, and smoothers for each variable included. Here, `GAMG` means geometric applomerated algebraic multigrid, `DIC` refers to a diagonal incomplete-Cholesky, `DILU` refers to diagonal incomplete-LU, `PBiCGStab` refers to preconditioned bi-conjugate gradient (stabilized). The same solver settings are used for the equivalent delta field for the delta solver case.

Table 5.1: NZM decay run solver settings.

Variable	Solver	Preconditioner	Smoother	Tolerance	Relative Tolerance
`p`	`GAMG`	`GAMG`	`DICGaussSeidel`	1×10^{-8}	0
`U, k, epsilon`	`PBiCGStab`	`DILU`	`DILUGaussSeidel`	1×10^{-10}	0
`T, S`	`PBiCGStab`	`DILU`	`DILUGaussSeidel`	1×10^{-12}	0

No non-orthogonal correctors are used, and three corrector loops, also called inner loops, are used in each case. This means that the `pEqn.H` file is run three times per timestep.

5.1.2 Results and Discussion

Centerline decay of wake velocity and turbulent kinetic energy are shown in figs. 5.3 and 5.4, respectively. The curves for the total and delta solver follow the same line, showing that results are nearly identical between the two solvers. Additionally, the general trend of the decay of U_D and k are captured by these simulations, showing that the solvers can produce physically-realistic results, allowing them to be used to a reasonable extent in stratified wake

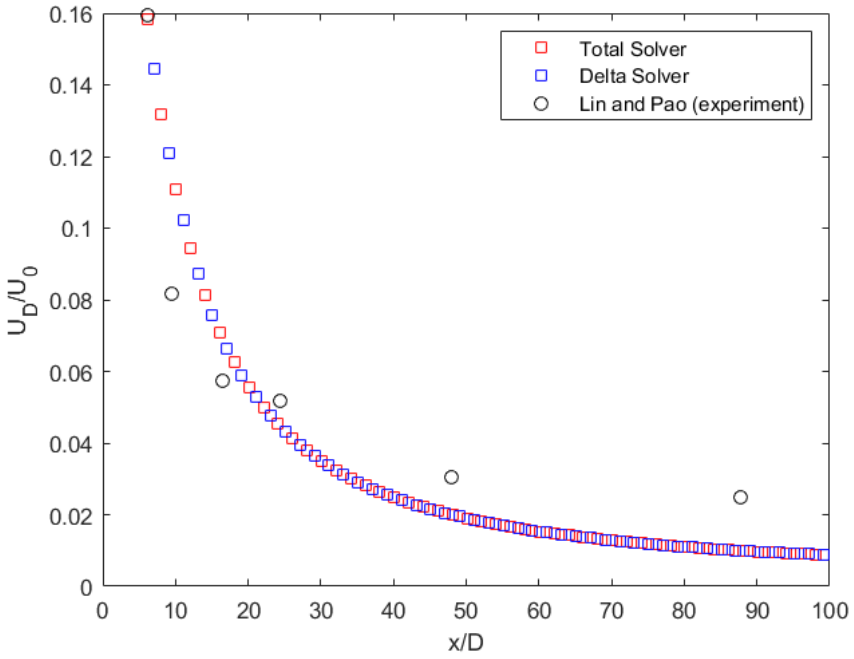

Figure 5.3: Centerline decay of velocity

simulations, at least before the onset of horizontal eddies which the 2D+t domain would be unable to resolve. Centerline velocity defect settles towards a lower value compared to experimental results, similar to the results of non-stratified wake simulations seen in the work of Hassid (1980a). Possible causes of this discrepancy include the choice of an isotropic turbulence model, as non-stratified turbulence is isotropic but stratified turbulence may not be. The RANS formulation itself may also be a cause of the discrepancy. Centerline turbulent kinetic energy is elevated compared to experimental results, but this was also encountered by Hassid in both non-stratified and stratified simulations, who points out that the experimental data measures only a single component of turbulent velocity fluctuations. In spite of these small differences, general trends are captured well and quantitative values are similar to experimental results, meaning that this model can be used to simulate stratified wakes to a reasonable extent.

5.2 Wave-Wave Interaction

A further case is run to ensure that all non-linear interactions between the background and perturbation are correctly implemented. Javam, Imberger, and Armfield (2000b), which studies wave-wave interactions through a DNS-type approach is used as a basis for this case, particularly its forcing function used to generate internal gravity waves. The approach to this

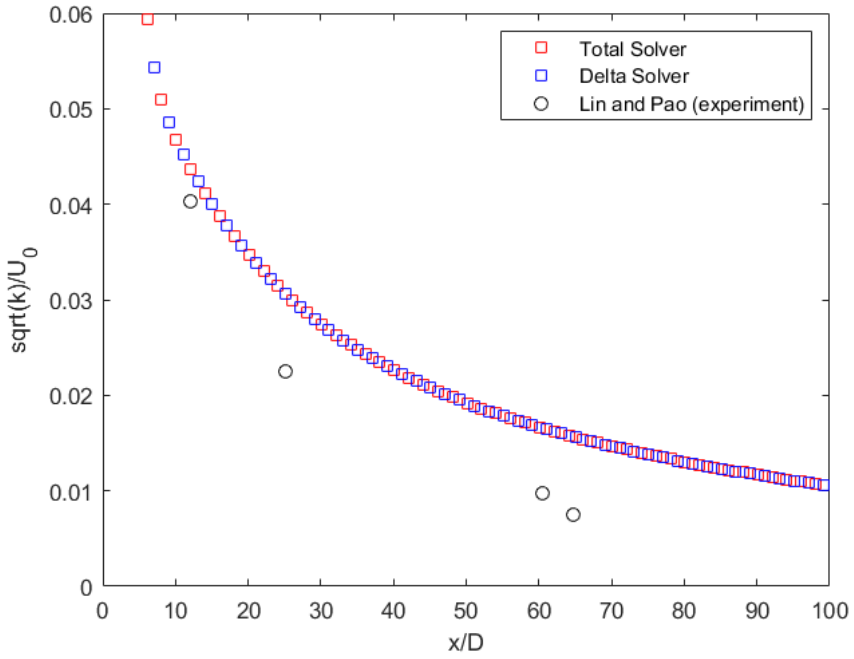

Figure 5.4: Centerline decay of turbulent kinetic energy

simulation is as follows: two internal gravity wave-makers are placed within a domain and these wave-makers will radiate IGWs in a known pattern. These wave-makers are placed at the same z level and spaced apart such that radiated IGW beams from both wave-makers will intersect each other at two points. Here, any wave-wave interactions can be observed. This simulation setup is repeated twice, once with both wave-makers acting on the perturbation fields so that all IGWs are located in the delta fields, and a second simulation where one wave-maker affects the background fields only and the second affects the perturbation fields only. The background-only wave-make is simulated on its own for 12 wave periods to generate time-constant background conditions. These background fields are mapped to another simulation on the same domain which sees the other wave-maker activated to generate IGWs within the delta fields. These IGWs in the delta fields may interact with the background IGWs such that any non-linear interaction will affect the evolution of the delta fields only. The resulting total fields from these two simulations are then compared, ideally being very close to each other barring any slight differences that will arise from treating the background IGW beams as constant in time. This will ensure that the non-linear terms in the delta solver and turbulence model are implemented correctly, as well as gauge how much a lack of scale separation between the background and perturbation fields may affect the solution.

5.2.1 Case Setup

This simulation takes place on a 150 m wide and 125 m tall 2D domain with beaches around the edges. 300,000 uniformly-sized cells make up the domain, and outlet-type boundary conditions are used on the four domain boundaries. The `empty` boundary condition is used on the $\pm x$ faces to force OpenFOAM to treat the domain as strictly 2D.

The forcing function is given by Javam, Imberger, and Armfield (2000b) in the form seen in eq. (5.4). This function is added to the right hand side of the vertical component of the conservation of momentum equation such that the forcing function vector is $(0, 0, f(y, z))$. This function, when added to the governing equations of the system, generates an "X"-shaped internal gravity wave beam of angular frequency ω and horizontal wavenumber k_x; F is an amplitude which is originally determined through a set of non-dimensional equations in Javam, Imberger, and Armfield (2000b).

$$f(y,z) = \begin{cases} F \sin(\omega t) \cos\left(k_x \left(y - y_{center}\right)\right) \times \\ \quad \exp\left(-300 \left|k_x \left(z - z_{center}\right)\right|^3\right) & \forall\ k_x \left(y - y_{center}\right) \in \left[-\frac{3\pi}{2}, \frac{3\pi}{2}\right] \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

Density stratification is realized through a temperature stratification, with $N = 5$ cycles per hour. The same `fvSchemes` and `fvSolution` files as those described in section 5.1.1 are used in these simulations. ω, k_x, and F are determined by this system of non-dimensional numbers given by Javam, Imberger, and Armfield (2000b), where the Keulegan number Ke is given as the ratio of the IGW period to the eddy roll-up period.

$$Ri = \frac{N^2}{\omega^2} = 2 \tag{5.5}$$

$$Re = \frac{\omega}{k_x^2 \nu} = 25{,}000 \tag{5.6}$$

$$Ke = \frac{F k_x}{\omega^2} = 10 \tag{5.7}$$

Given N and $\nu = 10^{-6}\ m^2/s$, $F = 4.9 \times 10^{-5}\ m/s^2$, $\omega = 6.17 \times 10^{-3}$ rad/s, and $k_x = 1.24$ rad/m. The case is run for 12,000 seconds, or nearly 12 wave periods. For the case which examines background-perturbation interactions, the background wave-maker is run for the entirety of the simulation then its results are copied onto an identical domain as the background conditions. The wave-maker acting on the perturbation fields then is activated and the simulation run again. The wave-maker on the $-y$ half of the domain is used to generate the background IGWs while the one on the $+y$ half of the domain is used to generate the IGWs in the perturbation fields. For the second case which has both wave-makers acting on the perturbation field, the simulation is only run once. This second simulation is treated

as a "ground truth" simulation, as both wave-makers are actively simulated and all IGWs change in time and interact with other IGWs.

5.2.2 Results and Discussion

At the end of the simulations, the total velocity fields are compared between the two simulation. The results for velocity magnitude, y component, and z component for the background-perturbation case at the final timestep are given in figures 5.5, 5.6, and 5.7, respectively. These values are normalized by F/ω as is done in Javam, Imberger, and Armfield (2000b). Figure 5.8 gives the total eddy viscosity for the same case at the final timestep, here normalized by molecular viscosity. The simulation where both wave-makers are acting on the perturbation fields is designated the "true" results. The difference in velocity magnitude and components is defined as the $\vec{U} = \vec{U}_b + \delta\vec{U}$ minus $\vec{U}_{true}$. This difference, prefaced with a Δ, is then normalized by either the maximum of the true velocity magnitude or corresponding component, or in the case of eddy viscosity, by the molecular kinematic viscosity. Figures 5.9 through 5.11 show the differences between the two simulations in the velocity magnitude, y component, and z component, respectively, at their final timesteps. The results are favorable, showing differences of up to 6% for velocity magnitude and components, primarily in the vicinity of the wave-maker which acts on the perturbation field in the background-perturbation interaction case. This may be due to the background field in this simulation containing developed IGWs which affect the IGWs in the perturbation fields as they form. Figure 5.12 shows the differences in eddy viscosity between the two simulations, and this is small compared to molecular viscosity, although this case is largely laminar. Given that there are no regions of spurious growth, the differences between the two cases can likely be explained by the failure of the one-way interaction between the perturbation and background in the absence of the necessary scale disparity assumed by the multi-scale localized perturbation method, and not by any implementation errors.

5.3 Net-Zero Momentum Wake in Non-Uniform Background Shear

A more complicated stratified wake case can be pursued with the multi-scale localized perturbation method, specifically one with a non-uniform background shear layer. The case of a stratified wake in a uniform background shear has been simulated by Chernykh and Voropaeva (2015), but the ability to specify any physically-consistent background current cannot be understated. The multi-scale localized perturbation method gives researchers the capability to quickly add a non-uniform background current to the domain and observe the development of the velocity field associated with the wake, as well as capturing all interactions of the wake and any subsequent IGWs with the background shear, such to those

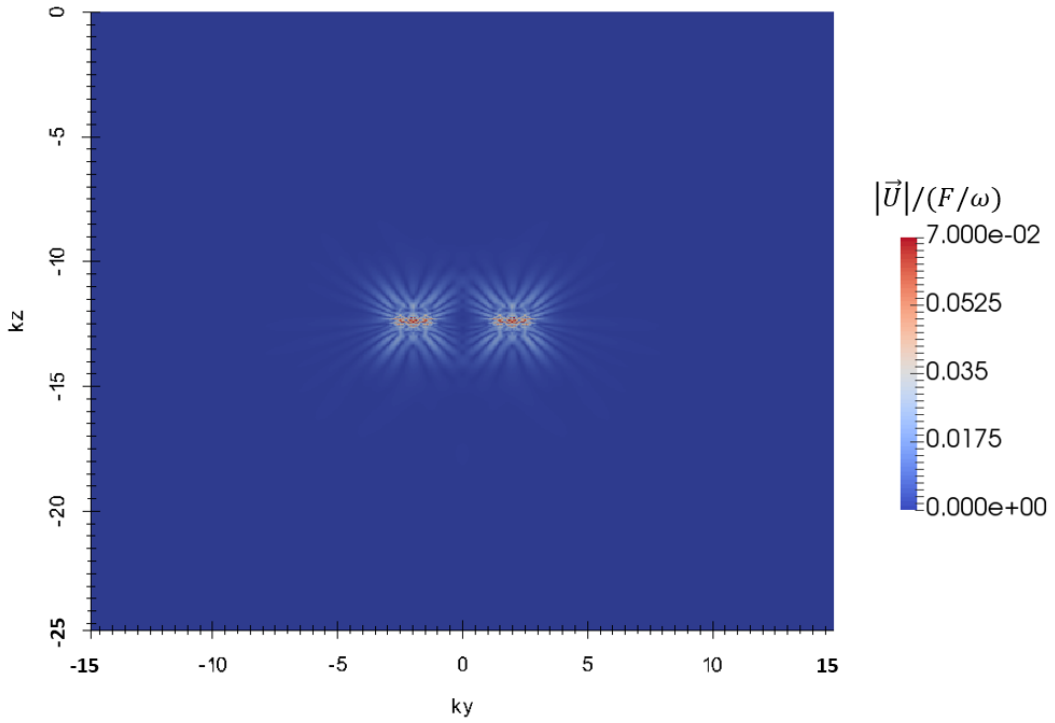

Figure 5.5: Magnitude of total velocity normalized by F/ω.

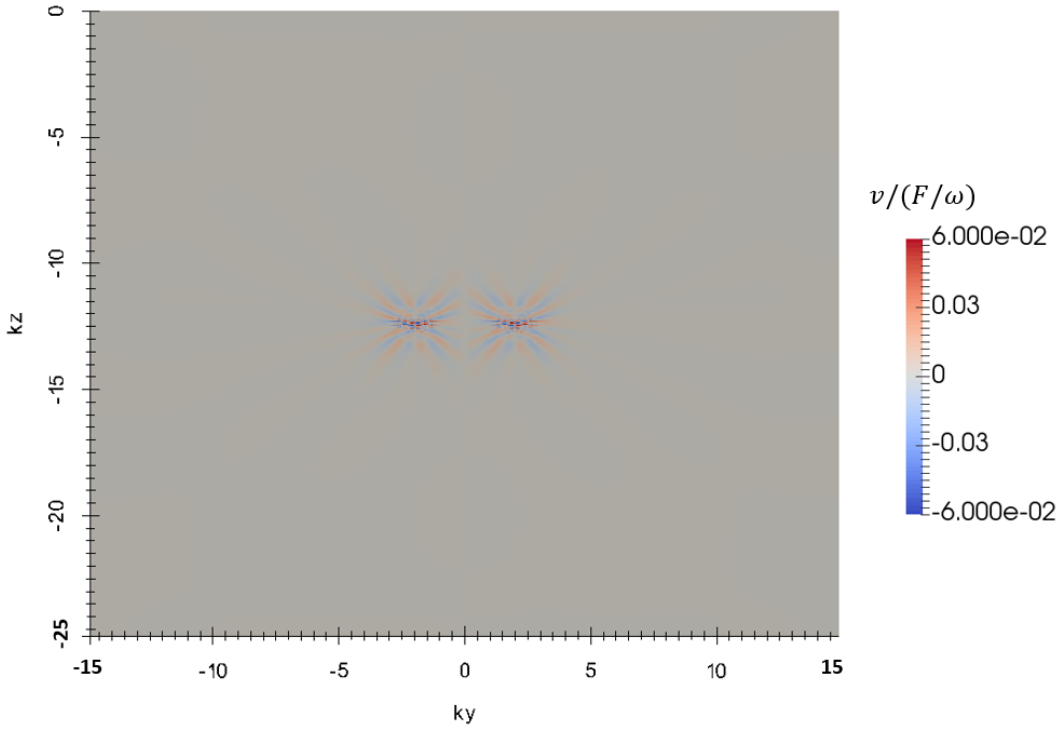

Figure 5.6: Total y component of velocity normalized by F/ω.

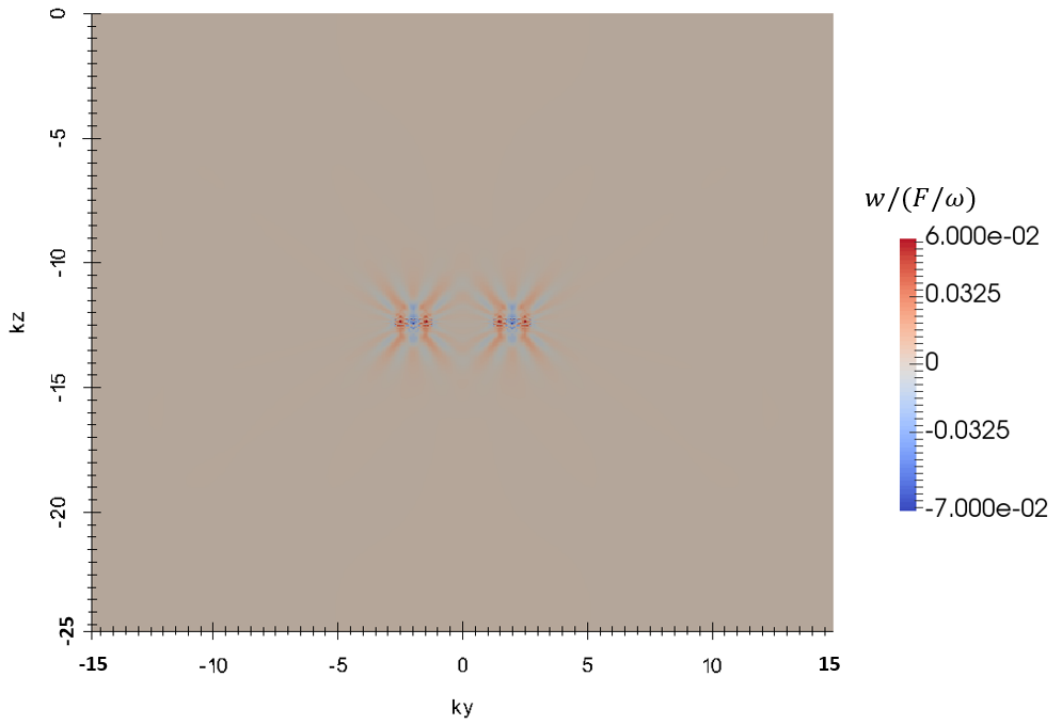

Figure 5.7: Total z component of velocity normalized by F/ω.

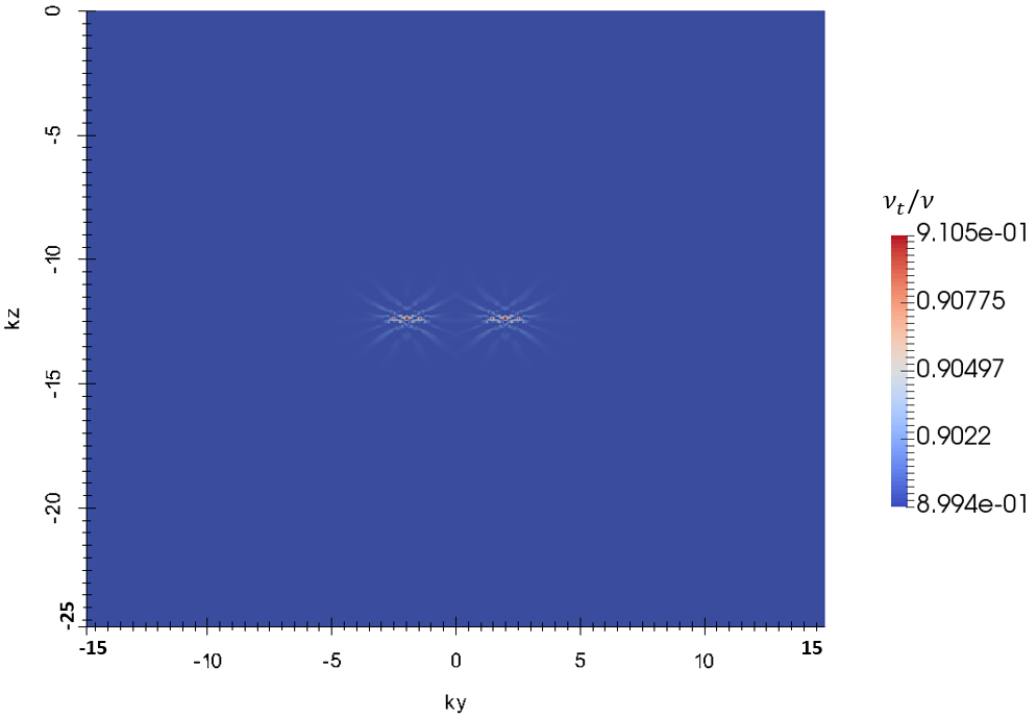

Figure 5.8: Total eddy viscosity normalized by molecular viscosity.

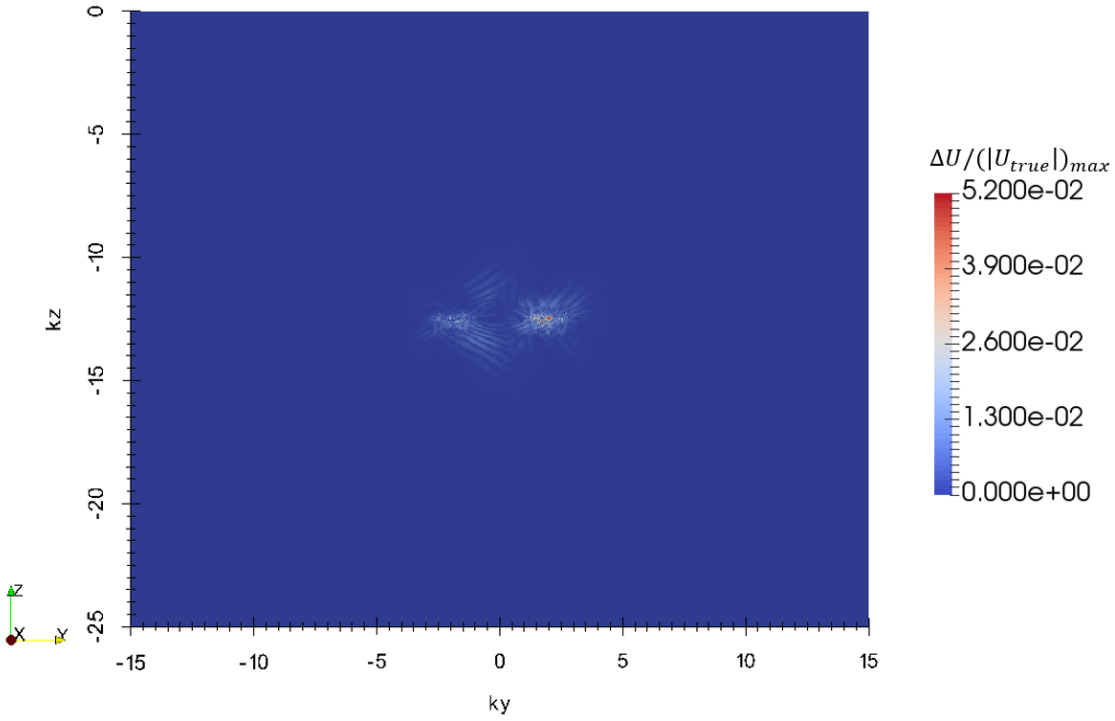

Figure 5.9: Difference in magnitude of velocity.

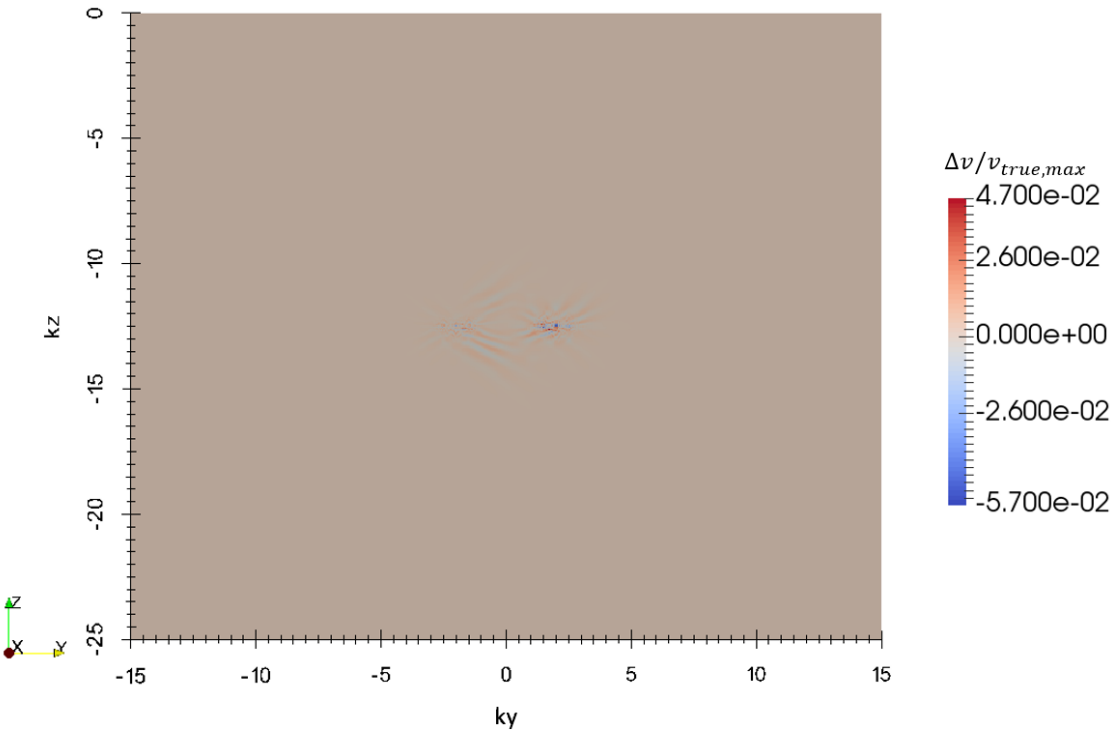

Figure 5.10: Difference in y-component of velocity.

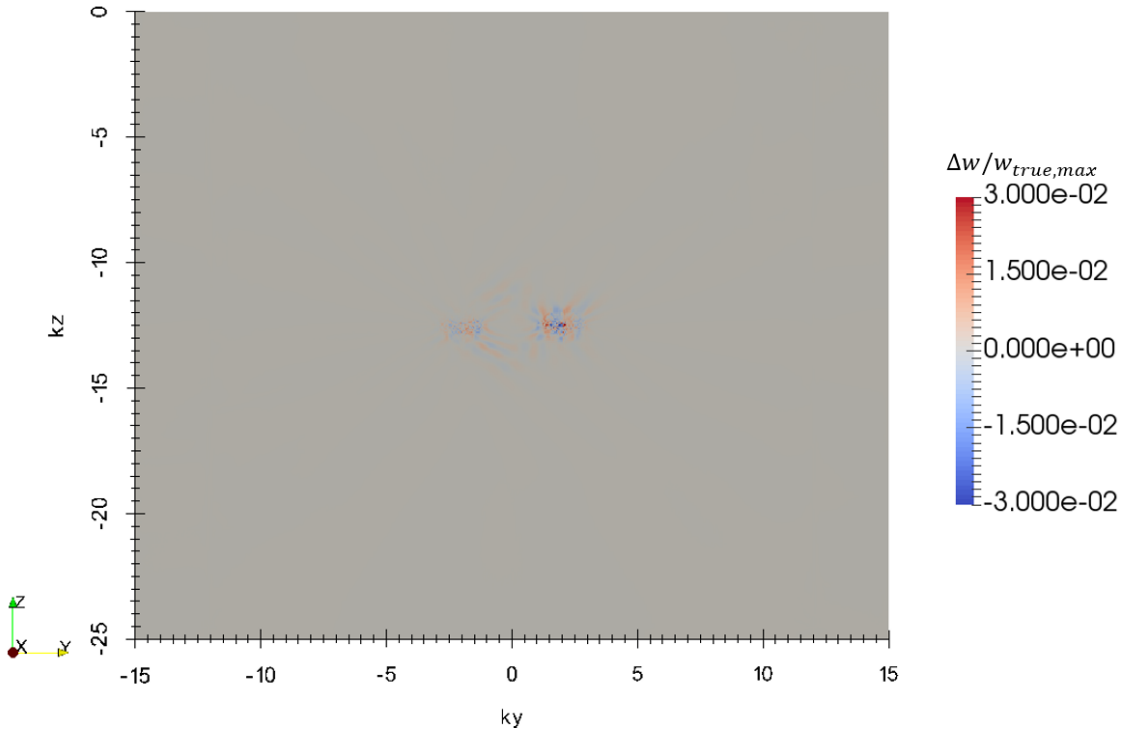

Figure 5.11: Difference in z-component of velocity.

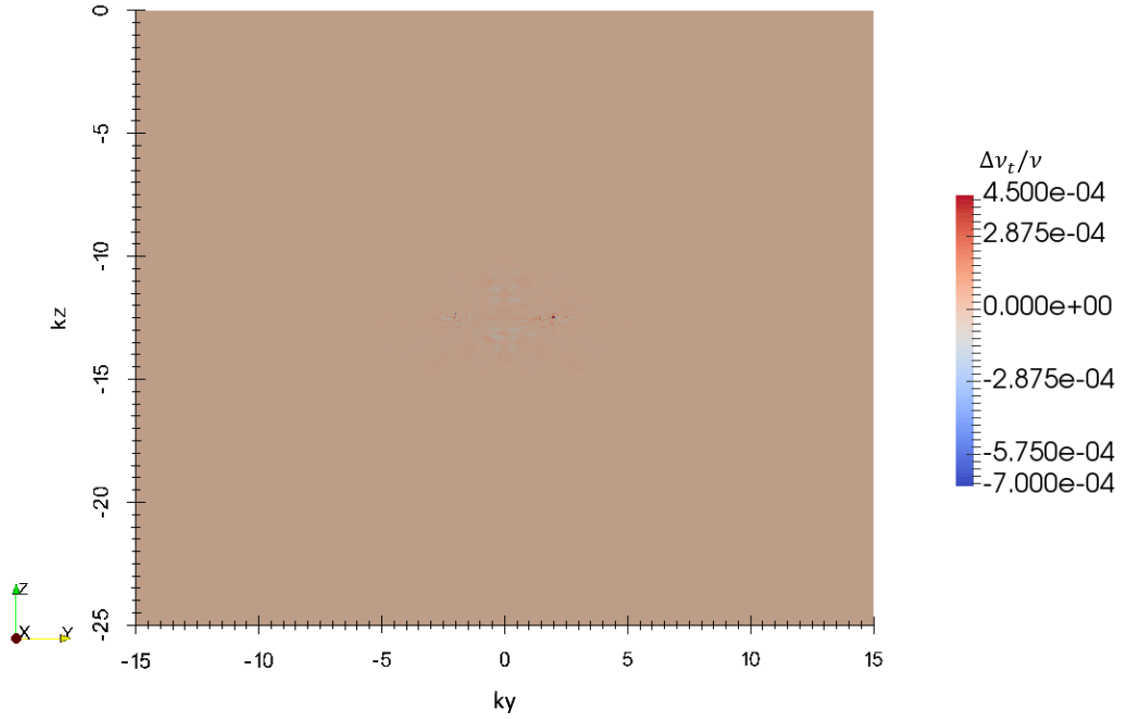

Figure 5.12: Difference in eddy viscosity.

observed in numerical simulations by Javam, Imberger, and Armfield (2000a).

5.3.1 Case Setup

The setup for this simulation closely follows that described in section 5.1.1, the difference being the addition of a 1D simulation to generate a background shear layer which is then interpolated spatially onto the 2D+t domain, then held constant in time as described in section 3.1.1. The domain is 500 m long in the z direction as in the previous case, but the domain is 1 km wide. A beach is applied on the 125 m of the domain closest to the lateral boundaries. There are 883,927 cells within the domain, with 751 in the vertical direction as before, 26 cells in the y direction in each section of the beach, and 1,125 cells in the horizontal direction within the center of the domain. The domain is 1 cell thick in the x direction.

1D Precursor Simulation

The 1D precursor simulation is done using the same `deltaBuoyantBoussinesqPimpleFoam` solver, but setting the background velocity, pressure, and temperature fields to 0 and the background k and ϵ fields to very small non-zero constants. This setup returns the delta form solver to its total form equivalent. Once this has been completed, a shear layer is initialized in the $\delta\vec{U}$ field, then the 1D simulation run in order to develop a turbulence field. The shear layer spans 40 m and is centered at $z = 50$ m. Equation (5.8) gives the initial shear layer velocity profile and is of an arbitrarily-selected form, and eq. (5.9) gives the temperature profile of the simulation. The temperature at the center of the domain $T(z = 0) = 20°C$.

$$\delta\vec{U} = (0, -0.1 + 25\sqrt{-g\beta(dT/dz)}\sinh^{-1}\left(\frac{2z}{15\sqrt{3}}\right), 0) \tag{5.8}$$

$$\frac{dT_b}{dz} = 0.00505 \tag{5.9}$$

The 1D simulation is run until the lower edge of the shear layer has a minimum Richardson number of nearly 0.25. This will allow IGW-shear interactions to be observed in the delta fields once the 1D precursor simulation results are interpolated to the 2D+t domain. The same `fvSchemes` and `fvSolutions` files from section 5.1.1 are used in this simulation, except for the tolerance on delta pressure is reduced to 10^{-7}; the same change is made for the following 2D+t simulation. Figure 5.13 shows the initial and final velocity profiles of the 1D simulation, and fig. 5.14 shows the final $\delta\nu_t$ profiles from the same simulation. Figure 5.15 shows the final $1/Ri$ profile from the 1D simulation, showing two regions of low Richardson number flow. Within these regions, we may be particularly likely to see wave-shear layer

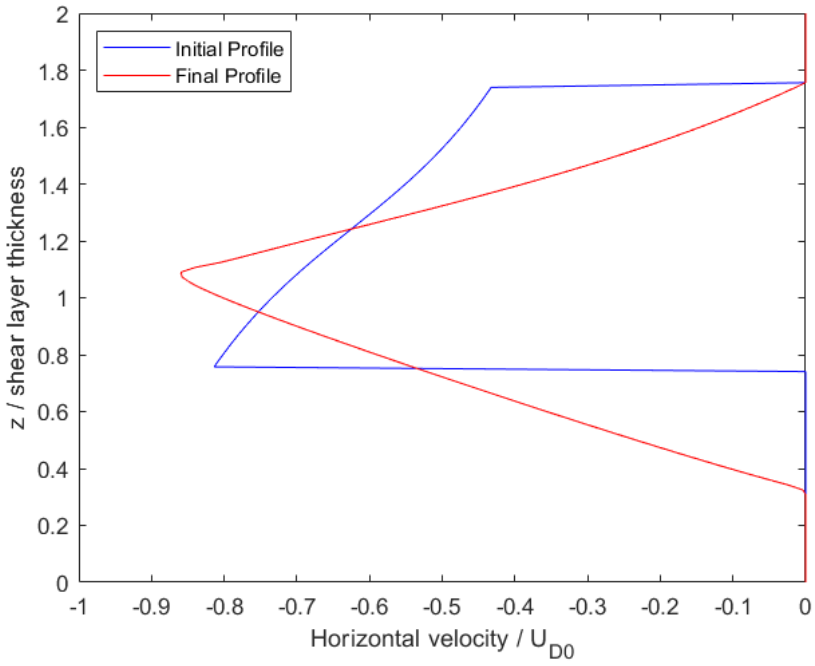

Figure 5.13: Initial and final shear layer velocity profiles; becomes y component of $\vec{U}_b$ in the 2D+t simulation.

interaction, with the smaller one at the top of the shear layer being below the critical Richardson number of 1/4.

2D+t Simulation

Once the 1D simulation is complete, the results are interpolated to the 2D+t domain using `mapFields` and set to the background fields, and the wake simulation is run for 3600 seconds, or $Nt = 11.6$. The same `fvSchemes` and `fvSolution` file is used from the 1D simulation. The initial equations describing the wake in section 5.1.1 are used to initialize the wake in the perturbation fields. The same beach layout used in the aforementioned case is used, noting that the beach will only affect the delta variables.

2D+t Simulation - Identically Zero Initial Perturbation

An additional simulation is performed using the same 2D+t domain with background conditions taken from the 1D precursor simulation. This simulation is done without initializing any perturbation fields to ensure that no perturbation is generated in the absence of any initial perturbation. The results of this simulation confirm this is the case, as no perturbation

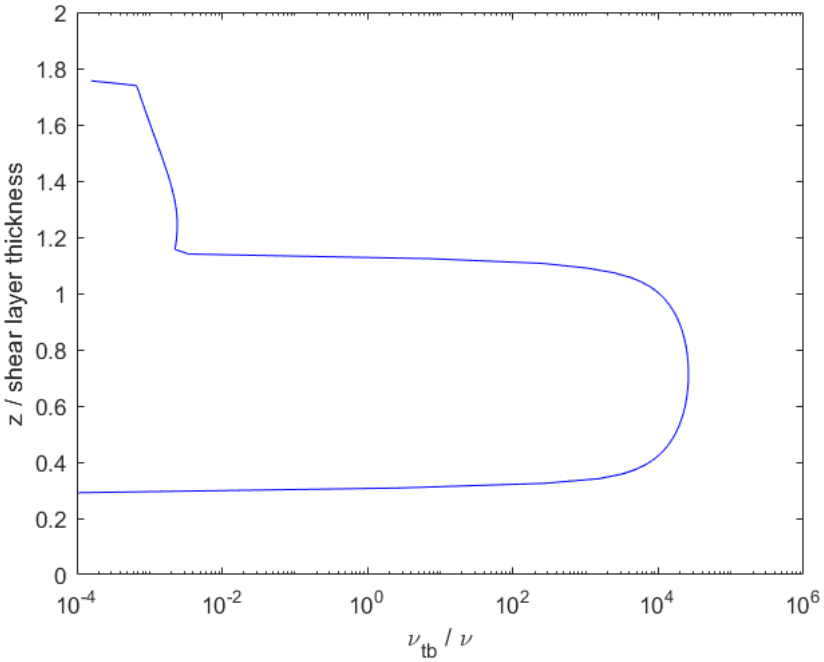

Figure 5.14: Final shear layer eddy viscosity.

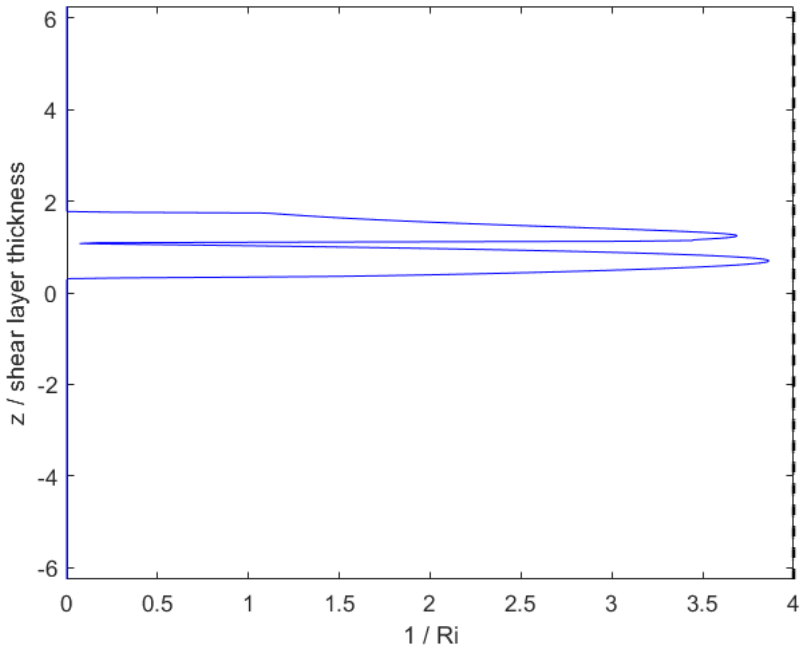

Figure 5.15: Vertical $1/Ri$ profile for the background shear profile.

is seen anywhere within the domain across any timesteps.

5.3.2 Results and Discussion

At the end of the 3600 seconds of simulation time, data are extracted from the 2D+t domain, including the perturbation velocity $\delta\vec{U}$ and the perturbation to eddy viscosity $\delta\nu_t$, the latter being an approximate indicator of turbulent activity as modeled by the RANS turbulence model. Figure 5.16 shows the magnitude of the perturbation velocity at the final timestep, and figures 5.17 through 5.19 show the x, y, and z components of the perturbation velocity, respectively. All aforementioned figures are normalized by the initial wake centerline velocity defect U_{D0}. The decaying wake is clearly visible in the center of the domain in the magnitude and x-component of $\delta\vec{U}$. The thermally-mixed region within the NZM in conjunction with the ambient stratification produce internal gravity wave which radiate in the y and z directions. When an internal gravity wave reaches the shear layer, it is immediately affected by it, demonstrating the one-way non-linear interaction between the background and delta fields. The internal gravity waves are weak compared to the shear layer, and are advected downstream by it. The increased background eddy viscosity along the top and bottom of the shear layer greatly diffuse the waves within the shear layer, yet a wave-like pattern is still retained as shown by figure 5.19. This type of wave-shear interaction was noted by Javam, Imberger, and Armfield (2000a), who found that particular combinations of internal gravity waves would pass through shear layers without becoming "trapped". While most wave-shear layer interaction occurs downstream of and above the wake, some interaction takes place upstream as seen in the perturbation eddy viscosity in figure 5.20. It should be noted that the IGWs are essentially laminar and advect negligible amounts of $\delta\nu_t$ into the shear layer, so this growth in eddy viscosity comes from interaction between the internal gravity waves in the delta fields and the shear layer in the background fields. A second region of low Richardson number also lies at the top of the shear layer, and wave-shear layer interactions are seen here as well, but to a smaller degree.

It should be noted, however, that no quantitative conclusions should be drawn from this case, as it has been manufactured as a demonstration case for the multi-scale localized perturbation method. The benefits of the multi-scale localized perturbation method are visible nonetheless; one can easily separate the phenomena associated with the shear layer and those associated with the wake, its collapse, and subsequent internal gravity wave generation while retaining the one-way interaction between these fields. Boundary conditions and sponge layers are easily implemented and can be applied only to the perturbation fields. If one were to simulate both the background components and perturbation components in a one-way coupled manner on the same domain as described in section 3.1.1, one could impose an inlet boundary condition for the background shear layer on the $+y$-normal boundary while imposing a wave-transmissive outflow boundary condition for the perturbation quantities on this same boundary. This would prevent any internal gravity waves from impinging on the inlet boundary condition and potentially reflecting back into the domain. Similarly, a

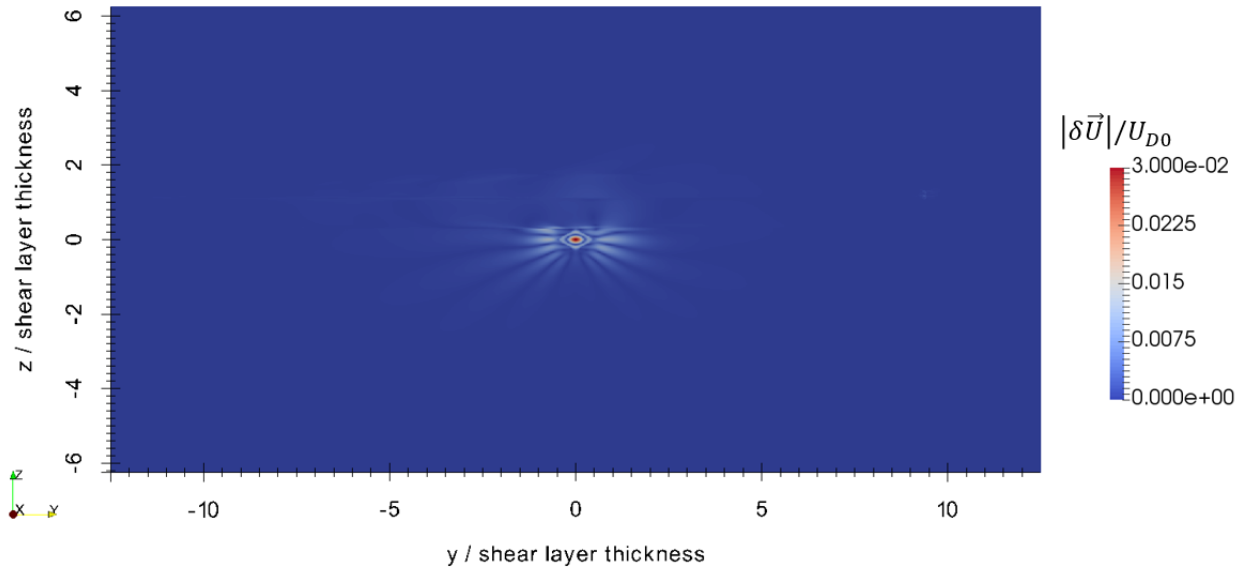

Figure 5.16: Magnitude of $\delta\vec{U}$ at $t = 3600$ seconds.

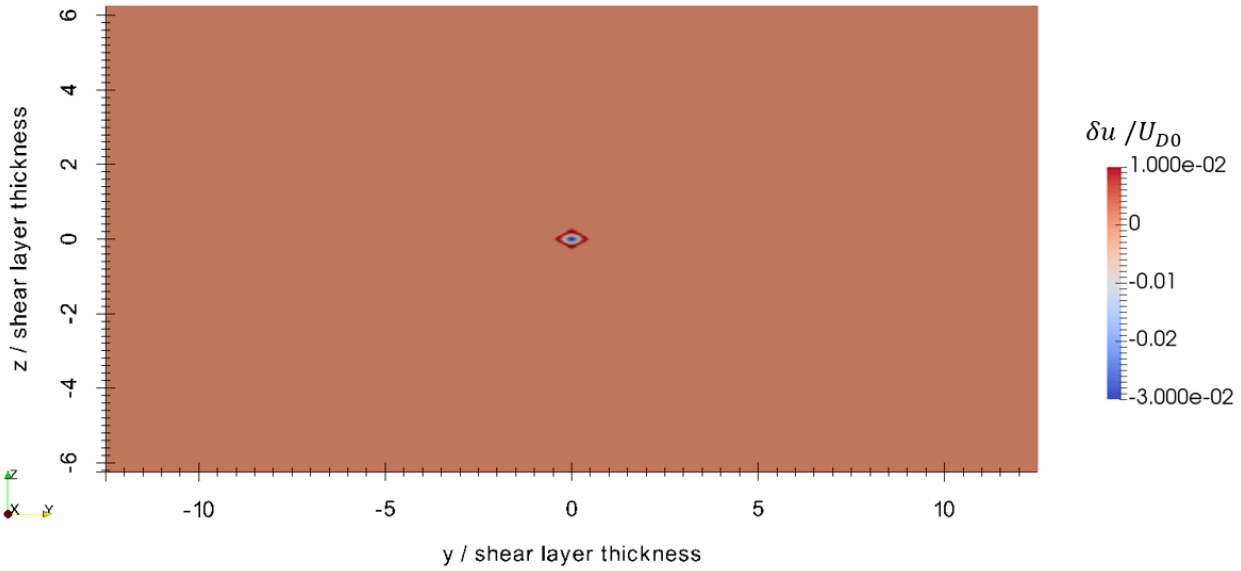

Figure 5.17: x component of $\delta\vec{U}$ at $t = 3600$ seconds.

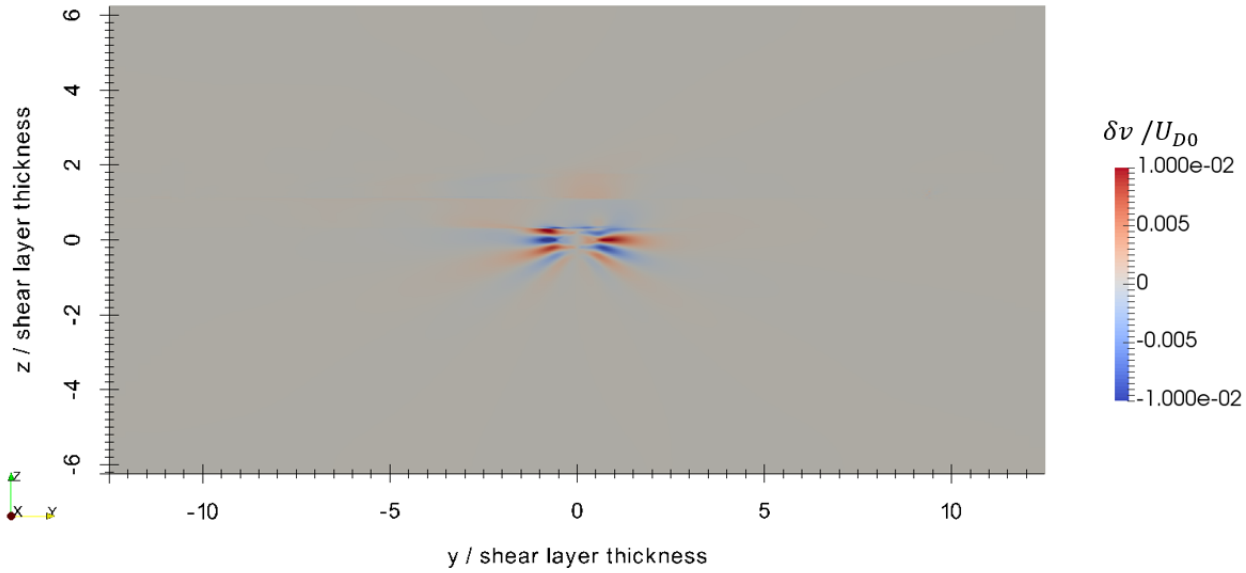

Figure 5.18: y component of $\delta\vec{U}$ at $t = 3600$ seconds.

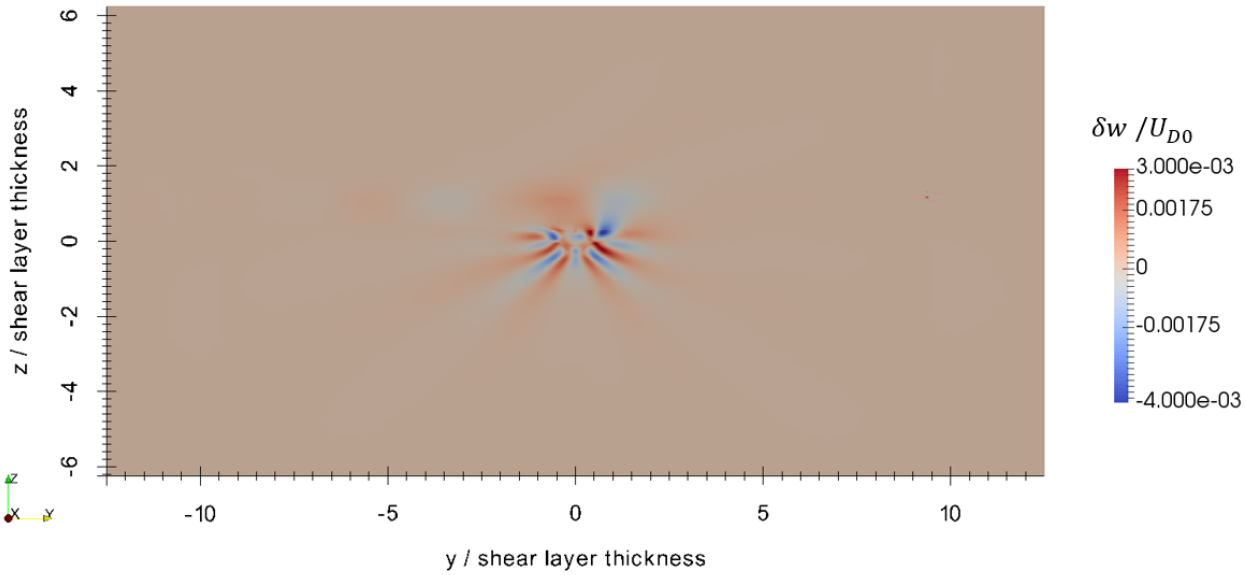

Figure 5.19: z component of $\delta\vec{U}$ at $t = 3600$ seconds.

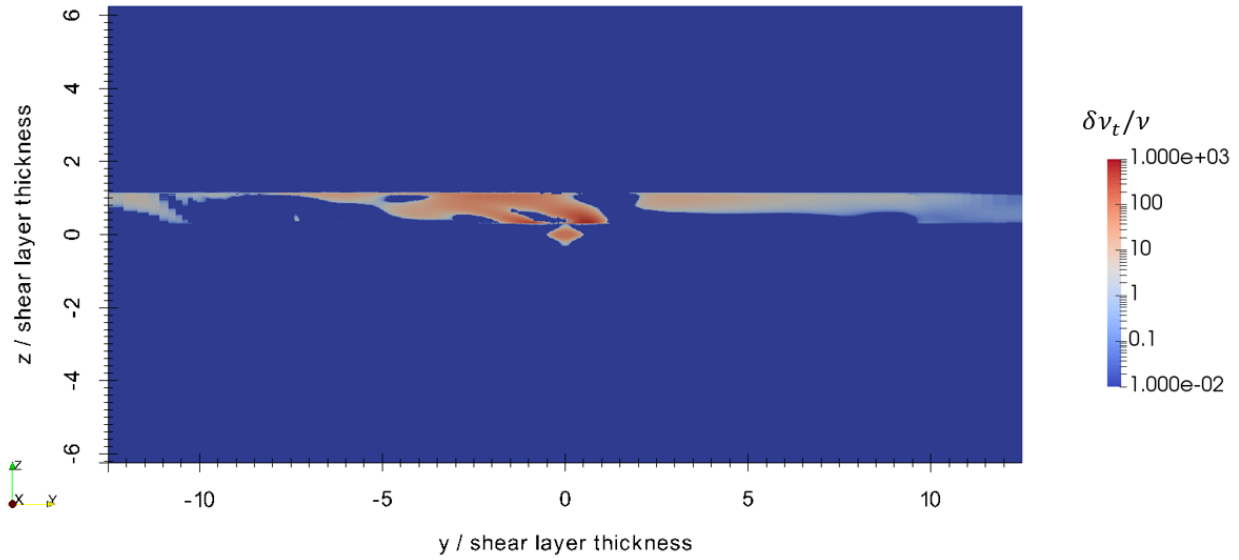

Figure 5.20: $\delta\nu_t$ at $t = 3600$ seconds.

sponge layer would only affect the perturbation, allowing the inlet to act unimpeded. The multi-scale localized perturbation method allows users to simulate more complicated flows than they otherwise would be able to, in particular including information from large-scale simulation within a small-scale domain in which a perturbation is introduced.

Chapter 6

Conclusions

6.1 Summary of Contributions

Derivation of an alternative formulation of the governing equations of fluid dynamics allows for the decomposition of simulation fields into a background component and a perturbation from this background. This decomposition, called the multi-scale localized perturbation method or the delta form for short, retains complex physics of the original governing equation, provides simplified simulation setup, and enables visualization of complex phenomena as it evolves in time and space. Once the decomposed variables are substituted into the original governing equation, the new system must be closed. The assumption of scale separation between the background and perturbation is used to close this new system of governing equations are discussed, and while is assumption may not be valid for a general fluid flow, they are reasonable for many forms of geophysical flows found throughout the atmosphere and ocean. Mathematically, this assumption results in background fields evolving as if the perturbation were not present, and the perturbation fields evolving under their own dynamics as well as the one-way, non-linear interaction of the background acting upon the perturbation. This process is repeated for as many governing equations as there are present for the original set of equations, and can be extended to any additional algebraic or differential equations.

Three separate solution methods are highlighted, giving the delta form great flexibility in solving problems. The delta form of the incompressible RANS equations, turbulence model equations, and scalar transport equations is documented and implemented within OpenFOAM, with a line-by-line breakdown showing how the mathematical equations are broken down and translated into OpenFOAM source code. Care is taken to ensure that the operators selected complement the linear solvers used to solve the system. Verification simulations of a stratified wake show that the total form solver and the delta form solver produce nearly identical results for cases without a background fields, and these results replicate simulation results from Hassid (1980a). Additional simulations show that differences between

simulations with background-perturbation interactions and no background-perturbation interactions are small and can be attributed to the breakdown of the assumption of scale dissimilarity rather than implementation errors.

Finally, a stratified wake case with a non-trivial background current is simulated to show the advantage of the multi-scale localized perturbation method by being able to quickly and cleanly separate the background and perturbation fields. A 1D vertical shear layer is generated within the domain, including associated turbulence fields, as the background conditions then a stratified wake is initialized as the initial perturbation conditions. As the case evolves, the wake collapses and generates internal gravity waves which propagate throughout the domain and into the background shear layer. This one-way wave-shear layer interaction between the background and perturbation fields is retained, meaning that physically-relevant phenomena are captured and manifest themselves in the perturbation fields, in this case being demonstrated within the delta turbulence fields as enhanced turbulence generation is evidence of this non-linear interaction. The perturbation to the velocity field shows that the internal gravity waves within the shear layer have their trajectory influenced by the background shear current. These cases are simplified and demonstrative, but show the great potential of the multi-scale localized perturbation method for more complex cases, including those with temporally and spatially-varying background fields.

6.2 Future Work

Extensions to the work presented in this thesis would include further applications of the multi-scale localized perturbation method, particularly to geophysical fluid problems. These problems feature disparate scales which is assumed in the derivation of the multi-scale localized perturbation method, making them natural next steps. In particular, multi-scale simulations using background data generated from meso-scale or global-scale simulations would be an interesting demonstration of the capabilities of the multi-scale localized perturbation method, especially using time-varying and spatially-varying background fields. Simulations showcasing the simultaneous solutions of a background field and perturbation field solver as highlighted in the derivation could be performed as well. Extensions of the multi-scale localized perturbation method to other turbulence models, including large-eddy simulation models, and different governing equations could be tested to examine its potential applicability.

The loosening of the assumption of disparate scales is also a further step, such as those which allows for more scale similarity between the background and perturbation. When coupled with the simultaneous solution of the background and perturbation, this might entail background-perturbation interactions affecting not just the perturbation field but also the background field. This approach may benefit numerical solutions to the turbulence model equations, as there would be much potential scale similarity between the background and perturbation turbulence.

Bibliography

Abarbanel, Henry D. I. et al. (1984). "Richardson Number Criterion for the Nonlinear Stability of Three-Dimensional Stratified Flow". en. In: *Physical Review Letters* 52.25, pp. 2352–2355. DOI: 10.1103/PhysRevLett.52.2352.

Abdilghanie, Ammar M. and Peter J. Diamessis (2013). "The internal gravity wave field emitted by a stably stratified turbulent wake". In: *Journal of Fluid Mechanics* 720, 104–139. DOI: 10.1017/jfm.2012.640.

Adcroft, Alistair et al. (2004). "Implementation of an Atmosphere–Ocean General Circulation Model on the Expanded Spherical Cube". en. In: *Monthly Weather Review* 132.12, pp. 2845–2863. ISSN: 0027-0644, 1520-0493. DOI: 10.1175/MWR2823.1. URL: http://journals.ametsoc.org/doi/abs/10.1175/MWR2823.1 (visited on 02/05/2020).

Bigg, G. R. et al. (2003). "The role of the oceans in climate". en. In: *International Journal of Climatology* 23.10, pp. 1127–1159. ISSN: 0899-8418, 1097-0088. DOI: 10.1002/joc.926. URL: http://doi.wiley.com/10.1002/joc.926 (visited on 01/30/2020).

Blumberg, Alan F. and George L. Mellor (1987). "A description of a three-dimensional coastal ocean circulation model". en. In: *Coastal and Estuarine Sciences*. Ed. by Norman S. Heaps. Vol. 4. Washington, D. C.: American Geophysical Union, pp. 1–16. ISBN: 978-0-87590-253-1. DOI: 10.1029/CO004p0001. URL: http://www.agu.org/books/co/v004/CO004p0001/CO004p0001.shtml (visited on 01/29/2020).

Bonnier, Marion and Olivier Eiff (2002). "Experimental investigation of the collapse of a turbulent wake in a stably stratified fluid". en. In: *Physics of Fluids* 14.2, pp. 791–801. ISSN: 1070-6631, 1089-7666. DOI: 10.1063/1.1429963. URL: http://aip.scitation.org/doi/10.1063/1.1429963 (visited on 01/16/2020).

Brucker, Kyle A. and Sutanu Sarkar (2010). "A comparative study of self-propelled and towed wakes in a stratified fluid". en. In: *Journal of Fluid Mechanics* 652, pp. 373–404. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112010000236. URL: https://www.cambridge.org/core/product/identifier/S0022112010000236/type/journal_article (visited on 01/16/2020).

Chalamalla, Vamsi K. et al. (2013). "Turbulence during the reflection of internal gravity waves at critical and near-critical slopes". en. In: *Journal of Fluid Mechanics* 729, pp. 47–68. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/jfm.2013.240. URL: https://www.cambridge.org/core/product/identifier/S0022112013002401/type/journal_article (visited on 01/30/2020).

Chalamalla, Vamsi K et al. (2017). "SOMAR-LES: A framework for multi-scale modeling of turbulent stratified oceanic flows". en. In: *Ocean Modelling* 120, pp. 101–119. ISSN: 14635003. DOI: 10.1016/j.ocemod.2017.11.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500317301786 (visited on 01/29/2020).

Chen, Changsheng, Hedong Liu, and Robert C Beardsley (2003). "An Unstructured Grid, Finite-Volume, Three-Dimensional, Primitive Equations Ocean Model: Application to Coastal Ocean and Estuaries". en. In: *Journal of Atmospheric and Oceanic Technology* 20, p. 28. DOI: 10.1175/1520-0426(2003)020<0159:AUGFVT>2.0.CO;2.

Chernykh, G. G. and O. F. Voropaeva (2015). "Dynamics of a momentumless turbulent wake in a shear flow". en. In: *Journal of Engineering Thermophysics* 24.1, pp. 12–21. ISSN: 1810-2328, 1990-5432. DOI: 10.1134/S1810232815010026. URL: http://link.springer.com/10.1134/S1810232815010026 (visited on 01/27/2020).

Chernykh, G. G. et al. (2012). "On numerical modeling of the dynamics of turbulent wake behind a towed body in linearly stratified medium". en. In: *Journal of Engineering Thermophysics* 21.3, pp. 155–166. ISSN: 1810-2328, 1990-5432. DOI: 10.1134/S1810232812030010. URL: http://link.springer.com/10.1134/S1810232812030010 (visited on 01/21/2020).

Chernykh, G.G., N.P. Moshkin, and A.V. Fomina (2009). "Dynamics of turbulent wake with small excess momentum in stratified media". en. In: *Communications in Nonlinear Science and Numerical Simulation* 14.4, pp. 1307–1323. ISSN: 10075704. DOI: 10.1016/j.cnsns.2008.01.014. URL: https://linkinghub.elsevier.com/retrieve/pii/S1007570408000208 (visited on 01/23/2020).

Colella, Francesco et al. (2011). "Multiscale modeling of transient flows from fire and ventilation in long tunnels". en. In: *Computers & Fluids* 51.1, pp. 16–29. ISSN: 00457930. DOI: 10.1016/j.compfluid.2011.06.021. URL: https://linkinghub.elsevier.com/retrieve/pii/S004579301100209X (visited on 01/11/2020).

Danabasoglu, Gokhan et al. (2014). "North Atlantic simulations in Coordinated Ocean-ice Reference Experiments phase II (CORE-II). Part I: Mean states". en. In: *Ocean Modelling* 73, pp. 76–107. ISSN: 14635003. DOI: 10.1016/j.ocemod.2013.10.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500313001868 (visited on 02/01/2020).

Danabasoglu, Gokhan et al. (2016). "North Atlantic simulations in Coordinated Ocean-ice Reference Experiments phase II (CORE-II). Part II: Inter-annual to decadal variability". en. In: *Ocean Modelling* 97, pp. 65–90. ISSN: 14635003. DOI: 10.1016/j.ocemod.2015.11.007. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500315002231 (visited on 02/01/2020).

de Stadler, Matthew B. and Sutanu Sarkar (2012). "Simulation of a propelled wake with moderate excess momentum in a stratified fluid". en. In: *Journal of Fluid Mechanics* 692, pp. 28–52. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/jfm.2011.489. URL: https://www.cambridge.org/core/product/identifier/S0022112011004897/type/journal_article (visited on 01/21/2020).

Diamessis, Peter J., Geoffrey R. Spedding, and J. Andrzej Domaradzki (2011). "Similarity scaling and vorticity structure in high-Reynolds-number stably stratified turbulent

wakes". en. In: *Journal of Fluid Mechanics* 671, pp. 52–95. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112010005549. URL: https://www.cambridge.org/core/product/identifier/S0022112010005549/type/journal_article (visited on 01/21/2020).

Dommermuth, Douglas G. et al. (2002). "Numerical simulation of the wake of a towed sphere in a weakly stratified fluid". en. In: *Journal of Fluid Mechanics* 473, pp. 83–101. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112002002276. URL: https://www.cambridge.org/core/product/identifier/S0022112002002276/type/journal_article (visited on 01/17/2020).

Doney, S. C. et al. (2004). "Evaluating global ocean carbon models: The importance of realistic physics". en. In: *Global Biogeochemical Cycles* 18.3. ISSN: 08866236. DOI: 10.1029/2003GB002150. URL: http://doi.wiley.com/10.1029/2003GB002150 (visited on 02/01/2020).

Dörnbrack, Andreas, Thomas Gerz, and Ulrich Schumann (1995). "Turbulent breaking of overturning gravity waves below a critical level". en. In: *Applied Scientific Research* 54.3, pp. 163–176. ISSN: 0003-6994, 1573-1987. DOI: 10.1007/BF00849114. URL: http://link.springer.com/10.1007/BF00849114 (visited on 01/30/2020).

Dutay, J.-C et al. (2002). "Evaluation of ocean model ventilation with CFC-11: comparison of 13 global ocean models". en. In: *Ocean Modelling* 4.2, pp. 89–120. ISSN: 14635003. DOI: 10.1016/S1463-5003(01)00013-0. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500301000130 (visited on 01/29/2020).

Fringer, O.B., M. Gerritsen, and R.L. Street (2006). "An unstructured-grid, finite-volume, nonhydrostatic, parallel coastal ocean simulator". en. In: *Ocean Modelling* 14.3-4, pp. 139–173. ISSN: 14635003. DOI: 10.1016/j.ocemod.2006.03.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500306000394 (visited on 07/01/2020).

Galmiche, Martin, Olivier Thual, and Philippe Bonneton (1997). "Direct Numerical Simulation of Turbulence in a Stably Stratified Fluid and Wave-Shear Interaction". In: *Applied Scientific Research* 59.2, pp. 111–125. ISSN: 1573-1987. DOI: 10.1023/A:1001171018994. URL: https://doi.org/10.1023/A:1001171018994.

Gargett, A. E., T. R. Osborn, and P. W. Nasmyth (1984). "Local isotropy and the decay of turbulence in a stratified fluid". en. In: *Journal of Fluid Mechanics* 144, p. 231. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112084001592. URL: http://www.journals.cambridge.org/abstract_S0022112084001592 (visited on 04/20/2019).

Garrett, C and W Munk (1979). "Internal Waves in the Ocean". en. In: *Annual Review of Fluid Mechanics* 11, p. 31. DOI: 10.1146/annurev.fl.11.010179.002011.

Gent, Peter R. and James C. McWilliams (1990). "Isopycnal Mixing in Ocean Circulation Models". en. In: *Journal of Physical Oceanography* 20, pp. 150–155. DOI: 10.1175/1520-0485(1990)020<0150:IMIOCM>2.0.CO;2. URL: https://journals.ametsoc.org/doi/abs/10.1175/1520-0485%281990%29020%3C0150%3AIMIOCM%3E2.0.CO%3B2.

Gourlay, Michael J. et al. (2001). "Numerical modeling of initially turbulent wakes with net momentum". en. In: *Physics of Fluids* 13.12, pp. 3783–3802. ISSN: 1070-6631, 1089-7666. DOI: 10.1063/1.1412246. URL: http://aip.scitation.org/doi/10.1063/1.1412246 (visited on 01/14/2020).

Groen, D., S. J. Zasada, and P. V. Coveney (2014). "Survey of Multiscale and Multiphysics Applications and Communities". In: *Computing in Science Engineering* 16.2, pp. 34–43. ISSN: 1558-366X. DOI: 10.1109/MCSE.2013.47.

Hassid, Samuel (1980a). "Collapse of turbulent wakes in stably stratified media". en. In: *Journal of Hydronautics* 14.1, pp. 25–32. ISSN: 0022-1716, 1555-5909. DOI: 10.2514/3.48175. URL: https://arc.aiaa.org/doi/10.2514/3.48175 (visited on 01/16/2020).

– (1980b). "Similarity and decay laws of momentumless wakes". en. In: *Physics of Fluids* 23.2, p. 404. ISSN: 00319171. DOI: 10.1063/1.862984. URL: https://aip.scitation.org/doi/10.1063/1.862984 (visited on 01/16/2020).

Hopfinger, E. J. (1987). "Turbulence in stratified fluids: A review". en. In: *Journal of Geophysical Research* 92.C5, p. 5287. ISSN: 0148-0227. DOI: 10.1029/JC092iC05p05287. URL: http://doi.wiley.com/10.1029/JC092iC05p05287 (visited on 06/30/2020).

Issa, R.I (1986). "Solution of the implicitly discretised fluid flow equations by operator-splitting". en. In: *Journal of Computational Physics* 62.1, pp. 40–65. ISSN: 00219991. DOI: 10.1016/0021-9991(86)90099-9. URL: https://linkinghub.elsevier.com/retrieve/pii/0021999186900999 (visited on 01/09/2020).

Jasak, Hrvoje, Aleksandar Jemcov, and Željko Tuković (2007). "OpenFOAM: A C++ Library for Complex Physics Simulations". en. In: IUC, Dubrovnik, Croatia, p. 20.

Javam, A., J. Imberger, and S. W. Armfield (2000a). "Numerical study of internal wave–caustic and internal wave–shear interactions in a stratified fluid". en. In: *Journal of Fluid Mechanics* 415, pp. 89–116. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112000008600. URL: https://www.cambridge.org/core/product/identifier/S0022112000008600/type/journal_article (visited on 06/22/2020).

– (2000b). "Numerical study of internal wave–wave interactions in a stratified fluid". en. In: *Journal of Fluid Mechanics* 415, pp. 65–87. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112000008594. URL: https://www.cambridge.org/core/product/identifier/S0022112000008594/type/journal_article (visited on 07/10/2020).

Klymak, Jody M. and Sonya M. Legg (2010). "A simple mixing scheme for models that resolve breaking internal waves". en. In: *Ocean Modelling* 33.3-4, pp. 224–234. ISSN: 14635003. DOI: 10.1016/j.ocemod.2010.02.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500310000144 (visited on 01/29/2020).

Launder, B E and D B Spalding (1974). "The Numerical Computation of Turbulent Flows". en. In: *Computer Methods in Applied Mechanics and Engineering* 3.2, p. 21. DOI: 10.1016/0045-7825(74)90029-2.

Lin, J T and Y H Pao (1979). "Wakes in Stratified Fluids". In: *Annual Review of Fluid Mechanics* 11.1, pp. 317–338. DOI: 10.1146/annurev.fl.11.010179.001533. eprint: https://doi.org/10.1146/annurev.fl.11.010179.001533. URL: https://doi.org/10.1146/annurev.fl.11.010179.001533.

Marchesiello, Patrick, James C. McWilliams, and Alexander Shchepetkin (2003). "Equilibrium Structure and Dynamics of the California Current System". In: *Journal of Physical Oceanography* 33.4, pp. 753–783. DOI: 10.1175/1520-0485(2003)33<753:ESADOT>2.0.

CO;2. eprint: https://doi.org/10.1175/1520-0485(2003)33<753:ESADOT>2.0.CO;2. URL: https://doi.org/10.1175/1520-0485(2003)33<753:ESADOT>2.0.CO;2.

Marshall, John et al. (1997a). "A finite-volume, incompressible Navier Stokes model for studies of the ocean on parallel computers". en. In: *Journal of Geophysical Research: Oceans* 102.C3, pp. 5753–5766. ISSN: 01480227. DOI: 10.1029/96JC02775. URL: http://doi.wiley.com/10.1029/96JC02775 (visited on 01/29/2020).

Marshall, John et al. (1997b). "Hydrostatic, quasi-hydrostatic, and nonhydrostatic ocean modeling". en. In: *Journal of Geophysical Research: Oceans* 102.C3, pp. 5733–5752. ISSN: 01480227. DOI: 10.1029/96JC02776. URL: http://doi.wiley.com/10.1029/96JC02776 (visited on 02/06/2020).

Martin, Michael A (2016). "Influence of momentum excess on the pattern and dynamics of intermediate-range stratified wakes". en. PhD thesis. Monterey, California: Naval Postgraduate School.

Mellor, George L. and Tetsuji Yamada (1982). "Development of a turbulence closure model for geophysical fluid problems". en. In: *Reviews of Geophysics* 20.4, p. 851. ISSN: 8755-1209. DOI: 10.1029/RG020i004p00851. URL: http://doi.wiley.com/10.1029/RG020i004p00851 (visited on 01/31/2020).

Meunier, Patrice, Peter J. Diamessis, and Geoffrey R. Spedding (2006). "Self-preservation in stratified momentum wakes". en. In: *Physics of Fluids* 18.10, p. 106601. ISSN: 1070-6631, 1089-7666. DOI: 10.1063/1.2361294. URL: http://aip.scitation.org/doi/10.1063/1.2361294 (visited on 01/17/2020).

Meunier, Patrice and Geoffrey R. Spedding (2004). "A loss of memory in stratified momentum wakes". en. In: *Physics of Fluids* 16.2, pp. 298–305. ISSN: 1070-6631, 1089-7666. DOI: 10.1063/1.1630053. URL: http://aip.scitation.org/doi/10.1063/1.1630053 (visited on 01/17/2020).

– (2006). "Stratified propelled wakes". In: *Journal of Fluid Mechanics* 552, 229–256. DOI: 10.1017/S0022112006008676.

Müller, Peter et al. (1986). "Nonlinear interactions among internal gravity waves". en. In: *Reviews of Geophysics* 24.3, pp. 493–536. ISSN: 87551209. DOI: 10.1029/RG024i003p00493. URL: http://doi.wiley.com/10.1029/RG024i003p00493 (visited on 01/31/2019).

Moody, Zachary E. et al. (2017). "On the structure and dynamics of stratified wakes generated by submerged propagating objects". en. In: *Journal of Operational Oceanography* 10.2, pp. 191–204. ISSN: 1755-876X, 1755-8778. DOI: 10.1080/1755876X.2017.1307801. URL: https://www.tandfonline.com/doi/full/10.1080/1755876X.2017.1307801 (visited on 02/03/2020).

Naudascher, Eduard (1965). "Flow in the wake of self-propelled bodies and related sources of turbulence". In: *Journal of Fluid Mechanics* 22.4, 625–656. DOI: 10.1017/S0022112065001039.

Oliveira, Paulo J. and Raad I. Issa (2001). "An Improved PISO Algorithm for the Computation of Buoyancy-Driven Flows". en. In: *Numerical Heat Transfer, Part B: Fundamentals* 40.6, pp. 473–493. ISSN: 1040-7790, 1521-0626. DOI: 10.1080/104077901753306601. URL: https://www.tandfonline.com/doi/full/10.1080/104077901753306601 (visited on 01/09/2020).

OpenCFD Limited, ed. *k-epsilon*. URL: https://www.openfoam.com/documentation/guides/latest/doc/guide-turbulence-ras-k-epsilon.html.

– ed. *Programmer's Guide*. URL: sourceforge.net/projects/openfoam/files/v2006/ProgrammersGuide.pdf/download.

Rodi, Wolfgang (1987). "Examples of calculation methods for flow and mixing in stratified fluids". en. In: *Journal of Geophysical Research* 92.C5, p. 5305. ISSN: 0148-0227. DOI: 10.1029/JC092iC05p05305. URL: http://doi.wiley.com/10.1029/JC092iC05p05305 (visited on 01/07/2020).

Sarkar, S. and A. Scotti (2017). "From Topographic Internal Gravity Waves to Turbulence". en. In: *Annual Review of Fluid Mechanics* 49.1, pp. 195–220. ISSN: 0066-4189, 1545-4479. DOI: 10.1146/annurev-fluid-010816-060013. URL: http://www.annualreviews.org/doi/10.1146/annurev-fluid-010816-060013 (visited on 01/28/2020).

Schetz, J. and A. Jakubowski (1975). "Experimental studies of the turbulent wake behind self-propelled slender bodies". en. In: *13th Aerospace Sciences Meeting*. Pasadena,CA,U.S.A.: American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.1975-117. URL: http://arc.aiaa.org/doi/10.2514/6.1975-117 (visited on 01/22/2020).

Shchepetkin, Alexander F. and James C. McWilliams (2003). "A method for computing horizontal pressure-gradient force in an oceanic model with a nonaligned vertical coordinate". en. In: *Journal of Geophysical Research* 108.C3, p. 3090. ISSN: 0148-0227. DOI: 10.1029/2001JC001047. URL: http://doi.wiley.com/10.1029/2001JC001047 (visited on 02/06/2020).

– (2005). "The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model". en. In: *Ocean Modelling* 9.4, pp. 347–404. ISSN: 14635003. DOI: 10.1016/j.ocemod.2004.08.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S1463500304000484 (visited on 01/29/2020).

Slinn, Donald N. and James J. Riley (1996). "Turbulent mixing in the oceanic boundary layer caused by internal wave reflection from sloping terrain". en. In: *Dynamics of Atmospheres and Oceans* 24.1-4, pp. 51–62. ISSN: 03770265. DOI: 10.1016/0377-0265(95)00425-4. URL: https://linkinghub.elsevier.com/retrieve/pii/0377026595004254 (visited on 02/05/2020).

Spedding, G. R. (1997). "The evolution of initially turbulent bluff-body wakes at high internal Froude number". In: *Journal of Fluid Mechanics* 337, 283–301. DOI: 10.1017/S0022112096004557.

Spedding, G. R., F. K. Browand, and A. M. Fincham (1996). "Turbulence, similarity scaling and vortex geometry in the wake of a towed sphere in a stably stratified fluid". In: *Journal of Fluid Mechanics* 314, 53–103. DOI: 10.1017/S0022112096000237.

Spedding, Geoffrey R. (2014). "Wake Signature Detection". In: *Annual Review of Fluid Mechanics* 46.1, pp. 273–302. DOI: 10.1146/annurev-fluid-011212-140747. eprint: https://doi.org/10.1146/annurev-fluid-011212-140747. URL: https://doi.org/10.1146/annurev-fluid-011212-140747.

Staquet, C. and J. Sommeria (2002). "Internal Gravity Waves: From Instabilities to Turbulence". en. In: *Annual Review of Fluid Mechanics* 34.1, pp. 559–593. ISSN: 0066-4189,

1545-4479. DOI: 10.1146/annurev.fluid.34.090601.130953. URL: http://www.annualreviews.org/doi/10.1146/annurev.fluid.34.090601.130953 (visited on 01/28/2020).

Stillinger, D. C., K. N. Helland, and C. W. Van Atta (1983). "Experiments on the transition of homogeneous turbulence to internal waves in a stratified fluid". en. In: *Journal of Fluid Mechanics* 131.-1, p. 91. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/S0022112083001251. URL: http://www.journals.cambridge.org/abstract_S0022112083001251 (visited on 04/17/2019).

Sumner, Jonathon, Christophe Sibuet Watters, and Christian Masson (2010). "CFD in Wind Energy: The Virtual, Multiscale Wind Tunnel". en. In: *Energies* 3.5, pp. 989–1013. ISSN: 1996-1073. DOI: 10.3390/en3050989. URL: http://www.mdpi.com/1996-1073/3/5/989 (visited on 01/10/2020).

Tang, Youhua et al. (2004). "Multiscale simulations of tropospheric chemistry in the eastern Pacific and on the U.S. West Coast during spring 2002: MULTISCALE SIMULATIONS OF TROPOSPHERIC CHEMISTRY". en. In: *Journal of Geophysical Research: Atmospheres* 109.D23. ISSN: 01480227. DOI: 10.1029/2004JD004513. URL: http://doi.wiley.com/10.1029/2004JD004513 (visited on 02/07/2020).

Tennekes, H and J. L. Lumley (1972). *A First Course in Turbulence*. Cambridge, Massachusetts: The MIT Press.

Vellinga, Michael and Richard A Wood (2002). "Global Climatic Impacts of a Collapse of the Atlantic Thermohaline Circulation". en. In: *Climate Change* 54, p. 17. DOI: 10.1023/A:1016168827653.

Wiersema, David J., Katherine A. Lundquist, and Fotini Katopodes Chow (2020). "Mesoscale to Microscale Simulations over Complex Terrain with the Immersed Boundary Method in the Weather Research and Forecasting Model". In: *Monthly Weather Review* 148.2, pp. 577–595. DOI: 10.1175/MWR-D-19-0071.1. eprint: https://doi.org/10.1175/MWR-D-19-0071.1. URL: https://doi.org/10.1175/MWR-D-19-0071.1.

Woods, J. D. (1969). "On Richardson's Number as a Criterion for Laminar-Turbulent-Laminar Transition in the Ocean and Atmosphere". en. In: *Radio Science* 4.12, pp. 1289–1298. ISSN: 00486604. DOI: 10.1029/RS004i012p01289. URL: http://doi.wiley.com/10.1029/RS004i012p01289 (visited on 04/20/2019).

Zhou, Qi and Peter J. Diamessis (2013). "Reflection of an internal gravity wave beam off a horizontal free-slip surface". en. In: *Physics of Fluids* 25.3, p. 036601. ISSN: 1070-6631, 1089-7666. DOI: 10.1063/1.4795407. URL: http://aip.scitation.org/doi/10.1063/1.4795407 (visited on 01/30/2020).