Spring 2021

Mimetic Coastal Ocean Modeling In General Coordinates And Using Machine Learning Based Predictions

Manuel Alonzo Valera
Claremont Graduate University

MIMETIC COASTAL OCEAN MODELING IN GENERAL COORDINATES AND USING MACHINE LEARNING BASED PREDICTIONS

by

Manuel Alonzo Valera

Claremont Graduate University and San Diego State University

2021

APPROVAL OF THE DISSERTATION COMMITTEE

This dissertation has been duly read, reviewed, and critiqued by the Committee listed below, which hereby approves the manuscript of Manuel Alonzo Valera as fulfilling the scope and quality requirements for meriting the degree of Doctor of Philosophy in Computational Science.

Jose E. Castillo, Chair
Computational Science Research Center
San Diego State University

Chris Paolini
Department of Electrical and Computer Engineering
San Diego State University

Peter Blomgren
Department of Mathematics and Statistics
San Diego State University

Ali Nadim
Department of Mathematics
Claremont Graduate University

Allon G. Percus
Department of Mathematics
Claremont Graduate University

ABSTRACT OF THE THESIS

Mimetic Coastal Ocean Modeling in General Coordinates and using Machine Learning based predictions.

by Manuel Alonzo Valera
Claremont Graduate University and San Diego State University: 2021

Abstract

Nonlinear internal waves are a ubiquitous and fundamental aspect of the coastal ecosystem understanding. However, they rely on extreme geographical conditions and precise dimensional equilibrium to be captured accurately. The General Curvilinear Coastal Ocean Model (GCCOM) was validated, serial and parallel versions for a set of experiments showcasing stratified and non-hydrostatic flow phenomena. Still, the 3D curvilinear capability has proven to be elusive. We apply cutting-edge numerical methods to improve upon the previously validated GCCOM, elevating it to field-scale capacity. This reformulation of the GCCOM equations uses novel 3D curvilinear mimetic operators, a buoyancy body force, and mimetic upwind and gradient-based momentum equations developed for this work. This model represents the most complete implementation of the 3D curvilinear mimetic operators utilizing the MOLE library or any other mimetic applications in literature to date. Results show it to be more physically accurate and better energy conserving than the validated GCCOM and other similar models, permitting the use of 3D curvilinear grids for arbitrary geometries, parallelizable arbitrary domain decomposition, and order-of-magnitude wider time steps. Additionally, we implement machine learning models to coastal ocean data to predict Dissolved Oxygen (DO) content with supervised methods; results show a Median Absolute Percentage Error (MAPE) of 2-6% for instantaneous indirect readings of DO and 0.18% for five days forecast of DO in coastal areas, using a previously predicted temperature of 1.60% MAPE. Dissolved Oxygen is known to be a critically important component to track in coastal environments but also expensive to measure and almost impossible to model with traditional methods due to high nonlinearity. The ML component of this thesis opens the possibility of high precision indirect estimates of biogeochemical quantities, along with highly accurate time series forecasts and a host of new applications of machine learning to environmental sciences.

DEDICATION

Dedicated to my Father and Grandfather, who died while I was abroad, doing the Ph.D.
To my Mother who I haven't seen in five years, and I miss dearly.
To my wife who has been my support and my reason to endure.

ACKNOWLEDGMENTS

First and foremost I want to thank Dr. Jose Castillo, my advisor, for giving me the opportunity of joining the Ph.D. program and prove that a student with less than stellar grades in college is still able to complete an advanced degree and excel while doing it. Also an important and dear thanks to the committee for their mentorship, especially Dr. Allon Percus for recognizing my full potential.

To everyone who have helped me grow professionally, like Drs. Angie Garcia, Mary Thomas, Barbara Bailey, Ryan Walter, Tom Connolly, Richard Loft, AJ Lauer and Peter Lauritzen and everyone else who have given me the opportunity to be part of something bigger. Thanks for sharing your insight and feedback whenever possible, I have grown and matured many times over what I had on my previous life in Venezuela.

An even more special honor to my friends, who have contributed to my mental health in good times and bad alike, to Johnny Corbino, Jared Brzenski, Stephanie Lauber, Adrian Cantu, Angel Boada, Matt Bantelman and everyone else who I have shared a conversation, a celebration or a rant with. Some of you have lifted my spirit when I needed it the most and I will be forever grateful for that.

Last but never least, thanks to my family for believing in me although they couldn't understand how or why I did it, I hope you can forgive my absence.

Thanks Mom, thanks Dad. I love You.

Per ardua ad astra

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Science is a conversation with nature. This dialogue can happen differently; since our early origins, we have interpreted and understood what nature says and used it to our benefit. On the other hand, theoretical science is a monologue that needs the response of an experiment to become confirmed science, and experiments need to be formalized as a theory if the knowledge created is to be replicated and conserved. Science advanced under this dichotomy until a few decades ago when the experiment and the theory met a branch new way of communicating with nature, using the power of computers, which simulate reality and bring to life algorithms and differential equations.

For the first time in history, we could use the third cornerstone of science that has been hidden from us until then. This computational science can work as a bridge between theory and experiment. Still, it can also partially replace one of the two other facets of science, and at the same time is its realm of research. This miracle is a multi-generational happening that our kind has the responsibility and the exciting opportunity to develop and apply for the first time in all sorts of fields of knowledge.

Some aspects of computational science are more natural to some applications. For example, engineering and physics problems can apply computational fluid dynamics, and machine learning has been developed and applied in statistical analysis for decades. The application of different aspects of computational science is a continually exciting and evolving exercise, and a naturally interdisciplinary activity, offering an almost infinite combination of possible applications and cross-cultural exchange.

The focus of this work is the development of models for coastal ocean dynamics, specifically computational fluid dynamics models capable of capturing several scales and physics phenomena while generating accurate results in a timely fashion. By the end of this dissertation, we will also introduce several machine learning models that can also

be used in coastal ocean science to monitor and predict critical quantities for the environment and ecological equilibrium.

The oceanographic community has developed and validated several global and regional models capable of interacting and encompassing scales from kilometers to hundreds of kilometers. These models can also track and forecast global currents, ice sheet covering, sea surface temperature, etc. Still, the lower scales remain lacking a sufficiently good model that can capture the thermodynamics and hydrodynamics of the sub-kilometer to sub-meter scales, along with the coastal ocean interface that introduces highly nonlinear behaviors and so far intractable problems.

The first requirement of a coastal ocean model is to be non-hydrostatic. Hydrostaticity is an assumption that works in the open ocean. It implies that the vertical velocities are too small compared to the horizontals, for which the Navier-Stokes (N-S) equations, the fluid dynamics equations, are significantly simplified. Hydrostaticity works in the open ocean because the horizontal scales are many times larger than the ocean depth. Still, as we move near the coast, these scales start becoming comparable, and in many cases, the water column depth would be equal or even more significant than the horizontal scales modeled. Thus, the Navier-Stokes equations must be non-hydrostatic in coastal ocean models.

A second requirement for coastal ocean models is to capture multiple physical systems at the same time. Stratification is one main feature of oceanic waters and becomes especially important in the coastal ocean. Stratification is the natural ordering of water density in layers. Practically all ocean water is stratified, with each layer receives a denomination and even specific ecosystems within it. On the coastal region, the number of layers diminishes with lower depths but depending on the slope of the shelf in the particular area, ocean currents and tidal forces can induce mixing and shoaling of the contents of the lower layers towards the higher layers. The content of each layer is density, salinity, and temperature specific. Still, it also has a different biogeochemical composition that functions as nutrients for the various ecosystems they can reach. For example, in a gentle slope coast, cold, nutrient-rich, and oxygen-deprived

waters from the bottom layers will shoal into the uppermost layers with a certain regularity. This process will provide the necessary nutrients for the higher ecosystems to survive. A similar process occurs in steep slope regions, but in this case, the layer shoaling becomes more chaotic, creating bores and turbulence that trigger cascading energy events. These events feed energy from the broad ocean currents into the sub-meter and millimeter-scale, providing the power for crucial metabolic processes. Stratified seawater is a core feature of oceanic and coastal environments. The natural movement created by tidal forces and oceanic currents will transport waves inside the bulk of the water mass on what is called internal waves (IWs) [35]. In stratified media, this means energy transport between density layers when the IWs start interfering with each as a product of changing bathymetry, at the point where they become nonlinear internal waves (NLIWs) and trigger turbulent mixing events across scales. These mixing events are responsible for feeding oceanic-current (kilometers) scale energy into the millimeter and sub-millimeter scale, enabling the production of food for the whole chain, and is a crucial phenomenon to understand and model [97].

Finally, coastal ocean models require energy conservation and handling rough geometries very well, since dissipation effects can take over the mixing and turbulence typical of these environments [90]. In addition, the bathymetry increased abruptly and in exotic ways in many coastal regions, especially underwater canyons. These two specific requirements, along with the novelty of the application, are the primary motivation to introduce mimetic operators in the development of a new version of the General Curvilinear Coastal Ocean Model (GCCOM) [82].

Curvilinear or general coordinates will -partially or fully- adapt the grid cells to the bathymetry shape [47], and different degrees of generality or non-traditional grid geometries are widely used in oceanographic and atmospheric models. For example, the curvilinear grid can be done partially. In this case, the vertical coordinate remains rectilinear while the horizontal grid lines are curvilinear. This spawns a family of grid geometries called σ (sigma) grids. The fully general (curvilinear) coordinate includes the curvilinear quantities of the vertical coordinate as well and, while is more expensive

to solve, is also more accurate in rough or steep submarine terrain, since it is guaranteed, first, that the boundaries of the problem perfectly adapt to the actual shape of the bathymetry, without the need of shaving cells, and secondly, since the whole issue is translated using a general coordinates transformation, to a computational field that is equivalent to a unit cube. This process also simplifies the application of boundary conditions. The coastal ocean modeling community has developed several models, each designed for a specific use, most recently including nonhydrostatic solvers, and usually treating the grid with some degree of boundary following coordinates. σ-grid models remain the most popular choice as of now. Still, the steep bathymetry of submarine canyons and the increasing need for High-Performance, scalable models with higher resolution make the need for a fully-3D curvilinear model apparent. In the coastal ocean dynamics research group of San Diego State University, of which I form part, the need for a new model has been attacked from the ground up, giving birth to the GCCOM.

The GCCOM is a non-hydrostatic, 3D curvilinear, Navier-Stokes model with turbulence capabilities. It has been validated for stratified flows and internal waves modeling [40], and it is the product of several generations of researchers, each of which has expanded and improved into its capabilities. However, as we will later discuss, there is a good case for overhauling its core functioning parts. Mimetic operators are a natural choice for us to explore since this is the expertise of the head of the research group Dr. Jose Castillo. Mimetic operators are discrete analogs of the first order invariant operators' divergence, gradient, curl and Laplacian from vector calculus. For this thesis's purposes, we will operate as tool-users of the MOLE library [31], knowing that they are capable of generating the necessary operators and that these have been validated and tested for the 2D curvilinear and 3D form. The development of 3D curvilinear mimetic operators is a novel concept that escapes the scope of this thesis. Regardless, these new operators have also been used in the development of the model described here, and additionally at some point new implementations of the MOLE library operators had to be developed for the kind of problems here described, dipping our toes into tool-making territory as well.

This thesis's secondary but equally impactful objective is to explore the application of machine learning models to coastal ocean dynamics. Several different models have been applied to regression problems to obtain indirect dissolved oxygen estimations in the coastal waters in the last year. Dissolved oxygen (DO) in seawater is known to be linearly dependent on temperature, salinity, and depth in the open ocean. Still, this clear dependence is lost when we move towards the coastal region, which becomes highly nonlinear. Similarly, more fundamental variables like temperature become not predictable by traditional methods in the coastal regime. Therefore, the DO estimation problem can be attacked as a nonlinear regression model. Previous purely nonlinear regression models had limited success in estimating DO in coastal waters [95, 76], while more advanced machine learning approaches have been scarce, only adequate in accuracy, and long in time scales for forecasting [71, 99, 51, 98, 89], limiting their applicability. Furthermore, a DO forecast would be a beneficial tool for coastal fisheries, ecological managers, and similar activities, but the task's complexity has not yielded any advances in this matter. In the second part of this thesis, we describe the application of supervised machine learning techniques to obtain an almost exact reading of DO in coastal waters, which was published in the last year in [87], as well as a novel implementation of time series forecast using Long Short-Term Memory neural networks for five days forecasts of DO and temperature in a coastal site.

1.1 Motivation

The first main objective for a new GCCOM formulation is to be able to reproduce internal waves accurately. Internal waves are the primary process driving the energy from ocean currents into more minor scales, where krill and plankton grow, becoming fundamental phenomena in the ocean food chain [97]. Internal waves (IW) occur when a stratified body of water is perturbed, which is usually generated by ocean currents and tidal forces. Just like surface waves breaking on a beach, the large slopes of the continental shelf and bathymetric features cause internal waves to break, giving rise to turbulence and the formation of underwater (internal) bores [92]. These features enhance mixing and affect small scales at which microscopic marine life lives. These

underwater features also deliver nutrients to coastal ecosystems like kelp forests and coral reefs, maintaining productive nearshore waters. The rising of global and oceanic temperatures will most likely affect the vertical stratification of coastal areas [68]. However, the implications of these changes in the coastal ecosystems and economic activities are still unknown.

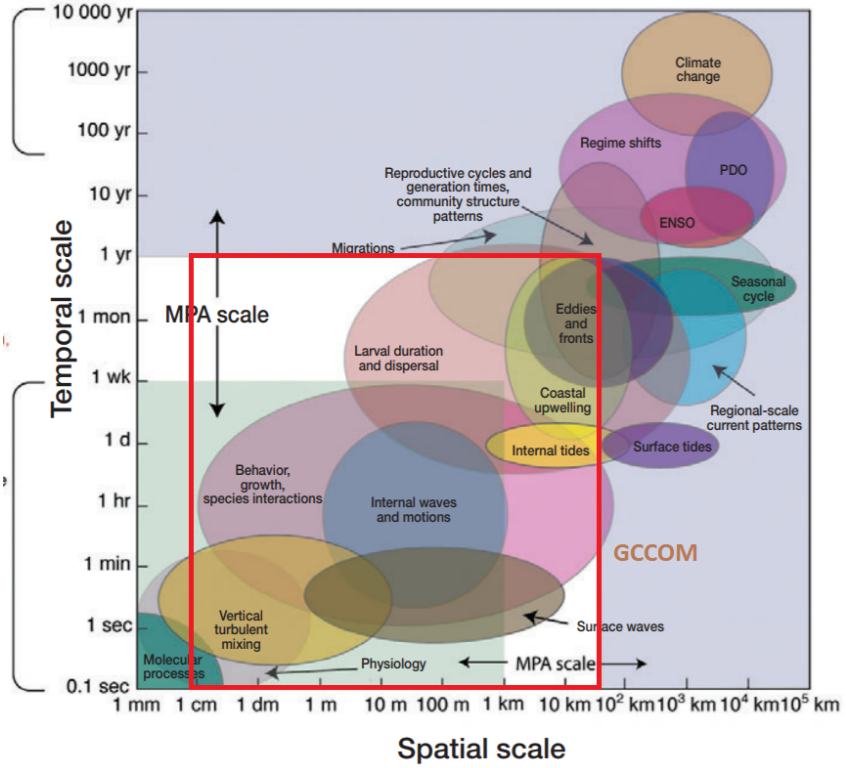


Figure 1.1. Scales and biogeochemical-physical phenomena present in Coastal Ocean Dynamics with GCCOM-related scales marked in red. From [22].

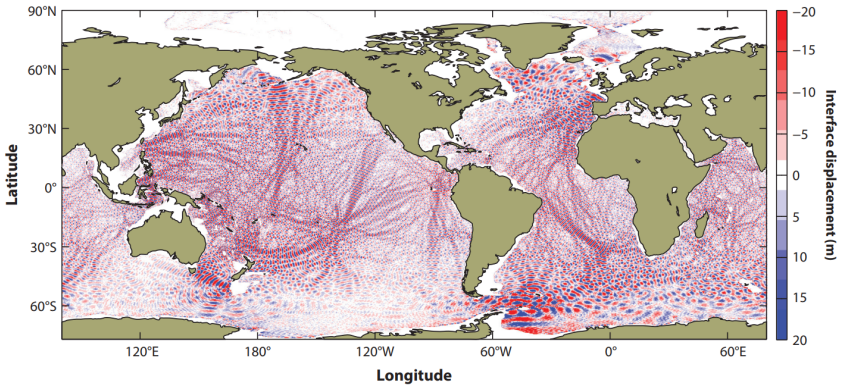


Figure 1.2. Map of internal wave beams propagating in the world's oceans. From [97].

Internal waves travel undisturbed in the open ocean, using the stratified ocean as waveguides to propagate with minimal dissipation, in a soliton-like state (see Figure 1.2), until turbulence, mixing, and bore creation overcome this behavior. This happens especially at the coastal ocean and is evident in underwater canyons [92, 93, 38, 91]. The Internal Waves/Coastal Shelf interaction is often understudied, in part because coastal ocean high-resolution models are scarce, and even fewer are models that can adapt to the coastal bathymetry well enough to capture the physics realistically, such as a 3D curvilinear model. Coastal models capable of handling IW also need to be non-hydrostatic, render turbulence and be able to perform and scale in HPC architectures to model all of the (multi-scale, multi-physics) processes present in this phenomenon, from the ocean current in the kilometers wavelength to the sub-meter scale of turbulent mixing, as illustrated in Figure 1.1, and in the other sense, from fluid dynamics to convection-diffusion, equation of state, and even biogeochemical models, that as we will see may be able to be replaced by Machine Learning models.

Several models with partial fulfilling of this list were produced and used in the last three decades [75, 25, 61, 67, 79, 39, 72, 9, 64], including the General Curvilinear Coastal Ocean Model (GCCOM) developed at San Diego State University (SDSU) that can handle 3D geometries [40]. GCCOM was recently validated, and part of this project's scope is to use it in a specific real-world scenario, aiming to emulate the real-world physics of internal wave formation and mixing in the Monterey Bay and La Jolla underwater canyons. This included realistic density and temperature profiles along and across the bay.

On the other hand, the temperature has proved a good enough parameter to estimate nutrients in surface water, and water column [53]. Furthermore, exploratory work has demonstrated the potential to use Machine Learning techniques as regression models to infer nutrient contents of seawater from temperature-salinity data at any depth [37, 42, 45, 88]. The second part of this thesis describes supervised machine learning methods employed to estimate highly nonlinear quantities in coastal ocean environments, specifically dissolved oxygen, using simple estimators or time series data

for forecasting. These machine learning models will be able to take the function of a traditional biogeochemical module in a coastal ocean model, used instead as a post-processing tool, and be capable of forecasting conditions of these markers in time with the help of AI workflows here proposed.

1.2 Outline

This thesis is organized as follows, Chapter 2 includes the description of the GCCOM model development history and latest advancements, as well as the validation experiments we will replicate later in the thesis and some alternative approaches recommended for future studies. Chapters 3 and 4 describe the history and development of the mimetic operators and the ideas and design of the Mimetic GCCOM model here presented, respectively, followed by the first batch of results in Chapter 4.2 for the Mimetic GCCOM. The second part of this thesis is comprised of Chapters 5 and 5.3 where their machine learning models and results are described. Finally, conclusions and further work are presented in Chapter 6

CHAPTER 2

THE GENERAL CURVILINEAR COASTAL OCEAN MODEL

The General Curvilinear Coastal Ocean Dynamics model (GCCOM) [82] is a nonhydrostatic, 3D-curvilinear capable model. It is also the product of many generations of coastal ocean dynamics graduate students. The first version of GCCOM was briefly introduced at [82] as an N-S solver for curvilinear grids in oceanographic applications. Later, a paper formalizing this framework was published, including a seamount test forced with a constant velocity flow [82]. This early was capable of handling stratification and Coriolis forces in general curvilinear coordinates and became the basis for future model iterations. Later, Abouali [3] refined the numerical methods to include a forward-backward advection scheme by Kawamura [54] and three-staged Runge-Kutta time integration. During this time, Abouali also created a 2D curvilinear version of the model utilizing mimetic operators and later proved the 3D curvilinear mimetic operators to be feasible to solve Poisson equation problems with high-order accuracy cementing the basis for this work [2, 5]. At this moment in time, GCCOM was being run in serial, for which Thomas [80] created a parallel framework for the model to run on. At the same time, Garcia [40, 41] worked along with Ryan Walter and Paul Choboter in improving the models' capabilities significantly, including external forcing compatible with ROMS output for model coupling, an implicit Poisson equation solver, and designing and implementing several benchmark tests that went on to validate the model for continuously stratified flows in 3D curvilinear coordinates. Shortly after validation, a complete re-implementation and re-validation of the model was carried out using a PETSc framework for parallelization [86, 84, 87]. The model showed the same physical behavior as before, up to machine precision, and was tested to scale up to 300 processors across 12 nodes in a six million points grid for a seamount test case. This

promising advancement was hindered by the tiny time steps the model needs to run with the physically correct behavior. The reasons for this time step requirement remain a problem to be solved, and the Fortran, fully validated and parallelized version of the model has realized its potential in its current form [82, 40, 87, 21].

In the rest of this chapter, we will describe how the GCCOM is formulated, how its governing equations work, and outline the motivation to improve in a new iteration of the model, built from the ground up with cutting edge methods and retaining as much of the previous version as possible but without inheriting the shortcomings.

2.1 Governing Equations

GCCOM solves the full Navier-Stokes, seawater density equation of state and temperature/salinity convection-diffusion equations, written in vector notation as follows:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\frac{1}{\rho_0} \nabla p - \frac{g(\rho - \rho_0)}{\rho_0} \hat{k} - \nabla \cdot \vec{\vec{\tau}}, \tag{2.1}$$

$$\frac{\partial T}{\partial t} + \vec{u} \cdot \nabla T = \nabla \cdot (k_T \nabla T), \tag{2.2}$$

$$\frac{\partial S}{\partial t} + \vec{u} \cdot \nabla S = \nabla \cdot (k_S \nabla S), \tag{2.3}$$

$$\nabla \cdot \vec{u} = 0, \tag{2.4}$$

$$\rho = \rho(T, S) = \rho_{ref} + \rho_{dS}(S - S_{ref}) + \rho_{dT}(T - T_{ref}) \tag{2.5}$$

With specific equation of state parameters
$\rho_{ref} = 1027, S_{ref} = 35.0, T_{ref} = 10.0, \rho_{dS} = 0.781$ and $\rho_{dT} = -0.1708$. The equations are nondimensionalized using the following variables, with problem specific non-dimensional factors:

$$
\begin{aligned}
x_i^* &= x_i/L^* & p^* &= p/(\rho_0(U^*)^2) \\
u_i^* &= u_i/U^* & T^* &= T/T_0 \\
t^* &= t(U^*/L^*) & S^* &= S/S_0 \\
\rho^* &= \rho/\rho_0 & g^* &= L^*/(U^*)^2
\end{aligned}
\tag{2.6}
$$

GCCOM uses the Boussinesq approximation [18] to ponder a nonphysical analog to pressure, and this approximation takes advantage of the continuity equation 2.4, which when applied to the momentum equation 2.1 yields a quantity dimensionally equivalent to pressure. This pseudo-pressure is then derivated to obtain a correction term for the updating of the velocity fields. This is called a predictor-corrector method for velocities. The solution of the pseudo-pressure is represented as a linear system of a 27-point stencil, with curvilinear coefficients associated as shown by Torres [83], this step is solved using an iterative solver provided by the PETSc library, usually general conjugate gradient. However, the option to use a wide array of solvers is present. The correction step is admittedly much more straightforward, using an Euler time scheme to correct the velocities.

2.1.1 Body Force

Typically, a body force is a term that defines the action of external forces in the system; in the case of N-S, the typical body force is a buoyancy term. Buoyancy can be defined in several ways, typically re-scaling the density field time the gravity force. In our case, Buoyancy, b is defined as:

$$
b = -g\frac{\rho - \rho_0}{\rho_0} \tag{2.7}
$$

This is the second-to-last term we see in 2.1 and is present in the $\hat{k}$ direction only. This specific buoyancy definition considers splitting the density between background stratification and deviation from the mean, being this latter component the active body force in the problem. This assumption holds only for *small* density variations.

2.1.2 Large Eddy Simulation

GCCOM also takes advantage of Large Eddy Simulations (LES) to handle turbulence inside the model; LES was first proposed by [78] as a way to reduce computational demands of Direct Numerical Simulations. LES functions as a low-pass filter of the minor scales in the simulation, called Smagorinsky scale C_s, and it requires tuning of this scale for each application, but typical values range from 0.8 to 0.22-0.24. Furthermore, the Length scale ℓ is proportional to the discretization and defined as $\ell = (\Delta x \Delta y \Delta z)^{1/3}$ with these constants we define a turbulent eddy viscosity ν_T that replaces the bulk density of the problem and is defined as:

$$\nu_T = (C_s \ell)^2 \sqrt{2 \bar{S_{ij}} \bar{S_{ij}}} \tag{2.8}$$

$$\bar{S_{ij}} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}) \tag{2.9}$$

Where the term $\bar{S_{ij}}$ is called the strain-tensor. Finally, the τ in the last term of 2.1 is replaced by:

$$\tau_{ij} = -2\nu_T \bar{S_{ij}} \tag{2.10}$$

2.2 Numerical Algorithms

The choice and application of numerical algorithms in any model must be a conscious decision and generally justified by the nature of the modeled equations. For N-S, we deal with nonlinear, hyperbolic equations with a fully elliptic component in the pressure solution (Poisson equation). In this section, we will define the algorithms used by the validated version of GCCOM.

2.2.1 Space-based Derivation

2.2.1.1 Kawamura Advection Scheme

The first critical component of the GCCOM is the use of a forward-backward composite advection scheme especially crafted for terms like $a\frac{du}{dx}$ in general coordinates;

this scheme was first described by [54] and has been implemented into every single version of GCCOM until now, making it one of its trademarks. The Kawamura advection scheme is described by:

$$(f_\xi \frac{\partial u}{\partial \xi})_{i,j,k} = (f_\xi)_{i,j,k} \frac{-u_{i+2,j,k} + 8(u_{i+1,j,k} - u_{i-1,j,k}) + u_{i-2,j,k}}{12\delta\xi} \tag{2.11}$$

$$\times |(f_\xi)_{i,j,k}| \frac{u_{i+2,j,k} + -4u_{i+1,j,k} + 6u_{i,j,k}) - 4u_{i-1,j,k} + u_{i-2,j,k}}{4\delta\xi} \tag{2.12}$$

In general coordinates, the term f_ξ becomes the covariant component of the velocities or $f_\xi = u\xi_x + u\xi_y + u\xi_z$. We see from this formulation that we need the application of metric grids in each iteration of the solver (here only described w.r.t. one direction) to solve for arbitrary geometries. This carries out a significant overhead in calculations but also in memory requirements for the model, and to solve this, it is necessary to convert all the quantities and equations beforehand, carry out the calculations in the transformed equations, and then transform back only when needed. A description of an attempt by the writer to do exactly this is described in 2.5.1.

2.2.2 Time-based Integration

The time integration scheme of GCCOM is the 3-stages Runge Kutta algorithm, which is 2nd order accurate in nonlinear systems as what GCCOM solves. This time scheme is robust and relatively stable. Still, it needs the updating of the right-hand side of the algorithm at each stage, something equivalent to solving the model time update three times fully and then aggregating the solutions. As we will see later, this scheme is replaced in the Mimetic GCCOM by a strong-stability-preserving version of Runge-Kutta which only uses one update of the right-hand-side ($f(u^n, ...)$), without sacrificing stability, representing another important gain in overhead and memory requirements for the model. For illustration purposes, the RK3 algorithm follows:

$$u' = u^n + \frac{\delta t}{3} f(u^n, ...) \tag{2.13}$$

$$u'' = u^n + \frac{\delta t}{2} f(u', ...) \tag{2.14}$$

$$u^{n+1} = u^n + \delta t f(u'', ...) \tag{2.15}$$

2.3 Hydrostatic-Pressure Gradient Force (HPGF) Algorithm

The hydrostatic pressure gradient formulation, which is currently being used on GCCOM [29], is based on the split of the pressure term on the hosted equations into non-hydrostatic and hydrostatic parts, i.e. $p = p_h + p_{nh}$, and recalling the hydrostatic pressure to be $\partial p_h / \partial z = -\rho g$, we get:

$$\frac{\partial p}{\partial z} = -\rho g + \frac{\partial p_{nh}}{\partial z} \tag{2.16}$$

This expression effectively becomes the z-contribution of the pressure gradient and the body force g on the original equations, hence disregarding the need for additional body force terms to account for the action of gravity. However, for reasons that will be discussed in the next section, the hydrostatic pressure gradient is calculated and accounted for in the horizontal directions instead, as part of the horizontal pressure gradient correction, in the form:

$$\frac{\partial p_h}{\partial z} = -\rho g \tag{2.17}$$

$$p_h = \int_z^0 \rho g dz \tag{2.18}$$

$$\frac{\partial p}{\partial \{x, y\}} = \frac{\partial}{\partial \{x, y\}} \int_z^0 \rho g dz + \frac{\partial p_{nh}}{\partial \{x, y\}} \tag{2.19}$$

2.3.1 The Hydrostatic Inconsistency

The motivation to use the HPGF algorithm will become evident in this section. In transformed equations problems, we formulate our experiment in two different

spaces: Logical and physical. The physical space has curvilinear coordinates, while the logical space is a rectilinear unitary cube governed by a bijective relationship to the physical grid. Now let's imagine a cell with the one-to-one, bijective projection of each point to each space. The two coordinate systems that govern them are generally different for each space, and each coordinate can be written as a composition of the opposite coordinate system. Finally, we need to realize that in boundary-fitted coordinates, the curvilinear physical space closely follows the bathymetry on the bottom. Then, when bathymetry is too steep, the curvilinear cell is stretched so that the computational horizontal direction becomes almost parallel to the physical vertical direction; in this case, models' the physically hosted equations need to be carefully applied to account for these effects. In the HPGF algorithm, the vertical contribution of hydrostatic pressure is always expected in the horizontal direction since we calculate the vertical component of the hydrostatic pressure for each horizontal cell instead (equation 2.17), solving the problem of steep bathymetry with the added cost of requiring a vertically-aligned coordinate [74, 20, 60].

2.3.2 Equivalency of Buoyancy and HPGF body forces

If instead of splitting the pressure gradient into two pieces, we just define the pressure split as [72]:

$$p_h = p_{atm} + g \int_z^0 \bar{\rho}(z')dz' \tag{2.20}$$

for $\bar{\rho}(z)$ the initial stratification profile or background stratification, and p_{atm} the pressure at the surface of water, then the momentum equation for the physical hosted equations become:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\frac{1}{\rho_0}\nabla(p_{nh} - p_h) - \frac{g(\rho - \bar{\rho})}{\rho_0}\hat{k} - \nabla \cdot \vec{\vec{\tau}} \tag{2.21}$$

This formulation does not take any grid considerations into account and, in principle, should be calculated along with the fully 3D curvilinear transformed equations

application by using the Jacobian of the transformation if needed. Additionally, we can infer that in our formulation, the atmospheric and nonhydrostatic pressures will account for the bulk of the calculation. The hydrostatic contribution will be minimal, as is expected in models where the vertical and horizontal scales are comparable.

2.3.3 Formulation and Implementation of HPGF

In order to calculate this integral on the curvilinear grid, a piece-wise polynomial spline is adapted from similar models calculations, called here the HPGF algorithm, that translates into:

$$\int_{z_k}^{z_{k+1}} \frac{\partial p_h}{\partial x} dz = \int_{k}^{k+1} \frac{\partial p_h}{\partial x} h_k d\xi \tag{2.22}$$

$$= h_k(f^{(0)}\xi + 1/2! * f^{(1)}\xi^2 + 1/3! f^{(2)}\xi^3 + 1/4! f^{(3)}\xi^4)|_0^1 \tag{2.23}$$

Where ξ is the computational variable of space, h_k is a local variable defined inside the cell, and the $f^{(n)}$ terms are spline coefficients carefully selected. This algorithm is borrowed from ROMS, and other similar σ-grid models and h_k and the calculation of the coefficients assumes the physical vertical coordinate to be aligned, so is not suited for 3D curvilinear grids, and the spline calculation requires the parallel distribution of the code work to be limited to horizontal partitions, which limits the performance.

The HPGF algorithm brings several significant consequences with it. For one, it permits the validation of the GCCOM model by overcoming the hydrostatic inconsistency; this is no small feat considering the model's complexity and essential use case. Unfortunately, the negative consequences end up being too many to keep this algorithm. First, it restraints the models' grid capabilities to sigma grids, which dramatically hinders one of the main features of GCCOM, handling fully 3D curvilinear geometries, which remain coded and capable inside the model, just never used. Additionally, the spline calculation requires the parallel model to allocate the vertical

entirely in one processor, hindering scalability. Finally, the most critical problem of the HPGF in this implementation is that it requires a tiny Δt choice to produce valid results, otherwise introducing artifacts such as intrusions of high-temperature water, or similar effects, which quickly overcome the physics of the problem. This Δt problem has been demonstrated to come from implementing the HPGF algorithm, but attempts to repair it have not been successful.

Still, even with its limitation, there is a case to be made for solving the HPGF algorithm implementation in the validated GCCOM model, since still to this day this remains the most advanced and complete version of GCCOM, especially because of its parallelization and fixing it would enable the simulation of submarine canyons with internal waves right away.

2.4 GCCOM Validation Experiments

To validate the model, we have to prove that GCCOM is capable of handling continuous stratification in the typical ranges of the coastal ocean and nonhydrostatic effects that may arise in the experiment. The following experiments showcase this behavior and prove the correctness of the model's results quantitatively. We introduce here the experiments we intend to replicate later in this thesis. For the complete list of GCCOM validation experiments, refer to the literature [40, 3].

It is somewhat essential to remark that the grid configurations of Internal Wave Beam are all sigma. At the same time, the Lock Exchange is a rectilinear grid, and the Seamount test case is a curvilinear grid as described by [4].

2.4.1 Lock Exchange

The Lock Exchange (LE) experiment consists of a tank where two fluids of different densities are present, one heavy fluid with higher viscosity on the right half of the container and a light (lower viscosity) fluid on the left side. For GCCOM, the dimensions and parameters have been kept consistent with [46, 39, 59] to make the comparison of validation and results straightforward. The tank has length, Lx = 0.8m, and depth Lz = 0.1m, the width of the tank for GCCOM is Ly = 0.1m, and the density

profile changes smoothly from one constant density value to the other at the center of the tank, thanks to the following density function:

$$\rho(x, y, z, t = 0) = \rho_{min} + \frac{\Delta\rho}{2}(1 - erf(\frac{x}{\delta})) \tag{2.24}$$

where $erf(f)$ is the computational error function, $\delta = 0.01m$ is the width of the interface and $\rho_{min} = 1025.9525$ is the density of the light fluid. The density difference between fluids is selected so that the reduced gravity $g' = 0.01m/s^2$ i.e:

$$g' = g\frac{\Delta\rho}{\rho_0} = 0.01m/s^2 \tag{2.25}$$

with ρ_0 a reference density of 1000 kg/m^3 and g the gravitational acceleration of 9.81 m/s^{-2}. The GCCOM resolution matches [59] and [39] with 400 cells in the horizontal and 100 in the vertical, for a $\Delta x = 0.002m$ and $\Delta z = 0.001m$ and the minimum 6 cells in the $y - direction$, while [46] used much finer grids for Direct Numerical Simulation and is regarded as the benchmark for numerical LE simulations. The GCCOM experiment is run for $180s$ total with a dimensional time step of $\Delta t = 0.0005$

The narrow density interface in LE acts as a smooth barrier along which the gravity-induced currents will develop. In the experimental setups of this experiment, the two fluids are separated by a gate that must be removed quickly but not perturbing the interface since the majority of the important behavior occurs in it. Hence, at the starting time, there are no velocities present in the experiment. Still, these quickly develop at the start of the simulation, when the fluids intrude mutually to each other domain in a pair of fronts that travel opposite directions symmetrically. This experiment is chosen for non-hydrostatic flows simulations for mainly two reasons. One is that the fluid front speeds are easily measurable and they propagate with constant velocity after a short transient [46, 65], the other reason is the apparition of Kevin-Helmholtz (KH) instabilities or billows at the fluid interfaces, starting at the center of the domain and propagating outwards towards the tank walls. KH billows

frequently occur in nature where the vertical and horizontal velocities become similar in a density gradient and are purely non-hydrostatic phenomena, as can be appreciated in [39] where the hydrostatic and non-hydrostatic versions of the experiment are shown.

While the KH billows show a qualitative sign of the model working correctly as a non-hydrostatic N-S solver, the quantitative validation of the LE experiment is done by studying its wavefront speed. The fluids' propagation speed along the top and bottom walls is constant after a short transient and is unique to the non-hydrostatic solution as well, and can be expressed in terms of the Froude number (Fr), which we will explain next. Besides the front velocity u_f, another buoyancy-induced velocity ($\tilde{u}_b$) can be defined for the experiment as:

$$\tilde{u}_b = \sqrt{\tilde{g}' \frac{L_z}{2}} \tag{2.26}$$

As it is clear, this buoyancy velocity is unique to the LE dimensions, and the non-dimensional quantity u_f/u_b is defined as the Froude number $Fr = u_f/u_b$. Moreover, we can also see that the two most important factors in the LE experiment design are the density difference $\Delta\rho$ and the half-height of the tank $\tilde{h} = \frac{L_z}{2}$ with whom any-sized LE experiment may be physically validated for nonhydrostatic, stratified flows. For this experiment's configuration, Fr works out to be equal to $1/\sqrt{2} = 0.7071$.

2.4.1.1 GCCOM LE Results

The validated version of GCCOM was able to obtain a median Froude number of 0.7176, with a 1.8% error difference from the theoretical value, vs $Fr = 0.654$ from [39] (7.5% error difference) and 0.675 from [46] (4.5% error difference). Figure 2.1 shows various stages of the simulation, where the KH instabilities are clearly defined and seen developing correctly, growing from the domain center towards the walls.

2.4.2 Seamount

The seamount experiment is here included as described in [4], this means with homogeneous density, and with a bottom bathymetry defined by a Gaussian bump as:

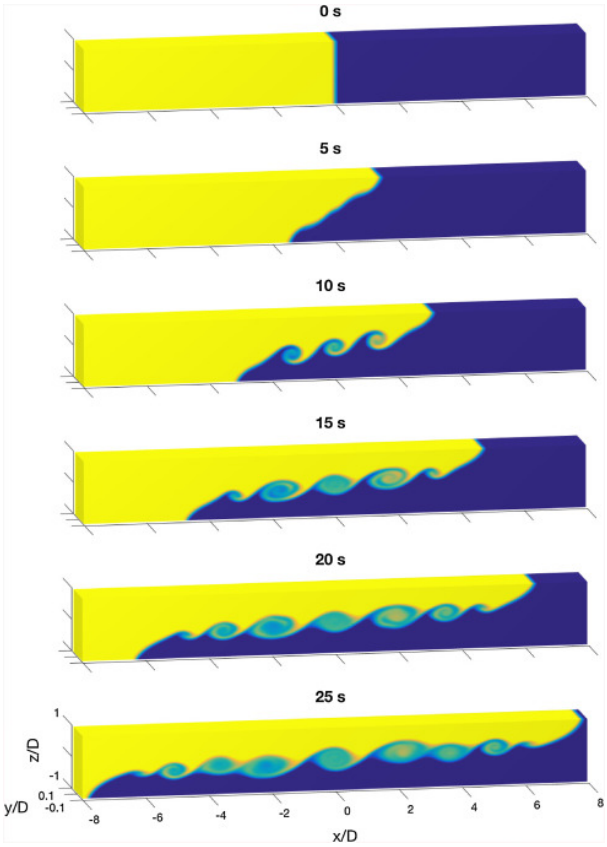


Figure 2.1. GCCOM validation results for LE experiment, featuring the time evolution and the Kelvin-Helmholtz billows formation. From [40].

$$D(x,y) = L^*(-1 + a\exp(-b(x^2+y^2))) \tag{2.27}$$

Where $L^* = 1000m$ is the length scale of the problem, and its maximum depth, $(a,b) = (0.5,8)$ and the domain size is $(x,y) = [-1.8,1.8] \times [-1.4,1.4]$ or $2.8km \times 3.6km$. A grid consisting of $99 \times 65 \times 38$ points was created for this problem, and a sigma version was compared to a curvilinear version. There is not a specific description of the curvilinear grid created for this problem, but a redistribution of the points in the x plane was carried out according to:

$$A = \frac{1}{2\beta}\ln[\frac{1+(e^{\beta}-1)D/L_i}{1+(e^{-\beta}-1)D/L_i}] \tag{2.28}$$

$$x_i = D1 + \frac{\sinh[\beta(\xi_i - A)]}{\sinh\beta A} \tag{2.29}$$

Where L_i is the domain size in the $i-th$ direction, $\beta = 5$ the clustering parameter, and $D = L_i/2$ the location of the clustering. The domain is forced on the left X-Z plane with a linearly increasing velocity, from zero at the bottom to 1 at the top. This experiment shows typical advective refraction behind the seamount, as appreciated in Figure 2.2, and is not meant to be a physically validable result, only qualitatively.

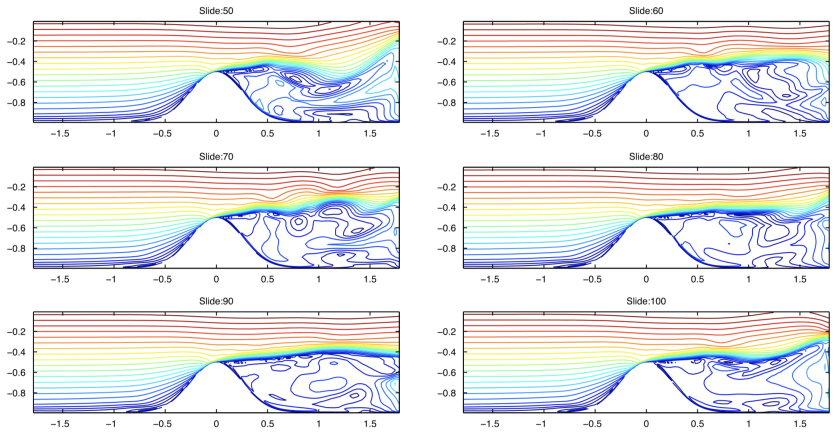


Figure 2.2. GCCOM seamount experiment results for several time frames. Slide 100 is equivalent to t=20000s. From [4].

2.4.3 Internal Waves Beam

The second experiment GCCOM was validated with is the internal wave beam (IWB) experiment [91], also called *tidal flow over a ridge* [27] is field-scale experiment as it has horizontal dimensions of kilometers and depth of $L_z = 1000m$. In this experiment, a small ridge at the bottom of an open ocean is forced at each side with a tidal force of known frequency ω, while the region is stratified with a constant density gradient. The stratification profile has a natural buoyancy frequency, first described by David Brunt and Vilho Väisälä, the Brunt–Väisälä frequency is a fundamental quantity for continuously stratified media, marking the maximum allowed frequency at which the internal wave propagates forward; it is defined by 2.4.3

$$N = \sqrt{-\frac{g}{\rho_0}\frac{\partial \rho}{\partial z}} = 0.007\frac{1}{s} \tag{2.30}$$

It can be shown that, for a continuously stratified media, the buoyant restoring force becomes an analog to a mass-spring system oscillating vertically about z_0, in the form:

$$\frac{d^2\Delta z}{dt^2} = -g' = -\frac{g(\rho(z_0) - \rho(z_0 + \Delta z))}{\rho(z_0)} \tag{2.31}$$

$$= -g(-\frac{d\rho}{dz}\Delta z)\frac{1}{\rho(z_0)} \tag{2.32}$$

from which the natural frequency at 2.4.3 arises.

Knowing the system's natural frequency, we can introduce a tidal wave of known frequency ω such that $\omega = Nsin(\varphi)$ creates an oscillation along a specific line at an angle φ from the vertical. From this predictable behavior, internal waves form beams inside a continuously stratified media with a tidal flow interacting with it.

The experiment is set up with a small ridge at the bottom, which will act as sufficient perturbation to induce the beams to be created. The bathymetry of this gaussian ridge is formulated as:

$$D(x) = L_z - a_b \exp(-\frac{x^2}{2L_b^2}) \tag{2.33}$$

Using $a_b = 20m$ and $L_b = L_x/100m$, with L_x the horizontal length of the domain.

The nonhydrostatic internal wave beam angle, without Coriolis force is given by:

$$\varphi = \tan^{-1}(\sqrt{\frac{(\omega/N)^2}{1 - (\omega/N)^2}}) \tag{2.34}$$

We define L_x as a function of the ω/N parameters so that the beam reflection with the surface is always inside the domain, this is done by choosing L_x to be:

$$L_x = \frac{4L_z}{\tan(\varphi)} \tag{2.35}$$

The parameters in this experiment match [91] and are chosen to keep nonlinear processes, and multiple beams at a minimum [62]. Additionally, boundary reflections are filtered out using sponge layers (SL) at each side following the construction rule:

$$SL = -\frac{u - u_{bc}}{\tau_s} sl(r) \tag{2.36}$$

with r the distance to the boundary, $\tau_s = 100s$ the damping time scale, $u_{bc} = u_0 sin(\omega t)$ the tidal flow forcing at the boundaries, and $u_0 = 0.01m/s$ its amplitude.

The simulation is run for $20T$ tidal periods ($= 2\pi/omega$) because of spin-up concerns on the nonhydrostatic runs (only nonhydrostatic simulations of $\omega/N > 0.8$ require extensive spin-up time). The time step used by GCCOM is $\Delta t = 0.01$ while [91] uses $\Delta t = T/500$

A series of ω/N ratios from 0.2 to 0.8 are run and the angle obtained is measured, then it is compared with the theoretical curves of hydrostatic and non-hydrostatic predicted angles, which differ slightly; we have introduced the nonhydrostatic φ angle before (2.4.3) while the hydrostatic expression for it is $\varphi_h = tan^{-1}(\omega/N)$.

2.4.3.1 GCCOM IWB results

IWB simulations were run for GCCOM at ω/N values of 0.2,0.4,0.6 and 0.8, and once the run finished and the transient has died out, the beam angle is determined by taking a least-square fit of the location of the highest RMS velocity over the last ten tidal periods, in a region in the center of the domain. A sample of the solution can be seen in Figure 2.3.

The quantitative appearance of the beams becomes evident after some time in the GCCOM solution, as they seem to grow from the ridge up until reflecting from the surface. This is a known property of internal waves reflecting off submarine structures in the open ocean as explained in Section 1.1, here appreciated from a side view. Finally, the qualitative analysis of the solutions comes from calculating the angle from

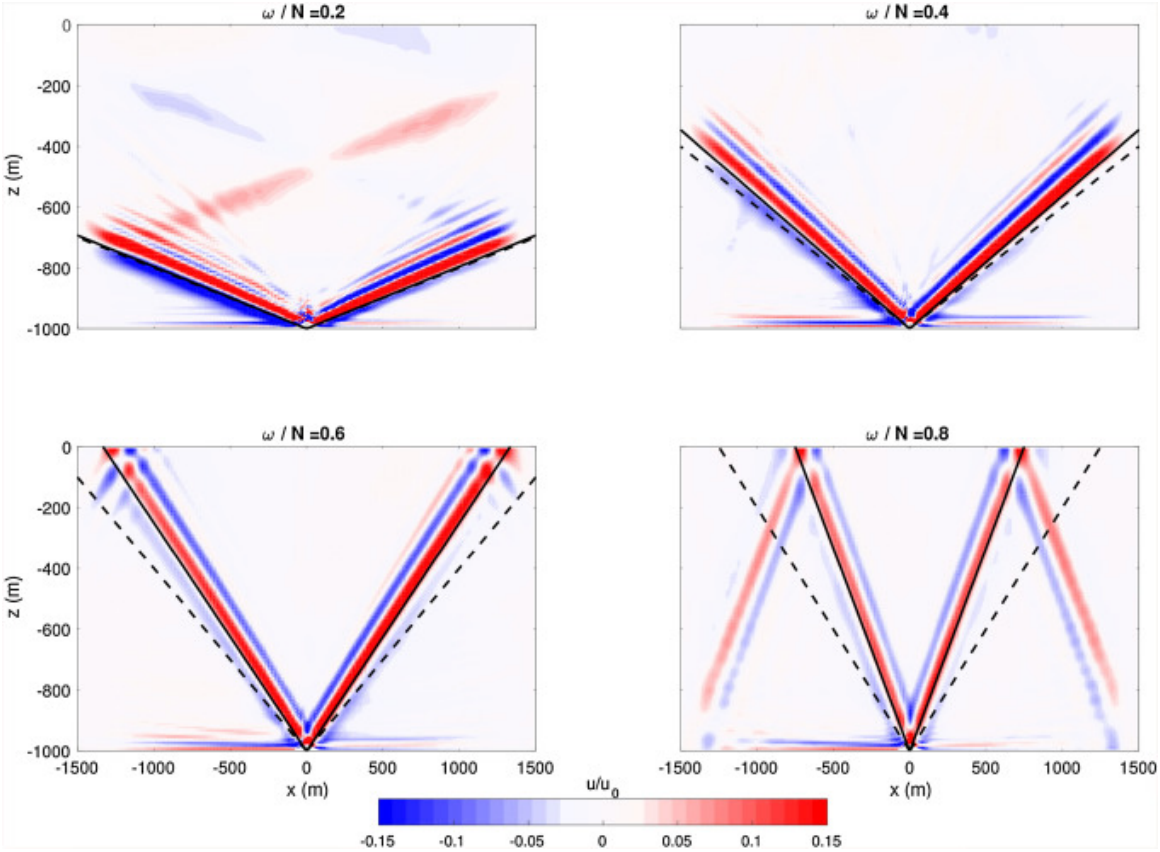


Figure 2.3. GCCOM validation results for the IWB experiment. The beams angle is dependent of the ω/N ratio. From [40].

the vertical, which is specific for hydrostatic or non-hydrostatic calculation. This can be seen in Figure 2.4.

GCCOM attained a good agreement with the theoretical nonhydrostatic angle for the conditions here described, then, compared with the results from [91] the results are also comparable. Altogether, this set of experiments validated the nonhydrostatic and continuous stratification features of the model and elevated it to a similar accuracy level to the rest of the coastal ocean models in the field.

At this point, the necessity to go higher in resolution and broader in field-scale problems was pressuring, so these kinds of problems were attempted for internal waves shoaling and mixing in Monterey Bay and La Jolla underwater canyons. Still, the requirements of small time-step proved to be too restrictive to get timely.

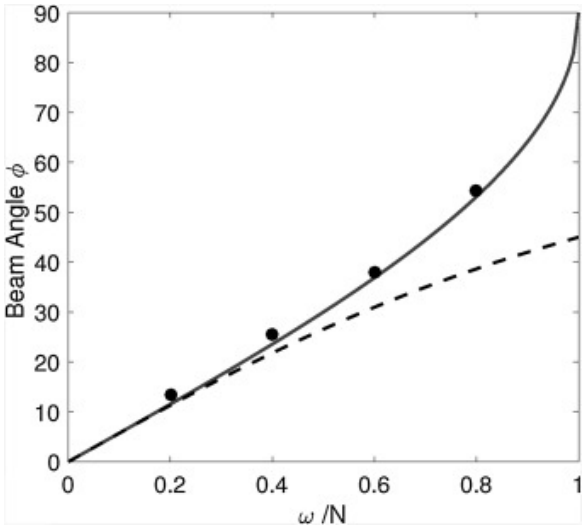


Figure 2.4. GCCOM nonhydrostatic angles obtained for the IWB experiment. From [40].

2.5 Alternative GCCOM Formulations

The writer proposed several improvements and upgrades and partially implemented them during his tenure as a graduate student. The current section describes these alternative approaches and their implementation status. All of the work in this section has been recorded in the GCCOM bitbucket repository in several different branches. The total amount of effort dedicated to these incomplete works amounts to almost two years altogether.

2.5.1 Flux-Vector

Because of the hydrostatic inconsistency causing stiffness in the formulation of the equations, the possibility of an upgrade to the code was proposed. The idea of this upgrade was to finally pose the GCCOM in the computational formulation that permits the complete transformation of all of the involved quantities, the advancement of these quantities in the computational space only, and the conversion back to physical space only when needed to visualize the status of a variable. This form is usually called vectorial form, or flux-vector form of the Navier-Stokes equations, and it can immensely improve the computational performance and memory requirements of the model.

The main idea of the Flux-Vector transformation of the equations is that each equation from the N-S system, including the temperature and salinity equation, can be

rewritten in terms of the derivatives taken, in vectorial variables $Q, E, F, G, E_v, F_v, G_v$ such as:

$$\frac{\partial Q}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = \frac{\partial E_v}{\partial x} + \frac{\partial F_v}{\partial y} + \frac{\partial G_v}{\partial z} \tag{2.37}$$

Where:

$$Q = \begin{bmatrix} p \\ u \\ v \\ w \\ T \\ S \end{bmatrix} \tag{2.38}$$

2

$$E = \begin{bmatrix} u \\ uu + p \\ uv \\ uw \\ uT \\ uS \end{bmatrix} \tag{2.39}$$

$$E_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ k_T q_x \\ k_S p_x \end{bmatrix} \tag{2.40}$$

$$F = \begin{bmatrix} v \\ vu \\ vv + p \\ vw \\ vT \\ vS \end{bmatrix} \tag{2.41}$$

$$F_v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ k_T q_y \\ k_S p_y \end{bmatrix} \tag{2.42}$$

$$G = \begin{bmatrix} w \\ wu \\ wv \\ ww + p \\ wT \\ wS \end{bmatrix} \tag{2.43}$$

$$G_v = \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ k_T q_z \\ k_S p_z \end{bmatrix} \tag{2.44}$$

where $q_i = \frac{\partial T}{\partial x_i}$ and $p_i = \frac{\partial S}{\partial x_i}$. These equations are now only a re-write of the N-S system, but some algebraic manipulation can be made so that the same general structure is preserved, introducing the transformed variables $\bar{Q}, \bar{E}, \bar{F}, \bar{G}, \bar{E}_v, \bar{F}_v, \bar{G}_v$, as we will see next.

2.5.2 Transformed N-S equations in Flux-Vector form

According to [47] (eq. 11-60) the conservative form of the full N-S equations can be written in flux-vector form, in curvilinear coordinates as:

$$\frac{\partial \bar{Q}}{\partial \tau} + \frac{\partial \bar{E}}{\partial \xi} + \frac{\partial \bar{F}}{\partial \eta} + \frac{\partial \bar{G}}{\partial \zeta} = \frac{\partial \bar{E}_v}{\partial \xi} + \frac{\partial \bar{F}_v}{\partial \eta} + \frac{\partial \bar{G}_v}{\partial \zeta} \tag{2.45}$$

Where:

$$\bar{Q} = \frac{Q}{J} \tag{2.46}$$

$$\bar{E} = \frac{1}{J}(\xi_t Q + \xi_x E + \xi_y F + \xi_z G) \tag{2.47}$$

$$\bar{F} = \frac{1}{J}(\eta_t Q + \eta_x E + \eta_y F + \eta_z G) \tag{2.48}$$

$$\bar{G} = \frac{1}{J}(\zeta_t Q + \zeta_x E + \zeta_y F + \zeta_z G) \tag{2.49}$$

$$\bar{E}_v = \frac{1}{J}(\xi_x E_v + \xi_y F_v + \xi_z G_v) \tag{2.50}$$

$$\bar{F}_v = \frac{1}{J}(\eta_x E_v + \eta_y F_v + \eta_z G_v) \tag{2.51}$$

$$\bar{G}_v = \frac{1}{J}(\zeta_x E_v + \zeta_y F_v + \zeta_z G_v) \tag{2.52}$$

$$\tag{2.53}$$

With the transformed shear stress tensor and T,S terms as:

$$\bar{\tau}_{xx} = \mu[\frac{4}{3}(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta) - \frac{2}{3}(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta) - \frac{2}{3}(\xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta)] \tag{2.54}$$

$$\bar{\tau}_{yy} = \mu[\frac{4}{3}(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta) - \frac{2}{3}(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta) - \frac{2}{3}(\xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta)] \tag{2.55}$$

$$\bar{\tau}_{zz} = \mu[\frac{4}{3}(\xi_z w_\xi + \eta_z w_\eta + \zeta_z w_\zeta) - \frac{2}{3}(\xi_x u_\xi + \eta_x u_\eta + \zeta_x u_\zeta) - \frac{2}{3}(\xi_y v_\xi + \eta_y v_\eta + \zeta_y v_\zeta)] \tag{2.56}$$

$$\bar{\tau}_{xy} = \mu(\xi_y u_\xi + \eta_y u_\eta + \zeta_y u_\zeta + \xi_x x_\xi + \eta_x v_\eta + \zeta_x v_\zeta) \tag{2.57}$$

$$\bar{\tau}_{zx} = \mu(\xi_z u_\xi + \eta_z u_\eta + \zeta_z u_\zeta + \xi_x x_\xi + \eta_x v_\eta + \zeta_x v_\zeta) \tag{2.58}$$

$$\bar{\tau}_{zy} = \mu(\xi_z x_\xi + \eta_z v_\eta + \zeta_z v_\zeta + \xi_y u_\xi + \eta_y u_\eta + \zeta_y u_\zeta) \tag{2.59}$$

$$\bar{q}_x = (\xi_x T_\xi + \eta_x T_\eta + \zeta_x T_\zeta) \tag{2.60}$$

$$\bar{q}_y = (\xi_y T_\xi + \eta_y T_\eta + \zeta_y T_\zeta) \tag{2.61}$$

$$\bar{q}_z = (\xi_z T_\xi + \eta_z T_\eta + \zeta_z T_\zeta) \tag{2.62}$$

$$\bar{p}_x = (\xi_x S_\xi + \eta_x S_\eta + \zeta_x S_\zeta) \tag{2.63}$$

$$\bar{p}_y = (\xi_y S_\xi + \eta_y S_\eta + \zeta_y S_\zeta) \tag{2.64}$$

$$\bar{p}_z = (\xi_z S_\xi + \eta_z S_\eta + \zeta_z S_\zeta) \tag{2.65}$$

The central equation to be solved would be 2.45, with the rest of this section's formulas being necessary to resolve it fully. In this analysis, we have omitted non-dimensionalization and body force. These need to be carefully added to the appropriate momentum equation (ζ).

There exists a further step to solve this system according to [47] and is the creation of special flux Jacobian matrices that escape the scope of this thesis but were not included in the transformation because they looked to be very poorly scalable because they are comprised of several of the variables of this formulation, making them extremely big, full matrices. This step, however, is regarded as necessary in literature to solve for the pressure. Instead, the writer opted for a hybrid approach, where the transformation is applied to solve the momentum. The velocities are recovered with the anti-transformed, and the pressure is solved with the usual Poisson equation or another alternative approach, such as the artificial compressibility method explained next.

2.5.3 Artificial Compressibility

2.5.4 Improvements to the HPGF algorithm

2.5.4.1 Status of Implementation

The Flux-Vector formulation was able to solve stratified flows over an underwater seamount in the same sense as the GCCOM does. Still, it could not handle the lock exchange experiment, no further tests were done, but the work is accessible and

testable for anyone with access to the GCCOM bitbucket repository (Arakawa3D). Debugging this implementation is tricky, and maybe better to port a MATLAB version of GCCOM along with this flux-vector formulation to get a better grip on what may be failing in the Fortran code.

The artificial compressibility has good agreement with the hybrid flux-vector implementation (also using the flux-vector formulation). The solution of the artificial compressibility is exceptionally similar to MATLAB tests of the same model implementation done in late 2020, and again all of the work is archived in the repositories. This excellent formulation probably only needs tuning the pseudo-time parameters and is very easy to test and tweak. It is available for future studies in the GCCOM bitbucket repository.

CHAPTER 3

MIMETIC OPERATORS

Mimetic operators are numerical representations of the mathematical operators Gradient ($\mathbf{G} = \nabla$), Divergence ($\mathbf{D} = (\nabla\cdot)$), Laplacian ($\mathbf{L} = \nabla^2 = \mathbf{D*G}$) and Curl ($\mathbf{C} = (\nabla\times)$). Their formulation is based on finite difference but quickly develops intrinsic and unique properties. Several advantages are inherent to a mimetic operator approach. First, once the operator is obtained, its action is represented in a matrix that becomes a single matrix-product vector for each operator's application. This means a much cheaper computational operation than the typical application of a spatial derivative scheme in a multiple for-loop. Secondly, it is simpler and intuitive to develop a numerical code using analogs of the mathematical operators for each function since these are reutilizable and adaptable to each situation with simple accommodations. Finally, arguably the most important advantage of a mimetic operator approach is their highly conservative nature and the fact that they preserve the approximation order to the problem boundary.

3.1 History and function

The origin of mimetic methods as we know them can be traced back to the idea of summation by parts numerical integration [58], but applied to spatial quantities instead of time-based integrations as is more traditional. In the past, this type of operation was not able to conserve order of accuracy at the border while using nodal grids, something that was partially solved later by using weighted inner products [49]. The Castillo-Grone operators [23], first introduced in 2003, are shown to conserve accuracy at the borders by using a stagged grid and matrix analysis to modify the matrices at the edge with the help of Vandermonde matrices with specific properties. This approach was extended to extra dimensions in time [24] but was still improved upon by removing free parameters and optimizing their bandwidth [31], and showed to

be more accurate than previous versions. Lastly, the Corbino-Castillo mimetic operators have been made available by an open-access library called the Mimetic Operators Library Enhanced (MOLE) [30].

3.2 Development and available implementation

As explained in [23], the main focus when constructing discrete approximations to the divergence and gradient is to satisfy local conservation, global conservation, and the divergence theorem:

$$\int_{\Omega} \nabla \cdot \vec{v}\ f\ dV + \int_{\Omega} \vec{v}\ \nabla f\ dV = \int_{\partial\Omega} f\vec{v} \cdot \vec{n}\ dS \tag{3.1}$$

Where $\vec{v}$ is a vector field and f is a scalar field. Then, by making f=1 in 3.2 and integrating over a single cell, we obtain local conservation and making Ω the whole region we observe global conservation. Discretization that follows these properties are regarded as *mimetic*, and the solution of this equation in one dimension can be obtained by integrating by parts in the [0,1] region as:

$$\int_0^1 \frac{dv}{dx} f dx + \int_0^1 v\frac{df}{dx} = v(1)f(1) - v(0)f(0) \tag{3.2}$$

From this formulation, 1D operators can be created for uniform nodal grids. However, to obtain higher-order operators, order-of-accuracy conservative to the domain's border, the staggered grid must be employed, and the operators must be modified with auxiliary matrices of specific properties. Although this explanation's scope remains outside this thesis, it is enough for us to know that the mimetic operators are designed and created to comply with the following properties.

$$\mathbf{G}f = 0, \tag{3.3}$$

$$\mathbf{D}v = 0, \tag{3.4}$$

$$\mathbf{CG}f = 0, \tag{3.5}$$

$$\mathbf{DC}v = 0, \tag{3.6}$$

$$\mathbf{DG}f = Lf, \tag{3.7}$$

$$\langle \mathbf{D}v, f \rangle_Q + \langle \mathbf{G}f, v \rangle_P = \langle \mathbf{B}v, f \rangle \tag{3.8}$$

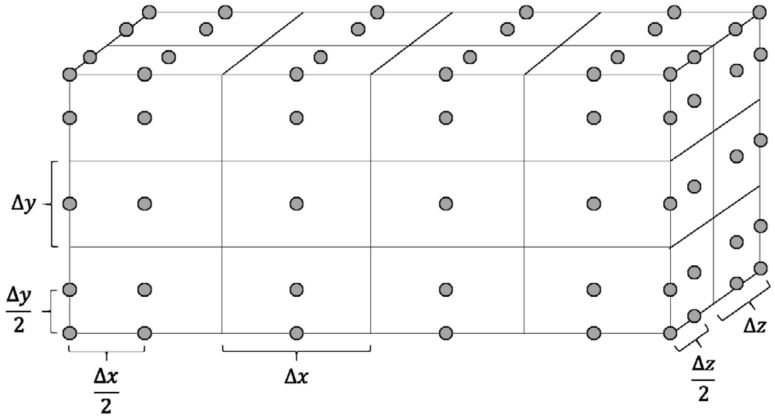


Figure 3.1. Arakawa C-type grid utilized by the MOLE library. From [31].

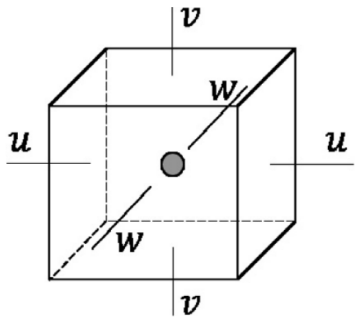


Figure 3.2. Arakawa C-type stencil used by the MOLE library for a single cell. From [31].

In which equation 3.8 is a discrete analogue of equation 3.2 with the addition of operator **B** called the mimetic boundary operator, and **P** and **Q** are weight matrices.

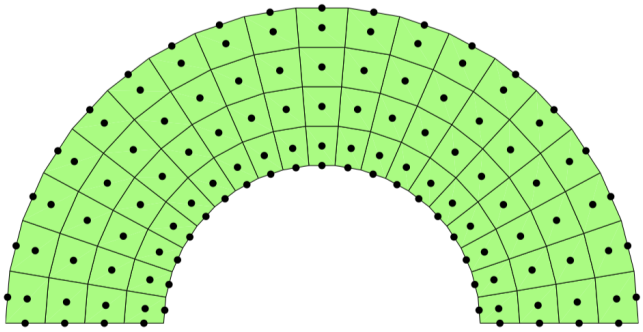


Figure 3.3. Scalar grid point distribution for mimetic operators, to be operated with G. From [2].

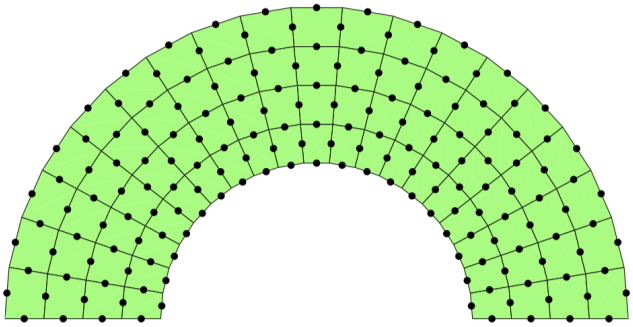


Figure 3.4. Vectorial grid point distribution for mimetic operators, to be operated with D. From [2].

The remaining formulation of the 3D operators arises from the incremental application of the methods described in [23] and refined in [31] to 1D mimetic stencils, then 2D, and so on. However, it is vital to remark some methods utilized for the mimetic operators' constructions in these papers and the MOLE library. First, the grid is staggered with C-type Arakawa stencil [7, 8]. Secondly, the logical-scalar grid, where the f quantities are operated, has a half-step in each direction at the start and end of the grid. Third, the finite difference applied to construct the operators is a central difference. Finally, as we will see later, the choice of building the 3D operators taking the *y-direction*, instead of the *z-direction*, as the vertical, has some unintended consequences down the line that we will explore further.

3.2.1 Previous Mimetic GCCOM-Related Solvers

Before this attempt, a previous effort has been put into making the GCCOM into a mimetic-operated model. This section seeks to recognize that work and frame it in the discussion of this thesis.

In 2013, Abouali [2, 5] created a MATLAB library capable of solving Poisson's problem with Robin boundary conditions on a 2D curvilinear mesh, using the Castillo-Grone mimetic operators. In his article, Abouali transforms the 2D Poisson problem into a diffusion problem in general coordinates with Jacobian:

$$J\tilde{\kappa} = \begin{bmatrix} (y_\nu^2 + x_\nu^2) & (y_\xi y_n u + x_\xi x_\nu) \\ -(y_\xi y_n u + x_\xi x_\nu) & (y_\xi^2 + x_\xi^2) \end{bmatrix} \tag{3.9}$$

The application of Castillo-Grone operators to 2D Poisson equation was satisfactory. However, it implied a correct implementation of $\mathbf{L} = \mathbf{D^*G}$ i.e. the Divergence and Gradient must have been correctly implemented as well. Additionally, this implementation was flexible enough to allow for Dirichlet, Neumann, or mixed (Robin) boundary conditions. Mimetic operators of up to 6th order of accuracy were tested, showing satisfactory error decrease in most cases, but some unexpected error increase in specific accuracy and grid points conditions. The author also reports high frequencies developing near the boundary, especially in curvilinear grids with different smoothness degrees. The product in the form of a MATLAB library can still be downloaded and run from the Mathworks repositories [1].

The main objective of Abouali was to port the entire GCCOM (then called GCM) in a mimetic operator setting, as stated in the final chapter of his Dissertation [2], however, the tools necessary to accomplish this task weren't still developed and he bumped into an increasing error with each iteration that made the solution degrade into instabilities. In Abouali's analysis, the amplification factor of the Castillo-Grone gradient operator was calculated, and it was found that this operator had severe wave-number restrictions to be stable [2]. Finally, because the Poisson problem was already implemented, an alternative semi-mimetic model was envisioned and proposed

as future work. This model would solve a Poisson problem using a mimetic operator and the rest with a traditional finite-difference. A 3D curvilinear Poisson equation solver was prototyped by Abouali in a conference paper [4], yielding better accuracy and fewer iterations needed to converge. Unfortunately, these solvers were never incorporated in the GCM model.

In hindsight, it is understandable that the mimetic operators in their Castillo-Grone formulation were still not suited for computational fluid dynamics simulations, as they were improved in stability and conservative properties years later for the Corbino-Castillo implementation. Furthermore, the high-frequency behavior in advection could also be, partially or completely, caused by the flow of information inside the domain, something we address in our implementation in two ways; One is to separate our problem in a gradient-based advection, instead of a divergence-based one as it is natural to write, and the other is to incorporate the ideas of upwind and downwind from finite-difference into mimetic operators models, with the help of auxiliary interpolation matrices. These ideas will be developed in the following chapter, but the reader may rest easy in that we accomplish the task of formulating a fully mimetic fluid dynamics model in three dimensions.

3.2.2 State of the art

Mimetic operators have been applied to several dissimilar fields with the common necessity of conserving accuracy very well to the boundary. In the Castillo-Grone formulation, shear ruptures have been modeled with slip-dependent friction, generating seismic waveforms in 2D solid media [69]. In contrast, the diffusion equation has been solved in 1D and its convergence studied [44]. Mimetic operators were applied to imaging processing and analysis, where they showed viability for image restoration [11], while also being employed to model seismic waves in vertically deformed grids with free-surface [36] and 2D acoustic wave propagation with the development of a *mimetic leapfrog* method [32]. A 3D tensor notation for the mimetic operators was developed and implemented in [13] to solve a diffusion problem in 3D. Still, to the writer's knowledge, this implementation was independent of the Corbino-Castillo formulation.

This alternative approach is interesting in that it uses the natural lexicographical ordering (*z-vertical*) and the aid of corner grid points to simplify the operators' construction. Recently and using the Corbino-Castillo version of the operators, anisotropic heat diffusion and Richard's equation have been modeled [16, 15, 14], as well as image restoration of glaucoma patients for automatic detection of the disease [10].

As can be gathered from the previous list, most applications of mimetic operators have been in the fields of acoustic and seismic waves, image restoration, and diffusion problems, and almost always applied to 1D and 2D problems, being the only 3D application of the operators the mentioned 3D Poisson solver. Finally, the 3D curvilinear formulation of the Corbino-Castillo operators is unpublished, yet they are already being tested. Therefore, according to the writer's literature review, this works represents the most complete and complex application of mimetic operators in a practical application setting, the first application of mimetic operators to solve Navier-Stokes equations, and the first implementation and use of 3D mimetic operators and their curvilinear counterpart.

CHAPTER 4

THE MIMETIC GCCOM MODEL, GCCOM-MOLE

Given the state of the GCCOM implementation and the privileged position of the writer to use the new version of the 3D curvilinear mimetic operators, it was decided in November 2020 to start a new version of the GCCOM model utilizing the latest version of the mimetic operators available. The task was conceptualized and prototyped during December 2020 and January 2021, when it was decided that a 2D MATLAB version would be ideal to start. Then a 3D version capable of handling curvilinear geometries would be eventually developed. The LE and IWB experiments were chosen as validation sources to compare with the GCCOM results and literature. The goals of this version were: 1.- To use a buoyancy implementation of the body force, which wouldn't be dependent on the vertically-aligned coordinate system, and allowed the use of the fully 3D curvilinear grids. 2.- Improve the maximum time-step able to validate results. The proposed mimetic model has the additional advantage of being the most complete and complex application of mimetic operators to date, showcasing its potential and pioneering its application on the coastal ocean field. In all, this new version of the GCCOM represents a significant step further in both mimetic operator development and coastal ocean modeling, setting up a new frontier for both fields simultaneously.

4.1 Design

The model layout was directly inspired by the validated GCCOM, with the difference of having every simulation element inside the script. The starting point is the grid design, which is created in the first section of the code. This includes grid copies for each of the c-type Arakawa positions, u, v, w, and center. The next step is to define the simulation parameters such as stratification, initial velocities, or any others. Then the mimetic operators are called from the library directly, with the following call:

```
D = div3DCurv(2, X, Y, Z);
G = grad3DCurv(2, X, Y, Z);
N = robinBC3D(2,m,Dx,n,Dy,o,Dz,1,0); %Dirichlet
L = D*G + N;
```

As can be seen, the call uses the X, Y, Z grids, which can be 3D curvilinear and the order of accuracy (2 in this case). For the Laplacian, we use the form $\mathbf{L} = \mathbf{D} * \mathbf{G}$ plus the boundary conditions operator, which is a separate call with the last two arguments controlling the coefficients for Robin boundaries; in the example, the configuration $(1, 0)$ denotes a Dirichlet boundary for the Laplacian operator.

Having our grids, operators, and parameters, in principle, we are ready to start the main calculation, which comprises of the following steps:

- Advect the velocities.
- Integrate the time operator, obtaining a velocity prediction.
- Solve a Poisson equation to obtain pressure.
- Apply the gradient to the pressure. Correct velocities.
- Advect and update temperature, density, and buoyancy.
- Repeat

Early in the mimetic GCCOM development, we detected a rapid destabilization of the velocity fields when applying the first step of this scheme repeatedly; after some analysis, we concluded the problem could be coming from the direction of the information transmitted in the simulation, two things made us believe this. First, the mimetic problems tested on have flow in one direction only, and second, the GCCOM uses a Wicker-Sckamarock forward-backward advection scheme, which is required for the model to work. This way, we were motivated to consider an upwind analog to mimetic operators, which we will discuss in the following sections.

Furthermore, the GCCOM uses the weakly conservative version of the N-S equations, which have the momentum coefficient outside of the operatorś action. As we are designing our new model as close to GCCOM as possible, we chose to keep this version in the Mimetic GCCOM, another reason was instabilities arising when using the

fully conservative version of N-S, but this was diagnosed and tested early in the development, so there may be a way to make it work, this will be one of the further work objectives.

4.1.1 A gradient-based momentum formulation

This gradient-based formulation replaces the natural way of implementing the momentum equation from a vectorial representation into a scalar analog. To the writer's knowledge, this approach to solve differential equations with mimetic operators is entirely new and one of the novelties of this work. The motivation for this inclusion is varied, including being the most successful implementation tested. Still, maybe the most important is that it follows more closely the GCCOM formulation of the N-S equations while at the same time permitting the interpolation of the quantities to the spaces where they are to be operated.

The momentum equation, except body forces, can be written for one of the velocities as:

$$\frac{\partial \vec{u}}{\partial t} = -\omega \times u - \frac{1}{2}\nabla(\vec{u}\cdot\vec{u}) - \frac{\nabla p}{\rho} + \frac{\mu}{\rho}\nabla^2\vec{u} \tag{4.1}$$

Where $\omega = \nabla \times u$ is the vorticity field of the flow. This is called the strongly conservative N-S momentum equation, and it includes the velocity coupling inside the operatorś action. Implementing this version of the momentum proved to be challenging, presumably because of the dot product of two velocities, as each product must be represented in the same grid space, something that is not trivial to do for the crossed terms of different velocities, as they live natively in different positions in the grid, and must be interpolated before operating it.

The same problem occurs if we try using the GCCOM version of the momentum, which instead reads:

$$\frac{\partial \vec{u}}{\partial t} = -(\vec{u}\cdot\nabla)\vec{u} - \frac{\nabla p}{\rho} + \frac{\mu}{\rho}\nabla^2\vec{u} \tag{4.2}$$

Again, we need to interpolate each term of the velocity to the appropriate space before operating outside of the Divergence. The gradient-based momentum solves this problem by rewriting the tricky part of the equation in the following form:

$$(\vec{u}\cdot\nabla)\vec{u} = (u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} + w\frac{\partial}{\partial z})\vec{u} \tag{4.3}$$

$$= (u_u\frac{\partial u}{\partial x} + v_u\frac{\partial u}{\partial y} + w_u\frac{\partial u}{\partial z})\vec{i} + (u_v\frac{\partial v}{\partial x} + v_v\frac{\partial v}{\partial y} + w_v\frac{\partial v}{\partial z})\vec{j} + (u_w\frac{\partial w}{\partial x} + v_w\frac{\partial w}{\partial y} + w_w\frac{\partial w}{\partial z})\vec{k} \tag{4.4}$$

for a velocity $\vec{u} = (u, v, w)$ and the notation u_v meaning component u interpolated to the position of v in the Arakawa C-stencil. In this form, we can apply the mimetic operators in the following way:

$$(\vec{u}\cdot\nabla)\vec{u} = (u_u\boldsymbol{G_x}u + v_u\boldsymbol{G_y}u + w_u\boldsymbol{G_z}u)\ \vec{i} + \tag{4.5}$$

$$(u_v\boldsymbol{G_x}v + v_v\boldsymbol{G_y}v + w_v\boldsymbol{G_z}v)\ \vec{j} + \tag{4.6}$$

$$(u_w\boldsymbol{G_x}w + v_w\boldsymbol{G_y}w + w_w\boldsymbol{G_z}w)\ \vec{k} \tag{4.7}$$

This is the basis of gradient-based momentum, and it also required some more treatment to work. For example, the gradient operator is usually a matrix that includes the three directions at once, so it had to be partitioned into three different matrices, each one with a component of the gradient operator, then used accordingly in the equation and the results added together. Nevertheless, this implementation was by far the most stable and error-resistant that we came up with, and it can be argued that it is also more intuitive to implement.

However, this gradient-based momentum was only part one of the new tools needed to make the Mimetic GCCOM work. The second part was the necessity of controlling the nonlinear effects common in nonhydrostatic environments, that were modulated in the GCCOM by a forward-backward advection scheme [54], which is

equivalent to an upwind scheme, that until now had not been known to be implemented in mimetic operators either.

4.1.1.1 Mimetic Upwind

The advection and convection-diffusion equations are often solved with the aid of upwind schemes, GCCOM does this utilizing the Kawamura scheme [54] which is a higher-order (4th) upwind scheme. Unfortunately, the mimetic operators are based on central differences, so the direction of the flow is not taken into account when solving a hyperbolic equation. This makes the application of mimetic operators to fluid simulations prone to errors [Versteeg & Malalasekera].

By using upwind, we create a framework in which the direction of the flow is calculated before applying the operators by using only the side of the variable that is upstream from the calculation point. The upwind scheme we implemented is based in the compact form, which when solving for a general advection equation such as:

$$\frac{\partial u}{\partial t} = -a\frac{\partial u}{\partial x} \tag{4.8}$$

can be solved numerically using Euler method (for illustration) as:

$$a^+ = max(a, 0),\ a^- = min(a, 0) \tag{4.9}$$

$$u_x^- = \frac{u_i^n - u_{i-1}^n}{\Delta x},\ u_x^+ = \frac{u_{i+1}^n - u_i^n}{\Delta x} \tag{4.10}$$

$$u_{i+1}^n = u_i^n - \Delta t[a^+ u_x^- + a^- u_x^+] \tag{4.11}$$

As we can see from the algorithm, for each position in the velocity u we have an accompanying vector a^+ or a^- which stores the coefficients of the advection in either its real value or zeros, this causes that when we take the one-directional derivative u_x^- or u_x^+ we always multiply of the terms by zero, and take only the element with the information from the upstream flow.

There is, however, one more element we must address before applying the mimetic operator to this scheme. All of the mimetic operators are formulated with a central difference, so the u_x^-, u_x^+ operation cannot be done in mimetic operators directly.

4.1.1.1.1 Mimetic Interpolators

The Mole library, besides the mimetic operator generators, has several auxiliary functions. One of them is a family of interpolators intended to calculate quantities from the center of the cell to one of the velocities locations. With some modifications, and the help of the creator of the library J. Corbino, we have come to the following use of the interpolators to work as one-sided differences:

```
I0 = interpol3D(m, n, o, 0, 0, 0);
I1 = interpol3D(m, n, o, 1, 1, 1);
```

As we will see next, the interpolator $I0$ will be analogous to making u^+ and $I1$ creates the quantity u^-. This enables us to solve the convection-diffusion equation as:

```
Solplus  = I0*T;
Solminus = I1*T;


aplus  = max(uvec,0);
aminus = min(uvec,0);


uDT   = D*(aplus.*Solminus' + aminus.*Solplus');


T_new = SSPRK101D(Tvec,-uDT,dt);
```

In this pseudocode version of the model, we see that applying the interpolators $I0, I1$ to the quantity T is enough to solve the temperature convection equation with a mimetic upwind analog. The last line of the pseudocode is the application of a time integration scheme that we will discuss later in this chapter. We see high frequencies arising rather quickly in the LE experiment without using this upwind method in the

temperature equation. This approach solved that problem and is the primary current approach for LE.

4.1.1.2 Partitioned Gradient and Mimetic Upwind

The previous approach can solve the convection-diffusion equation with mimetic operators with upwind, using the divergence operator and the strong conservative form of the equation. This is because all of the elements in this equation are calculated in the same layout, the central point of the Arakawa C-stencil, while the momentum equations are each calculated in their own space. As we mentioned above, the approach we took to solve this was to partition the gradient operator into three operators, one for the action of each derivative. However, this by itself does not solve the central difference problem, and the use of the interpolators above only yields the quantity $u^{-,+}$, not $u_x^{-,+}$.

To accomplish a partitioned gradient that is also a one-sided derivative, the writer designed and tested, and J. Corbino implemented a new family of operators, which are equivalent to the action of the gradient for one-sided derivative calculations. These operators are presumably mimetic but are not yet tested for the complete list of conditions a mimetic operator must hold.

These gradient operators are called $\boldsymbol{G_{ub}, G_{vb}, G_{wb}}$ in the backwards difference version and $\boldsymbol{G_{uf}, G_{vf}, G_{wf}}$ for their forward difference counterparts, and are summoned by a new function called *d3D* (or *d2D* in the 2D version) as:

```
[Gub, Gvb, Gwb] = d3D(m, n, o, Dx, Dy, Dz, 0, 0, 0);
[Guf, Gvf, Gwf] = d3D(m, n, o, Dx, Dy, Dz, 1, 1, 1);
```

Then, we use the ideas of upwind as well with these new operators, which will finally yield the necessary $u_x^{-,+}$ quantities. This is illustrated next using the momentum of the u-velocity, but the same procedure is applied for v and w.

$$u^+ = max(u,0),\ u^- = min(u,0) \tag{4.12}$$

$$u_x^- = \boldsymbol{G_{ub}} u_u,\ u_x^+ = \boldsymbol{G_{uf}} u_u \tag{4.13}$$

$$u\frac{\partial u}{\partial x}\bigg|_u = u^+ \boldsymbol{G_{ub}} u_u + u^- \boldsymbol{G_{uf}} u_u \tag{4.14}$$

$$v^+ = max(v,0),\ v^- = min(v,0) \tag{4.15}$$

$$u_x^- = \boldsymbol{G_{vb}} u_v,\ u_x^+ = \boldsymbol{G_{vf}} u_v \tag{4.16}$$

$$u\frac{\partial u}{\partial x}\bigg|_v = v^+ \boldsymbol{G_{vb}} u_v + v^- \boldsymbol{G_{vf}} u_v \tag{4.17}$$

$$w^+ = max(w,0),\ w^- = min(w,0) \tag{4.18}$$

$$u_x^- = \boldsymbol{G_{wb}} u_w,\ u_x^+ = \boldsymbol{G_{wf}} u_w \tag{4.19}$$

$$w\frac{\partial u}{\partial x}\bigg|_w = w^+ \boldsymbol{G_{wb}} u_w + w^- \boldsymbol{G_{wf}} u_w \tag{4.20}$$

$$\tag{4.21}$$

The subscript u, v, w denotes the quantity being calculated in that position of the Arakawa C-stencil. This means that the three components of the advection above live in a different space and must be interpolated appropriately. An auxiliary function called `InterpVtoV` has been crafted for the mimetic GCCOM that numerically takes care of the interpolations.

4.1.2 Numerical schemes

For the most part, and whenever possible, the template for GCCOM functioning has been transferred to the Mimetic GCCOM. This includes the equation resolution scheme and order, the predictor-corrector method, and the Boussinesq approximation for the pressure, solving the Poisson equation resulting of taking the divergence of the momentum. Nonetheless, a critical part of the GCCOM functioning has been altered, the time integration scheme.

4.1.2.1 Time-based Integration

GCCOM uses a explicit Runge-Kutta 3 stages (RK3), 2nd order method in a full-storage implementation. In 1980, the work of [96] demonstrated that an optimal storage of 2N can be attained (two storage registers of N-length) to implement 2nd order Runge-Kutta. Since then, [56] showed an extended family or low-storage R-K schemes suited for compressible Navier-Stokes, and more recently, an optimized family of low-storage schemes was described by [57], representing the most efficient R-K schemes to that date.

Strong-stability preserving methods were first introduced as total variation diminishing (TVD) methods in [77] and can be defined in terms of its effective SSP coefficient c_{eff} as:

$$c_{eff} = \frac{c}{s}, \tag{4.22}$$

$$\Delta t \leq c\Delta t_{FE} \tag{4.23}$$

for an s-stages Runge-Kutta method bounded by a SSP coefficient c and a maximum theoretical time step Δt_{FE} that satisfies monotonicity under forward Euler integration, who hence has $c_{eff} = 1$. Explicit Runge-Kutta methods have $c_{eff} \approx 1$ [57].

The RK3 time integration algorithm for the mimetic GCCOM has been replaced by the Strong stability-preserving (SSP) Runge-Kutta method described in [57], which enables for wider stability regions on the simulation, while at the same time providing a solution with only one calculation of the right-hand-side. Specifically, we implemented the low-storage 10 stages, 4th order SSP Runge-Kutta (SSPRK104) from [57]. This scheme was selected because of the simplicity of the pseudocode and because it was the best performing available scheme in many of the GCCOM experiments using PETSc. This method is explicit, and the algorithm, being written in terms of the two N-length objects necessary $q1, q2$, reads:

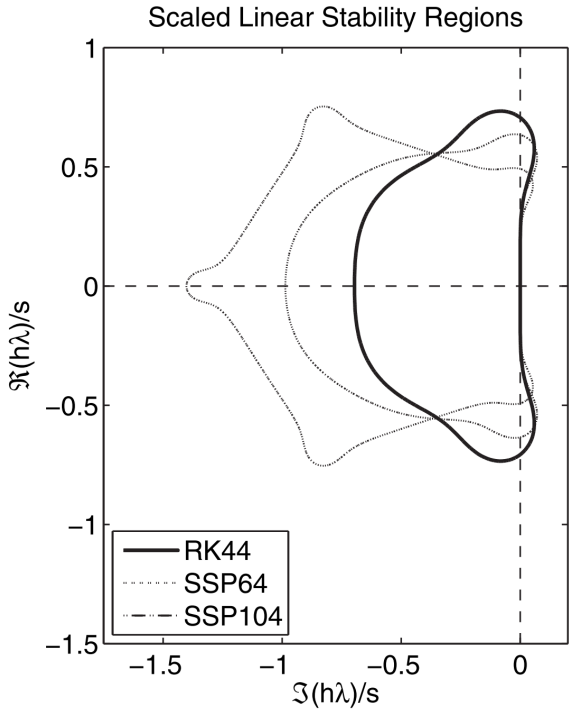
Listing 4.1. Matlab function for a 10 stages 4th order SSP low-storage time integration algorithm. This algorithm advances u a time increase dt using a RHS $F(u)$.

```
function [u] = SSPRK104(u,F,dt)
    q1 = u;
    q2 = u;

    for i = 1:5
        q1 = q1 + dt*F(q1)/6;
    end

    q2 = 1/25*q2 + 9/25*q1;
    q1 = 15*q2 - 5*q1;

    for i = 6:9
        q1 = q1 + dt*F(q1)/6;
    end

    q1 = q2 + 3/5*q1 + 1/10*dt*F(q1);
    u = q1;
end
```

This algorithm represents the first 4th-order 2N storage Runge-Kutta found in literature, and it has $c_{eff} = 0.6$ as shown in [57]. This SSPRK104 algorithm uses the same amount of storage than the low-storage RK3 from [96]. In the Mimetic GCCOM, several full-array calculations are saved with respect to the previous GCCOM version, since the application of curvilinear metrics, interpolations and upwind have been simplified, or are embedded in the mimetico operator being applied, additionally the previous GCCOM used a full-storage version of RK3 which is here reduced on number of objects utilized. However this current implementation doesn't represent an improvement in the number of memory access necessarily, since the increased number of stages may offset the savings in low storage, still this ten stages algorithm is selected for robustness and extended stability, while the optimal second order method (SSPRKs2) from the same paper [57] might be better suited in the future for more computationally heavy simulations. This SSPRK104 implementation attains two important advantages over the previous GCCOM RK3 implementation, first, is 2N storage instead of full-storage, while at the same time being 4th order accurate, making it more stable in nonlinear problems.

The extended stability region of these SSP methods can be appreciated in Figure 4.1 extracted from [57], where two of the SSP algorithms are compared with the traditional 4th order Runge-Kutta method; The SSPRK104 stability region is illustrated to be almost twice as large than the RK4 method. It is important, however, to clarify that the bulk of the extended stability region gained in the Mimetic GCCOM comes from the use of buoyancy as the only body force, and only a moderate advantage (50%) is obtained from the use of SSPRK104.

It is possible that for higher resolution COD problems a second order SSP method such as the SSPRKs2 would suffice, as it would represent a slight increase in $c_{eff} = 1 - 1/s$ for s the number of stages, with the drawback of having to reuse the previous time-step solution, however this is left as a piece of future work since the groundwork being laid out in this thesis requires a robust and encompassing algorithm for best results and stability.



(c) Fourth order methods of optimal methods; SSPs4 denotes the optimal fourth order method with s stages; RK44 denotes the classical fourth order Runge–Kutta method

Figure 4.1. Stability range vs wavenumber for RK4 vs SSPRK64 and SSPRK104. From [57].

During the examination of this thesis, a problem with the implementation of the SSPRK104 method was detected, and a correct implementation was then developed, tested and included in the Mimetic GCCOM repository, this includes a correct update of the right-hand-side function of the quantity being updated. Because of time contraints, however, the results of Chapter 4.2 were not repeated and they have been shown to work with a forward Euler time scheme.

4.1.2.2 Mimetic finite difference hybrid approaches

Thanks to the scripts' modular and highly customizable nature, it is possible to use hybrid FD-Mimetic designs to the Mimetic GCCOM. Several analogs to the GCCOM finite difference derivative routines have been ported to MATLAB for the mimetic model and routines to create the complete set of general curvilinear metric pairs that the FD GCCOM uses and needs. Whenever necessary, some parts of the mimetic model have been approximated with a finite-difference. Such instances are clearly defined as hybrid approaches in the results section. However, they are usually limited to the cases where the convection-diffusion equation where the interpolators $I0, I1$ caused instabilities and errors.

4.2 Mimetic GCCOM experiment results

This section describes the results obtained from the development of the Mimetic GCCOM model, we present qualitative and quantitative validation results for Lock Exchange in 2D and 3D, quantitative validation for 3D Seamount, and working status for Internal Wave Beam, for reasons later described. A number of secondary results, such as 2D seamount and IWB, turbulence model testing, and so on, are not here described, but it is sufficient to say that a fully curvilinear Navier-Stokes mimetic model has been developed, tested and validated for continuously stratified flows in both two and three dimensions, and that all of the work here presented and much more is readily available in version control repositories.

4.2.1 Lock Exchange 2D and 3D results

A 2D version of the GCCOM model was developed and the LE experiment was developed in it as a proof of concept and prototype for the 3D environment. A contour plot of the isotherms at the interface at different time stages is shown in Figure 4.2. In this plot, the K-H billows are seen developing, validating the model qualitatively for this 2D version. In Figure 4.3 we see the progression of the Fr number calculation along both the top and bottom walls with time. As we see, both series show perfect agreement, which is suitable for symmetry arguments. The Froude mean number including the transient values at the start of the simulation is $Fr = 0.6882$, only 2.67% off the theoretical value as can be seen in Figure 4.3 (left), while after the transient settles, the Froude number becomes very stable and ends up being $Fr = 0.7006$ or about 0.92% difference from the theoretical value.

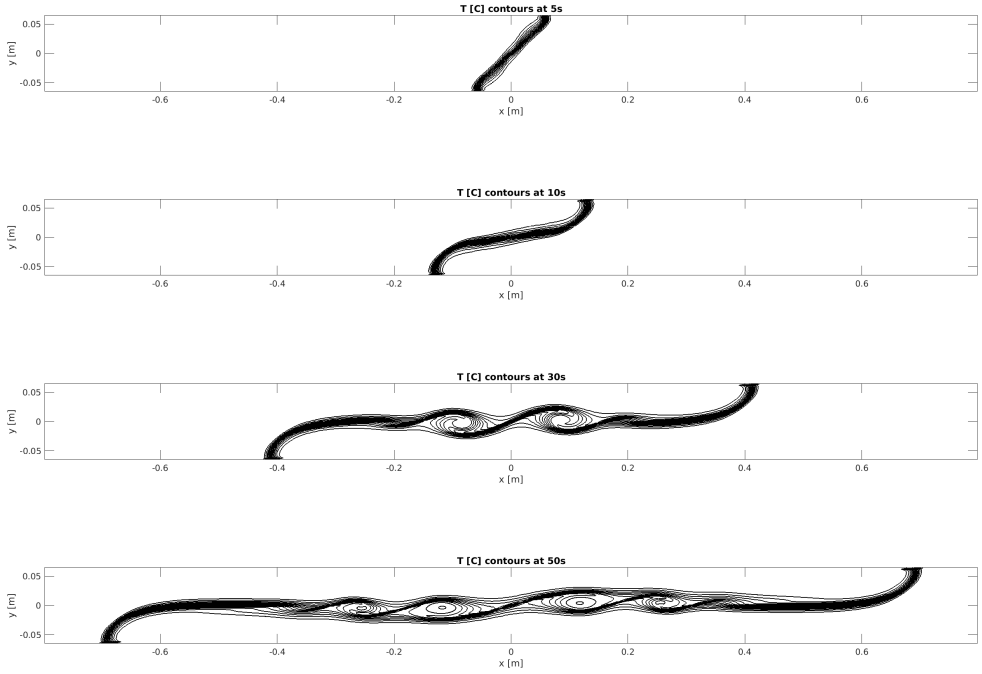


Figure 4.2. Lock exchange experiment results on the 2D version of size 400x100 with rectilinear coordinates.

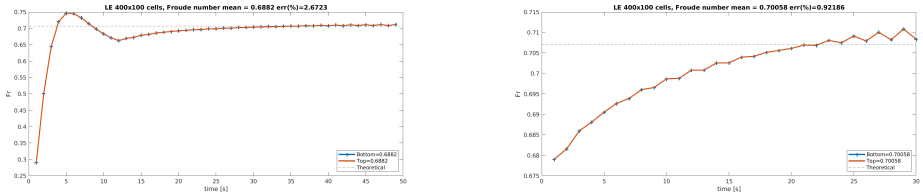


Figure 4.3. Froude number tracking for the LE experiment shown in Figure 4.2. (Left) Froude number mean is 0.6882 including the initial transient. (Right) Froude number mean is 0.7006 after the transient.

In the 3D case of LE, we added the minimum 6 points to the y-direction and two resolutions were tested; one ran with the same parameters as described in [40] or 401x6x101 points, while the second with lower resolution 101x6x101 with the dynamics of both solutions looking the same except for resolution related details. The solution at different times can be seen in Figure 4.4, this time plotting the isotherms in 3D to showcase the nature of this experiment and evoke a similar image to Figure 2 in [46]. The KH vortices are still visible in the isotherm surfaces at $22s$ and $29s$, and the contour plot of these images is very similar to the 2D case.

However, the Froude number of the 3D case resembles much better the theoretical agreement, obtaining $Fr = 0.7051$ or a 0.27% difference in the best case (401 horizontal points, with no-slip boundaries). At the same time, the full resolution experiment means Froude number was 0.739. The lower resolution 3D problem shows a mean Froude number of 0.69679, all of these numbers remain within validate range, as they are lower than other validated ocean models report. The plot from these Froude numbers can be seen in Figure 4.7 and 4.6 and a table compiling the literature results in the same light as the results here presented is shown in Table 4.1, with a bolded font for the experiments that show better results than the validated GCCOM.

One aspect that is important to keep in mind, featured in the GCCOM validation paper, is total energy conservation in the system. Figure 4.8 shows a plot of energy conservation for the 3D lock exchange experiment. On the top panel, we see the potential energy (PE) and kinetic energy (KE) curves for a full cycle; at the start of the simulation PE is maximum and KE is zero, while by the end of the simulation PE has been minimized and transformed into KE which has since peaked. The bottom panel

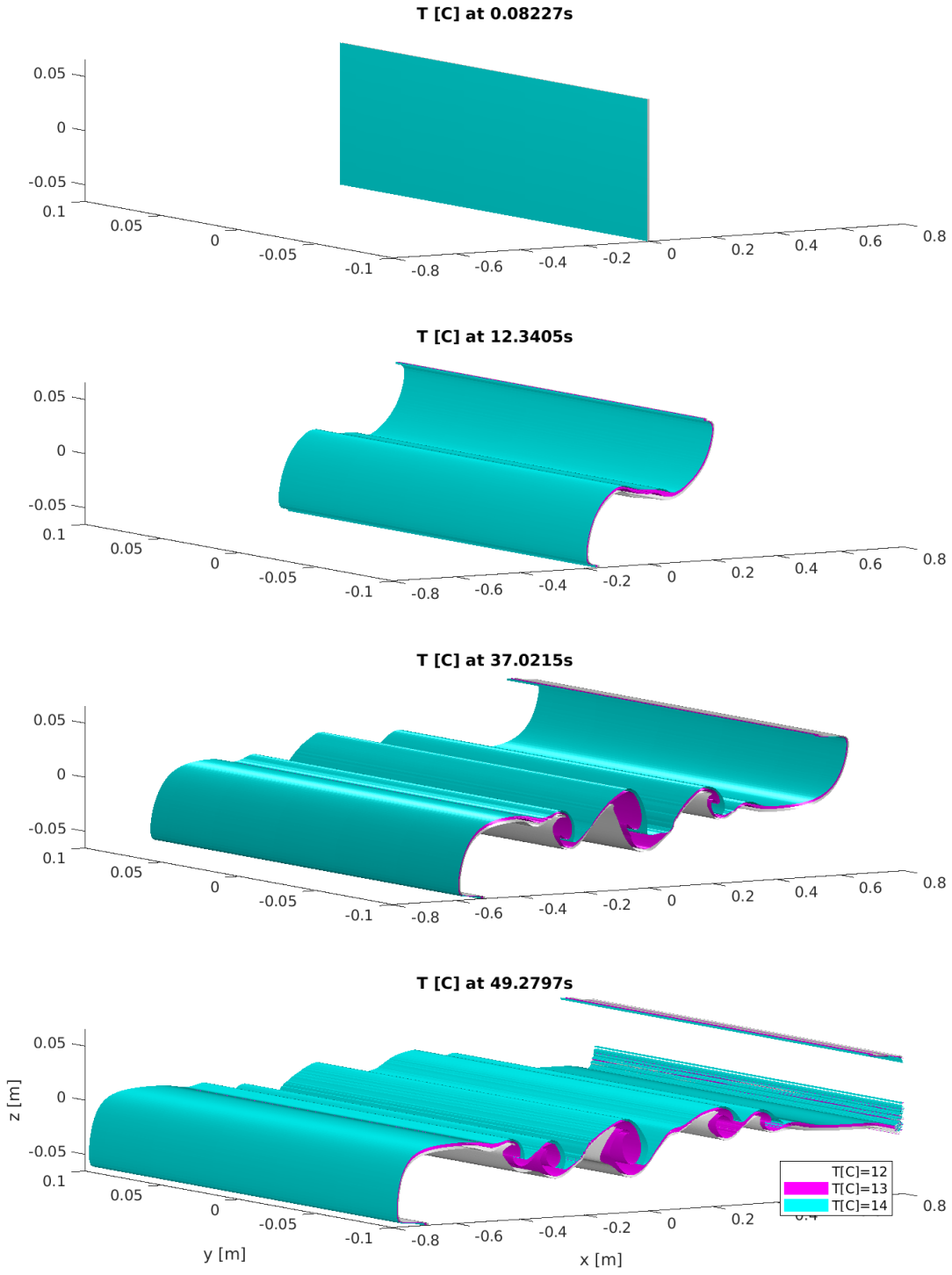


Figure 4.4. Isotherms plot for the 3D lock exchange experiment results of size 400x6x100 with rectilinear coordinates.

Table 4.1. Lock Exchange mean Froude numbers obtained vs literature and percentage errors from the theoretical free-slip Froude value. *Estimated resolution from [46]

Model	Fr	% error	Resolution
GCCOM	0.7176	1.48	401x6x101
Fringer et. al	0.654	7.5	400x100
Härtel et. al	0.675	4.5	768x61x91*
Mimetic-GCCOM 2D	0.6882	2.67	401x101
Mimetic-GCCOM 2D-Tr	0.7002	**0.97**	401x101
Mimetic-GCCOM 3D-FS	0.739	4.51	401x6x101
Mimetic-GCCOM 3D-NS	0.705	**0.29**	401x6x101
Mimetic-GCCOM 3D-LR	0.697	**1.42**	101x6x101

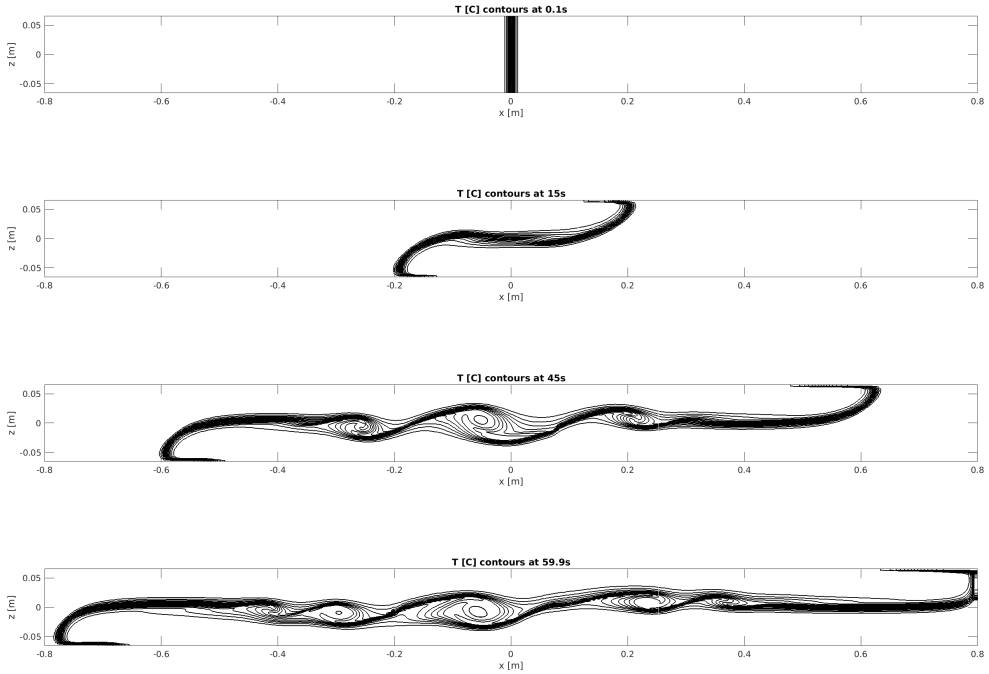


Figure 4.5. $X - Z$ **plane contour plot for the 3D lock exchange experiment of size 400x6x100 with rectilinear coordinates.**

shows the sum of the two energies, normalized by the initial total energy budget, this curve tells us if there is any energy dissipation in the system, and for this experiment, we see basically no energy escaping the system at all, up to the 10^{-4}%. Conversely, we don't have any additional energy being spuriously injected in the system either.

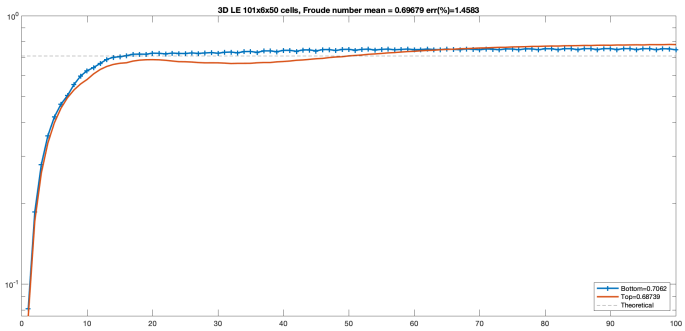


Figure 4.6. Froude number tracking for the LE-3D experiment with 101x6x101 points with free-slip boundary conditions. Froude mean number is 0.69679 or 1.45% **from the theoretical value.**

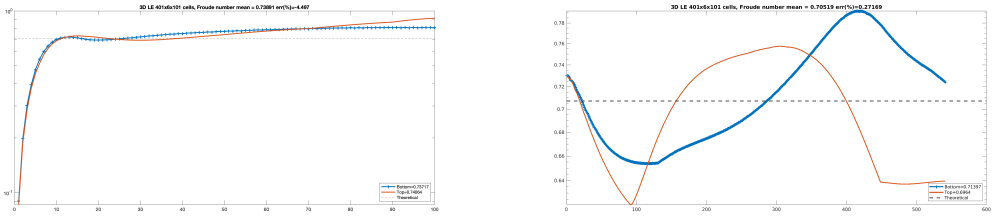


Figure 4.7. Froude number tracking for the LE experiment with free-slip (left) and no-slip (right) boundary conditions. Froude mean number is 0.705 for no-slip and 0.739 for free-slip or or 0.27% **and** 4.5% **from the theoretical value, respectively.**

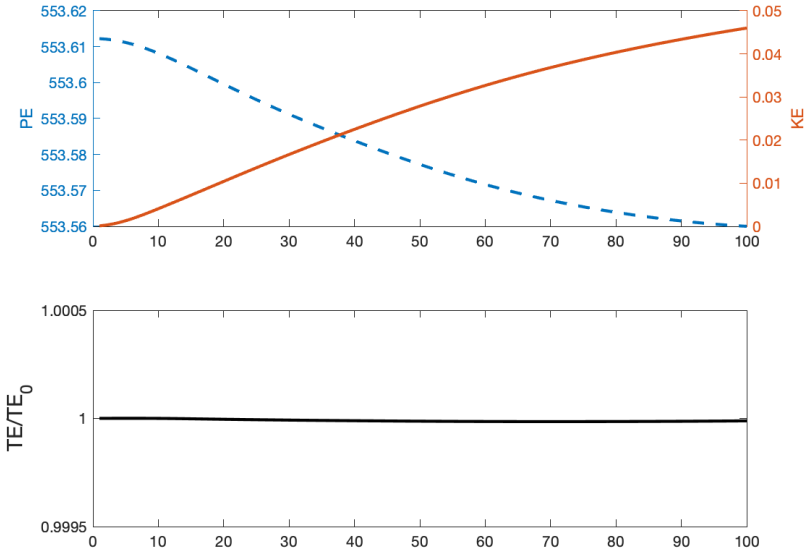


Figure 4.8. Energy conservation plots for the 3D Lock Exchange experiment.

4.2.2 Seamount 3D curvilinear results

For the Seamount experiment, we created a 3D curvilinear grid by varying the clustering β parameter of the grid points in the $X-Y$ and $X-Z$ planes, the front and

side views of this grid can be appreciated in Figure 4.9, having no straight lines in any direction, hence, a fully 3D curvilinear grid in the most general sense. The resolution of this grid aims to match that of [4] except for the y-direction where is only 7 points, this because of the poor scaling of the MATLAB code, the z-direction however exceeds the grid resolution of [4]. Another difference with the paper is the location and general shape of the seamount, which is centered at 0.7 in our case, this showcases the drag of the u-velocity with more space and detail.

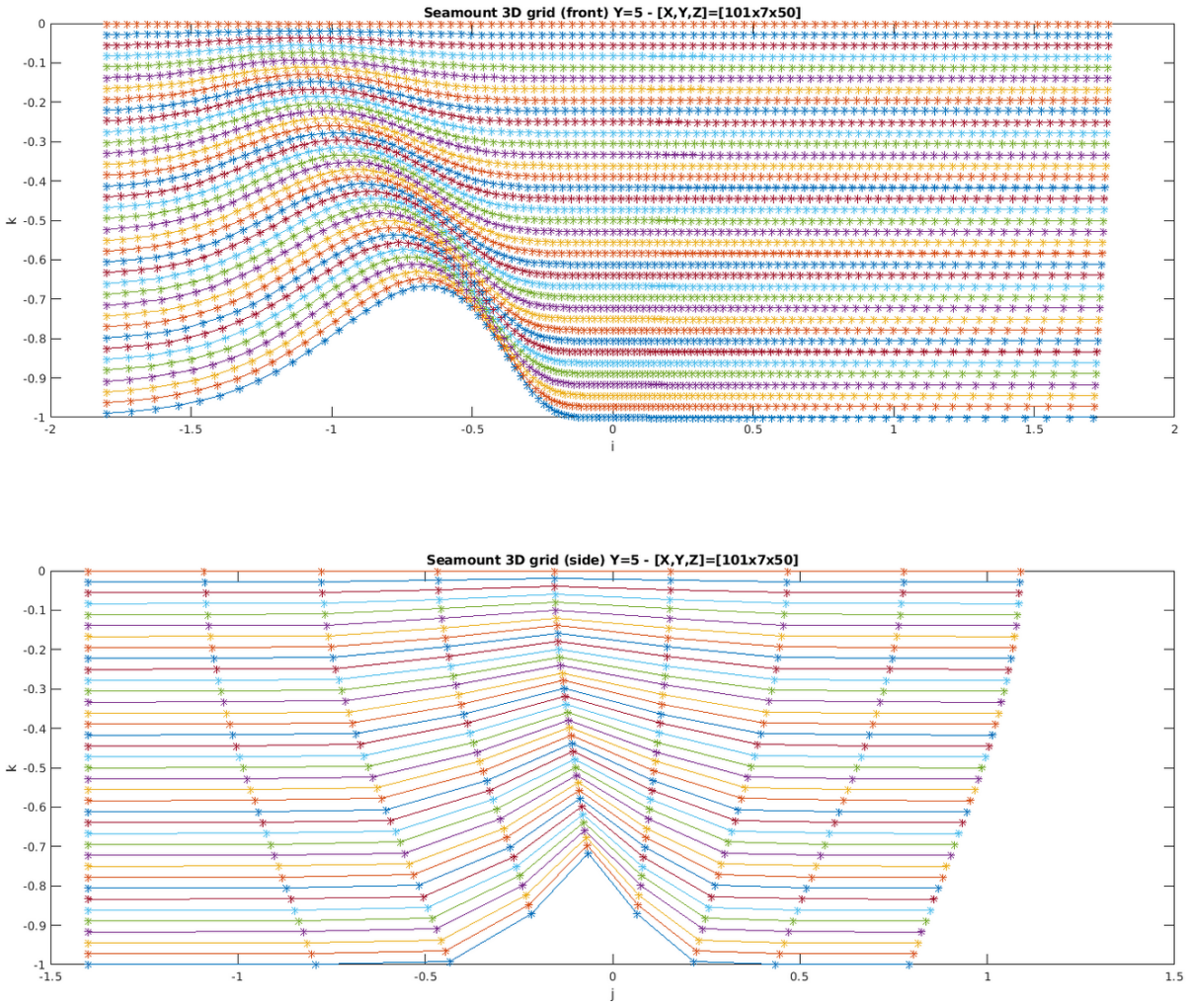


Figure 4.9. Seamount 3D curvilinear grid created from modulating β in the X-Y and X-Z planes.

Figure 4.10 is shown as an analog of Figure 2.2, a contour plot of the u-velocity at different times. The similarities are remarkable; exhibiting the desired dragging behavior that a seamount would inflict in a linearly increasing velocity profile

(top-right) shows the start of mixing occurring, still early in the simulation. In contrast, the other three frames show a much higher mixing degree as it is expected. The full movie visualization of this run is also available upon request.

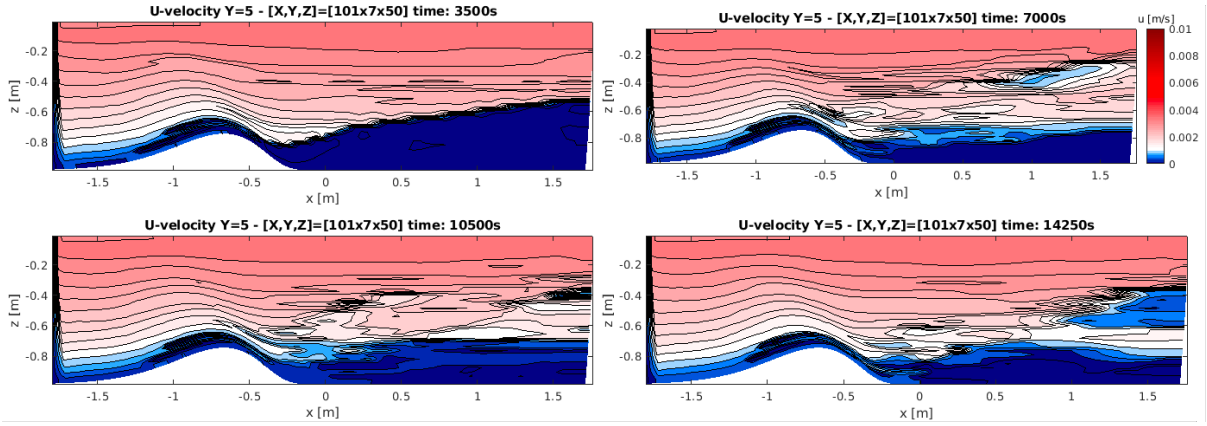


Figure 4.10. Seamount 3D curvilinear experiment results for various time frames.

There are still some problems in this simulation, specifically artifacts that arise at the top of the seamount, expressing as darker, discontinuous areas in the u-velocity. This is most likely, the upwind scheme used since these don't use 3D curvilinear operators but instead one-sided approximations of the rectilinear gradient operator. Unfortunately, at this date, the 3D curvilinear, one-sided, mimetic operators that would be needed to create an upwind scheme to solve the advection equation don't yet exist, and the need for them was only identified after these findings.

In all, the 3D seamount results are satisfactory, showing the potential capabilities of the mimetic GCCOM for fully 3D curvilinear geometries in coastal ocean flows. The general behavior and physics are correct. However, this experiment does not provide quantitative validation of the model, so the validation is only qualitative.

4.2.3 Internal Wave Beam 3D results

Results for the 3D IWB experiment are shown in Figure 4.11, from which we see different ω/N ratios plotted after a few seconds of simulation. The dashed line represents the expected beam angle to be formed for the u-velocity field. Top-left panel shows $\omega/N = 0.2$, followed by $0.4, 0.6, 0.9$ for top-right, bottom-left and bottom-right

panels respectively. There is a clear development of beams in the early stages of the simulation. However, after the expected longer times, these structures dissipate and don't develop as expected.

The Internal Wave Beam experiment needs every element of the model to be fully harmonized and working correctly. The grid in this experiment is sigma, and the grid deformation is minor, making most of the grid rectilinear. Still, the model dissipates the velocities in the areas of interest, probably because of the one-sided gradient-based mimetic advection scheme here developed and employed in first-order in nature. However, remarkably, the beams develop broadly in the expected angles and the first few iterations of the problem, hinting at the model being correct although needing further work to confirm this, specifically arbitrary order-of-accuracy one-sided gradient operators which would enable a higher order mimetic upwind scheme.

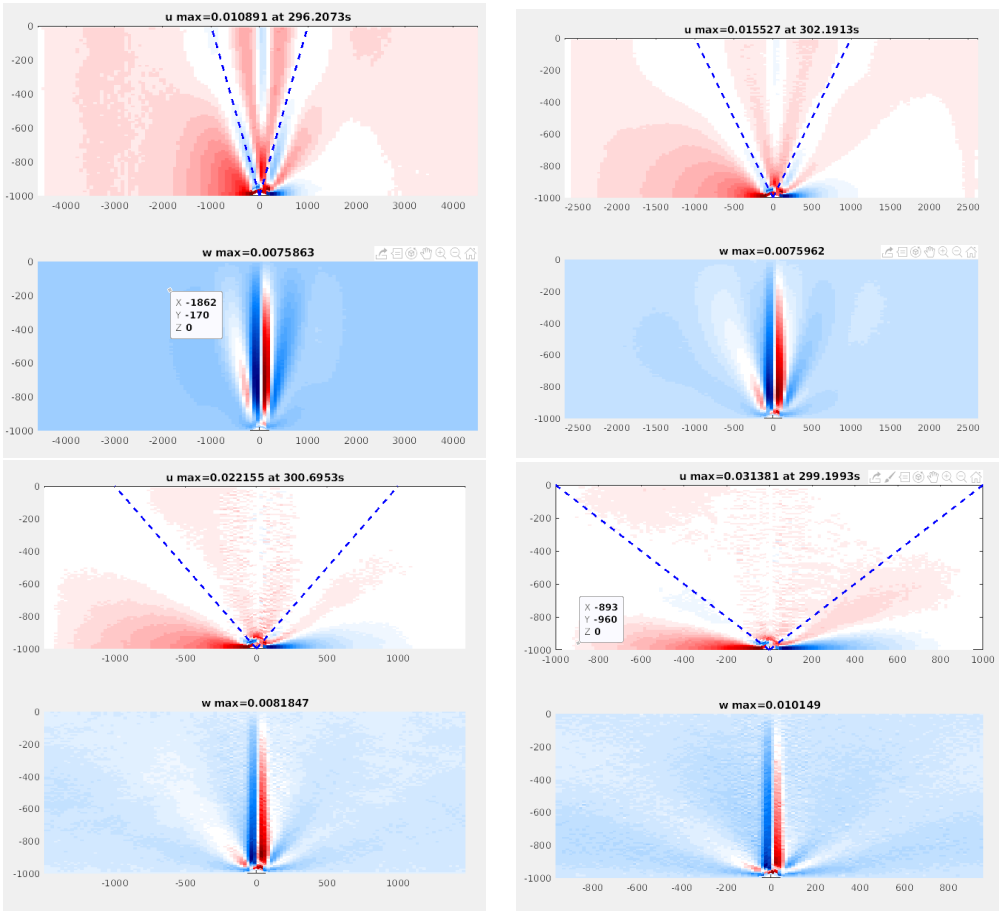


Figure 4.11. Set of different ω/N ratios Internal Waves Beam experiments with the related angle ϕ from the vertical as a dashed line.

This experiment was recreated in 2D, with identical results, to anticipate any possible effects of the 3D implementation taking over the accuracy of the model. The 2D version agreeing with Figure 4.11 clear any doubts that more work needs to be done to attain higher-order in the mimetic upwind scheme.

4.3 Discussion

Both mimetic operators and coastal ocean dynamics models have been advanced with these results. On the mimetic side, the developments of this thesis represent the most complete and ambitious mimetic project to date, even with the current shortcomings. The gradient-based advection implementation is entirely novel and identifies a problem that was unknown to the team before starting this project, which is the need for one-sided mimetic approximations due to the hyperbolic nature of it, while providing a first-order solution to this problem, enabling the successful resolution of the Lock Exchange problem with better performance than the validated GCCOM. Additionally, several auxiliary routines had to be created and implemented, such as interpolation schemes, visualization routines, grid creation scripts. All of these needed to work correctly with MATLAB objects (meshgrid) and the MOLE mimetic operators. These challenges proved to add to the enrichment of the model development and provide a template, a toolbox, and a sandbox for future coastal ocean mimetic model developers.

On the other hand, the gains the GCCOM model have attained with this mimetic version are evident, starting for the exclusive use of a buoyancy term and no other body force or pressure split, and more importantly, getting rid of the constraint of sigma grids, the hydrostatic inconsistency and the subsequent restrictions to time step and parallelization choices. Furthermore, this mimetic model, in its current form, even under the limitations of the MATLAB environment, has a better simulation/wall time ratio than the fully parallelized GCCOM because we are now only bounded by the CFL condition in the time step choice. Still, also we take advantage of Strong-stability preserving time schemes that stretch this choice even further. The Seamount experiment, for example, was able to run with $\Delta t = 10s$ when the GCCOM was

hard-capped to $\Delta t = 0.01$ nondimensional time (in this case seconds because there is a 1:1 scale in the length and velocity scales of this experiment), this is an extreme case with three orders of magnitude acceleration, but an illustrative result. For more strict cases, the Δt choice should be bounded by the CFL condition. The other clear computational advantage of the mimetic implementation is the arbitrary processor work distribution we can attain now, not being restricted to allocate the full vertical water column in a single processor to calculate the HPGF spline correctly (we skip this step now).

There are several other foreseeable advantages of the mimetic implementation. The loops have been all replaced by matrix-vector products, such as the mimetic operations are easily formulated this way. At the same time, the gradient-based implementation makes it more intuitive to implement complicated partial differential equations in a mimetic manner, since in practice, a differential formulation of a PDE system is often preferred to an integral or tensorial formulation. For example in the advection scheme of the Navier-Stokes system, crossed products of the velocities, along with the staggered grid, require the interpolation of most of the quantities in a different position of the stencil, and this needs to be done before the mimetic operations are applied, something that is not practical to do in the tensorial formulation and for most established CFD models, is done in a differential manner, working each direction equation separately. Finally, the mimetic operators conserve energy better because the accuracy order is the same even in the border of the domain, and this is appreciated in the results presented here.

When the project was first presented, some of the limitations now known were not yet identified, yet many other unknowns existed that made this project a formidable challenge. Most of these limitations have been overcome in one way or another, nonetheless, in the end, some knots are left to be untied by the next researcher. The mimetic operator development is a daunting task that was out of the scope of this project. Because of this and time limitations, the internal waves beam experiment could not be completed satisfactorily. However, an alternative 3D curvilinear experiment was

developed and tested; the Seamount shows promising and mostly correct behavior although the advection term is not suited for curvilinear geometries, a surprising result in which most probably, the strong engine of the Laplacian solver is offsetting the shortcomings of the advection solver to give a meaningful result to our simulations.

Future work will first be the necessity of one-sided versions of the mimetic operators or design that enables advection and hyperbolic equations solutions in general coordinates and conditions. Then, a low-level version of this model must be done to scale the problems up to a meaningful resolution for the coastal ocean community. Lastly, more field-scale experiments such as the Seamount and the Internal Wave Beam need to be validated in this model. The author trusts the newer generations to present the work and improve it to its greatest potential, along with bolder and broader applications.

We can summarize the multiple findings and advantages of these results in the following manner:

- Most ambitious mimetic modeling implementation to date.
- Novel gradient-based formulation of Mimetic Navier-Stokes.
- Novel implementation and validation of a Mimetic Upwind scheme.
- Validated for Non-hydrostatic continuously stratified media in 2D and 3D.
- Enhanced time scheme allows for 10x longer time steps.
- Better agreement with physics.
- Better conservation of energy.
- Field-scale capable.
- Paralellizable with arbitrary domain decomposition.

And the future work insight derived from the work here presented would be:

- One-sided mimetic operators.
- Fully 3D curvilinear mimetic upwind.
- Validation of higher-order phenomena in field-scale problems.
- Parallelization and implementation in C/C++

CHAPTER 5

MACHINE LEARNING APPLIED TO COASTAL OCEAN DATA

5.1 Motivation

Machine Learning (ML) and Artificial Intelligence (AI) in general have taken the scientific world by storm. In the last few decades, and with the increasing availability of both quality datasets and reliable ML libraries, the application of automated data processing techniques capable of forecasting and reinforced learning has become ubiquitous. However, each scientific field has its pace to adopt new technologies, and coastal ocean dynamics are still in the infancy of ML and AI applications. The writer had the fantastic opportunity of collaborating with Los Alamos National Laboratory early in his graduate student career in an ML project for graph-based representations of discrete fracture networks [85], something that sparked an essential line of research. While the need for ML applications in coastal ocean dynamics was evident, it was clear that the research grounds would need to be laid out. This chapter describes the studies published in [88] as well as newer experiments that aim to become another publication in the short term.

5.2 Machine Learning Models

Support vector machines are selected as an alternative non-tree-based method ([33, 51, 98, 17, 81, 52, 43]) , which is relatively underrepresented in applications of machine learning to environmental data. We decided to focus on these two techniques, instead of other more complex methods like artificial neural networks, for their straightforward approach and potential wider applicability. The strengths and particularities of each algorithm are discussed in the next section. The sci-kit learn

software is used in this work for machine learning algorithms, as well as model scoring and data selection and sampling for training and testing [66].

5.2.1 Random Forest

Random forest is a supervised algorithm consisting of an ensemble of decision trees ([19]). Each decision tree is constructed using recursive partitioning based on subsets of the variable space as the split points in the tree. Predictions are made by averaging the outcome of each decision tree. Each tree in the ensemble is grown using a different bootstrap sample of the original data. Using random sampling with replacement, approximately one-third of the data is "out-of-sample" or "out-of-bag" for each tree. This out-of-sample data act can be used to estimate predictions and prediction error and variable importance for each predictor variable. In addition, the subset of randomly selected variables tends to decorrelate the trees and produce more diverse trees for highly correlated data. These characteristics have helped popularize random forest as a reliable prediction algorithm used across a range of applications in environmental and biological sciences. Random Forests has become one of the most prominent machine learning methods being used in ocean sciences ([55, 81, 73, 34, 28, 50, 26, 6, 48, 12, 94, 43, 94, 89]) for which it was selected in this research.

In particular, for this work, we implemented random forest regression (RFR), which predicts a continuous quantity as a target. This makes the algorithm useful in a wider range of applications, but at the same time, more difficult to tune and obtain accurate results. Table 5.1 shows the parameters used by the *RandomForestRegressor* scikit-learn function other than the defaults. Both datasets were implemented with the same parameter configuration, and out-of-bag tests were carried out to obtain the optimal parameters for the models. Variable importance is determined using a node impurity measure (also known as Gini importance). This measure is the decrease in node impurities from splitting on a variable, or the reduction of the impurity gained by introducing a split on a specific node, averaged over all trees in the forest.

5.2.2 Support Vector Machines

Support vector machine methods are part of the hyperplane-based methods ([17, 33]). They rely on the idea of finding optimal separating hyperplanes where the data can be separated into categories. Hyperplanes defined between pairs for features, determine areas of themselves as decision boundaries for any further data, defining one decision area or areas for each couple of features and number of categories to be predicted. The algorithm can also be extended to a regression problem by considering the data points within a small distance of the decision boundary line. The best fit is the hyperplane that includes the maximum number of data points. In support vector regression (SVR), the kernel becomes the most critical parameter; this is the function used to find the shape of the separating hyperplane. Kernels have varied complexity functions such as linear, polynomial, sigmoid, or the most popular radial-basis function (RBF). They will define a geometrical area in each hyperplane that belongs to a specific label or value.

For SVR we can formalize the problem as follows: Given training vectors x_i and a response vector y we want to solve the minimization problem:

$$min(\frac{1}{2}w^T w + C\sum_{i=1}^{n}(\zeta_i + \zeta_i*)) \tag{5.1}$$

$$subject\ to\ \ y_i - w^T\phi(x_i) - b \leq \epsilon + \zeta_i, \tag{5.2}$$

$$w^T\phi(x_i) - b - y_i \leq \epsilon + \zeta_i*, \tag{5.3}$$

$$\zeta_i\zeta_i* \geq 0,\ i = 1, \ldots, n \tag{5.4}$$

where w is the solution to the minimization problem $w = (X^T X)^{-1}X^T y$ with X as the feature matrix, ζ_i and ζ_i* penalizing factors, C controls the width of the area of the ϵ tube or separation plane, and $\phi(x_i)$ is the kernel function which maps the data from the input space into the features space, where the problem is solved ([48, 66]).

The function for each kernel used will have specific parameters. To get the most accurate SVR prediction, each of these parameters needs to be tuned by trial and error

or various other tuning methods. For the work presented here we used the scikit-learn SVR algorithm with the nonlinear RBF kernel ($\phi(x) = exp(-\gamma||x - x'||^2)$) where γ is set to one over the number of features. The rest of the specified parameter values for each model can be seen in Table 5.2. In this case, the process to find the optimal values is more exhaustive. The SVR complexity grows quadratically with the number of samples and thus takes a much longer time to train than RFR. Additionally, each kernel can add further overhead to the calculations. After an exhaustive grid search strategy, we obtained the parameters shown in 5.2 for offshore and nearshore cases.

Table 5.1. RFR parameters used other than default.

RFR	offshore	nearshore
n_estimators (trees)	400	400
criterion	mse	mse
max_features	log2	log2

Table 5.2. SVR parameters used other than default.

SVR	offshore	nearshore
Kernel	RBF	RBF
C	1000	100
γ	0.20	0.20
ϵ	0.0031	0.019

5.2.3 Artificial Neural Networks

Artificial neural networks (ANN) ([63, 70]) are highly interconnected individual fitting functions of some sort, forming a network in analogy to the biological brain. Each of these functions is regarded as a neuron in the network. This similitude comes from both the biological and artificial neurons receiving signals from their connections, activating after accumulating some threshold signal and finally firing off an output signal of its own. The network architecture, layers, activation functions, and connectivity will ultimately define the ANN application. For example, a simple feed-forward, the back-scattering neural network will assert as an expanded regression model. Meanwhile, much more advanced ANNs can recognize ordinate amounts of data even in spectral or graphical form, becoming the cornerstone of the modern face and voice recognition algorithms, self-driving vehicles, deep learning, and many other applications.

In a feed-forward ANN, the information flow between the input and the output layer in one direction. Each neuron is connected to all neurons in the next layer but to none in the same layer, these links have an associated weight dependent on how much the error function was minimized. The operation of a regression-based ANN neuron can be written in the following form:

$$y_j = f\left(\sum_{i=1}^{n} x_i g_i - b_j\right) \tag{5.5}$$

Where f is an activation function and g_i the weight associated to the link from input x_i into current neuron j along with a bias b_j. The exact weights values for each link are unknown until the output layer is reached during the training process. The response is assessed, and the information is back-propagated by assigning values to them.

5.2.3.1 Recurrent Neural Networks

A specific kind of feed-forward neural network, Recurrent Neural Networks (RNNs), uses their internal memory to process sequences of inputs, creating a directed graph sequence that allows for temporal connections to be made. These can be applied to different kinds of corresponding data, such as handwriting or speech recognition, making them ideal for time series analysis. RNNs can be further classified in finite and infinite impulses, depending on if their directed graphs can be unrolled and replaced with a feed-forward network or not.

5.2.3.1.1 Long Short-term Memory Networks

A specific application of RNN is employed in the rest of this chapter for time series analysis, training, and forecast. Called Long Short-term Memory networks (LSTMs), these RNNs manipulate the stored states within the graph, augmenting the data sequences in what are called forget gates. LSTMs avoid the vanishing gradient problem (in which the memory of the node about the previous nodes can vanish quickly, preventing the network from further training, or losing the information of the recurrent

sequences) by permitting the errors to flow backward in the network unlimited times, in a mechanism called Constant Error Carousel units (CEC), this enables LSTMs to learn tasks that require information from even millions of steps behind in the training sequence. LSTMs have been successfully trained to recognize languages by speech recognition, read handwriting, and more popularly, predict time series. A schematic of a possible LSTM architecture can be seen in Figure 5.1, with branching paths signifying data copies, yellow boxes for different pointwise data operations, and five different layers as orange boxes, labels for the Forget (F_t), Input (I_t), and Output (O_t) gates can also be appreciated. Each LTSM network will be specifically tailored for the problem to be solved.

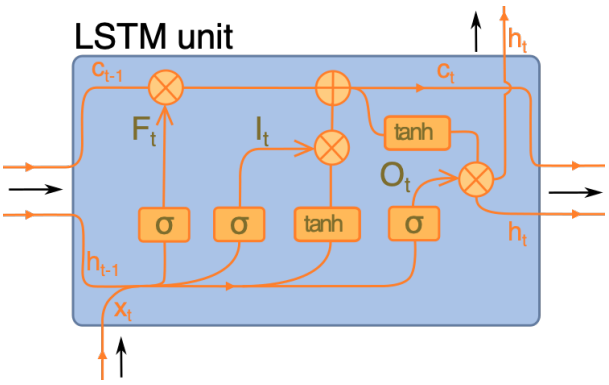


Figure 5.1. Diagram of a Long Short-Term Memory unit. The orange boxes represent layers while the yellow circles are pointwise operations. Each fork in the scheme repesents a copy of the data.

5.2.4 Time Series forecasting

A time-series forecast, in some sense, is similar to a regression in the mind that we are looking to estimate the next value of a series from an extensive dataset we know came from similar conditions. The main difference relies on that time-series data correlates them. At the same time, the regression does not care about the order of the samples. The time series training much retains the information of the time series somehow, with the ultimate goal of preserving the data correlation in time. However, there are many ways in which the data may be formatted in a time series context. First, we must consider that time series forecast is usually done in a single variable since the

values we are trying to obtain lie outside the dataset we train on (the future). Still, several variables may be used as input, called multi-variable problems. Likewise, a time series forecast problem usually solves for a single new value at a time, but it is also possible to solve for several outputs at a time. These are called multi-output variables. By the end of this chapter, we will be describing multi-variable, multi-output time series, forecasting models.

5.2.4.1 Multi-variable forecasting with lagged variables

One way to define a supervised machine learning problem while preserving the temporal correlation of the data is by using lagging variables. A one-dimensional time series can be reformatted with as many lagging variables as needed, but an overfitting tradeoff occurs when a balance must be maintained. We can see an illustration of the lagged variables process in Figure 5.2.

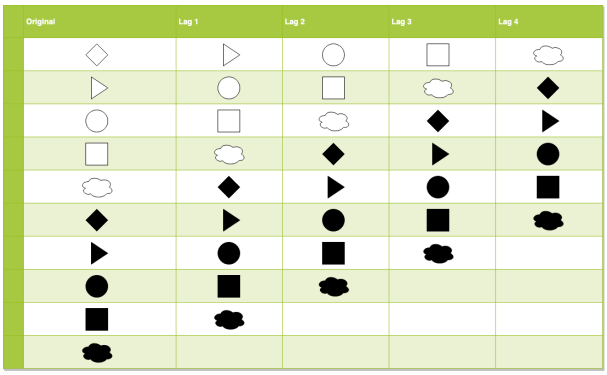


Figure 5.2. Schematic of the lagged variables data augmentation process. Each symbol represents a value in time, and each subsequent lagged column adds a lag of one position as a new time series.

In Figure 5.2 we see an *original* time series comprised of a series of symbols which represent unique values. The lagged variables process is a data augmentation technique that aims to preserve the embedded correlations of time series, by creating derivative *lagged* time series as independent variables. The example in Figure 5.2 shows a one-lagged time series four different times, each one labeled *Lag 1, Lag 2, Lag 3, Lag 4* respectively. In this example, we have expanded our data from one single feature (the *original* time series) to a five features space, with the drawback of reducing the number

of usable samples by 4, or the empty cells at the end of the last column. As we deal with time series of hundreds of thousands samples, this is a reasonable drawback to deal with.

Each lagged variable is an additional independent variable, as this can be assessed by calculation the autocorrelation function of the system, this is done later for each of our models and in every case most, if not all lags fall below of the 95% percentile mark. Adding other time series from different variables is possible, and these can also be lagged. In this thesis we set up as many lags as the maximum forecasting period is expected, this ensures that we have coherent information from a time window similar to the one to be forecasted, this means we have 40 lagged variables per variable used.

5.2.4.2 Multi-output iterative forecasting

A one-step time series forecasting model can be extended to produce more extended forecasts by an iterative process. This process consists in making one-step forecasting at a time, taking the predicted solution as actual data for the next step, and repeating this process as many times as desired. The result is a longer time-frame forecast which works proportionally as well as the one-step forecast. The trade-offs are one, a decreasing accuracy with more extended forecasts, and two, the assumption that we can predict each of the variables in the model. The first trade-off can be minimized by tuning the number of lags in the model, while the second trade-off is just a design choice that is not possible to address by itself and need to be handled carefully.

5.2.4.3 Workflows

Many time-series forecasting models can be combined to strengthen the predictions. In the case of multi-variable, multi-output iterative models, a way to ensure that the additional variables are also forecasted is to pre-train and pre-forecast the extra variables in a separate model, then replace the data necessary for the iterative forecast with the pre-forecasted variable, re-create the lags as needed, and forecast the multi-variable model iteratively. This strategy makes it possible to handle the second trade-off listed in the previous section.

5.3 Machine Learning experimental results

The results shown in the first section of this chapter were published on [88] and are here included by a suggestion of the doctoral committee, as a way to present the other facet of the author's work during graduate school, as well as to provide a context for the second section of this chapter, which is entirely novel. It is expected to produce an additional journal publication.

5.3.1 Offshore DO estimation

The first two models presented in the DO estimation are located offshore, in a transitional zone between the open sea and coastal ocean. In this regime, the DO relation with other variables is expected to be linear, and these models are expected to work as a proof of concept before moving to more non-linear regimes. In this regime the machine learning application is justified by the novelty, but traditional regression approaches would still yield very good results, as these have been used for decades in the open ocean [95, 76] and this data is located offshore and thus still linear in some degree.

Figures 5.3 and 5.4 show the results of the RFR and SVR models, respectively, the top figure shows the relation plot between observed and predicted quantities, while the middle field in RFR and bottom plot in SVR show the distribution of the residuals of the testing dataset. Finally, for Figure 5.3 we have a different plot at the bottom panel, which shows the relative importance of the variables in the model.

The offshore models results are outstanding and can be seen in Table 5.3, the R^2 is 0.997 and 0.986 for RFR and SVR respectively, this is extremely good in both cases, but random forests performs slightly better, and also provides variable importances, making it more valuable in this setting. The most essential variables offshore were depth (0.40), followed by temperature (0.35) and salinity (0.20), while time and upwelling had a very low relative importance in the model, this is expected since this dataset is taken from cruise data taken in discrete time windows, thus the time variable lacks enough correlation to be of importance, while the upwelling was added to the dataset later as a possible extra dimension who could improve the results, but this wasn't the case. In coastal ocean dynamics, there is a correlation between upwelling and

oxygen content, but this is generally dephased in time by some degree, and this correlation is not evident in this case. These results hint of a possible dimensionality reduction taking time and upwelling out of the model without affecting the models' performance too much. These tests could be explored as future work.

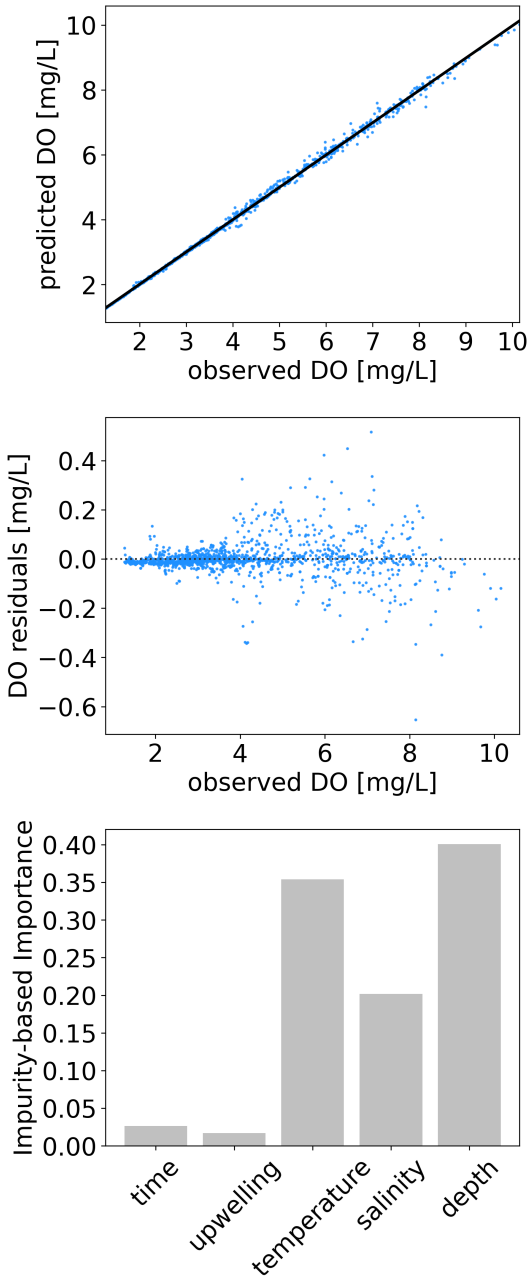


Figure 5.3. Offshore model relational plot for Random Forest Regression (top) with $R^2 = .997$, along with the residuals plot (center) and relative impurity-based importance (bottom).

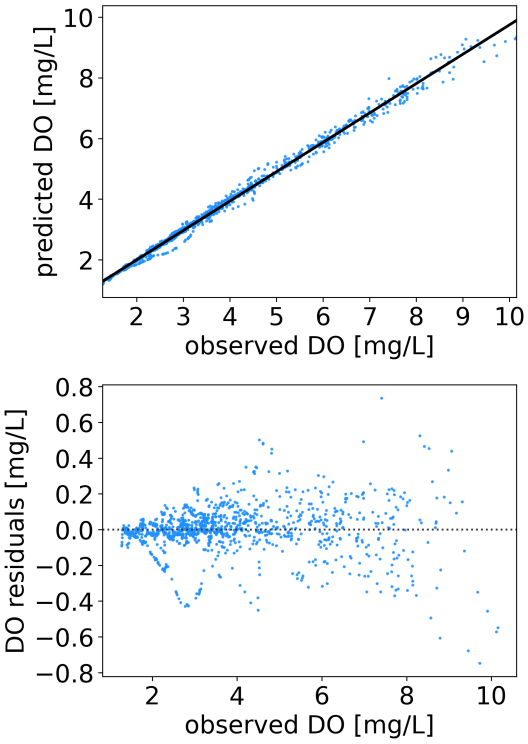


Figure 5.4. Offshore model relational plot for Support Vector Regression (top) with $R^2 = .986$, along with the residuals plot (bottom).

Table 5.3. Offshore oxygen prediction errors results.

	RFR	SVR
R^2	0.997	0.986
mae [mg/L]	0.044	0.089
mse [mg^2/L^2]	0.007	0.045

5.3.2 Nearshore DO estimation

In this section we apply machine learning to nearshore coastal seawater data. In this regime the typical linear relationship between temperature, salinity and dissolved oxygen completely disappear, part of this behavior is because of the multitude of factors that take part on the dynamics of the coastal ocean; Tides, upwelling and shallow depths mean a much higher degree of mixing and turbulence in the seawater this close to the coast, and the well behaved stratified water column of the open ocean is not present here or is not as consistent. Furthermore, developing a numerical reaction-diffusion model to obtain estimates of dissolved oxygen is a daunting task with nontrivial validation requirements. For these reason, a data-driven machine learning approach is best suited for this kind of problem and these results represent the best solution to a problem considered complicated and hard to solve at this point in time.

The results for nearshore DO can be seen in Figures 5.5 and 5.6, again corresponding to the RFR and SVR models, respectively, and with a similar layout to the previous section. In this case, we see a higher variability in the residuals in both models, telling us these results might not be as good as before. Still, more interestingly, the importance coefficients for the variable space have shifted dramatically; now, depth is the lowest ranking variable in importance. At the same time, time is the most important variable with 0.40, followed by temperature (0.33) and salinity (0.20); upwelling is consistently low in the importance ranking. These importances now tell us that we might want to cut depth and upwelling as predictor features to simplify the model further. In this case the dataset comes from continuously recording buoys with depth profiles, so while we have data from the whole water column, is more useful for the model to understand the time of the day, the season, or similar date related

relationships to infer DO content, making the depth variable redundant and in turn less relevant. The R^2 for the RFR and SVR models now falls to 0.987 and 0.946 respectively, this is, in the case of RFR, lower by a small fraction but much more dramatically lower in the case of SVR. This is a somewhat expected result since the statistical ensemble of random forests is robust against nonlinear relationships. At the same time, the support vector methods have mixed results in this type of problem. In any case, the SVR results are still excellent, just not comparable to the RFR model anymore. The summary of error results for the nearshore models is presented in Table 5.4.

Table 5.4. Nearshore oxygen prediction error results.

	RFR	**SVR**
R^2	0.987	0.946
mae [mg/L]	0.076	0.182
mse [mg^2/L^2]	0.022	0.091

5.3.3 Station-Based predictions

The final tests carried out in the regression analysis of DO data were, a station-based prediction where training data in one station would be used to predict DO in a different station, and a training percentage analysis to quantify the amount of training data needed to get accurate enough results.

The results of the station-based analysis are presented in Table 5.5, where the structure is divided into three main categories (self, OS, MB). These correspond to the data used to train the model. This model is then used to predict the targets in the second row of the table (predict). In the case of the left section of the table, the OS, SB, NB, MB stations were trained and used to predict their stations' data, in each case, RFR was used, and the R^2 never dropped from 0.990, giving us reasonable confidence in the models' performance. Next, the middle section trains in data from the OS station; this station is close to the SB, NB, and MB stations, so these were chosen as possible targets for prediction. The prediction power was very limited in this case, with R^2 varying from 0.426 to 0.567. The last case in Table 5.5 was training a model in MB

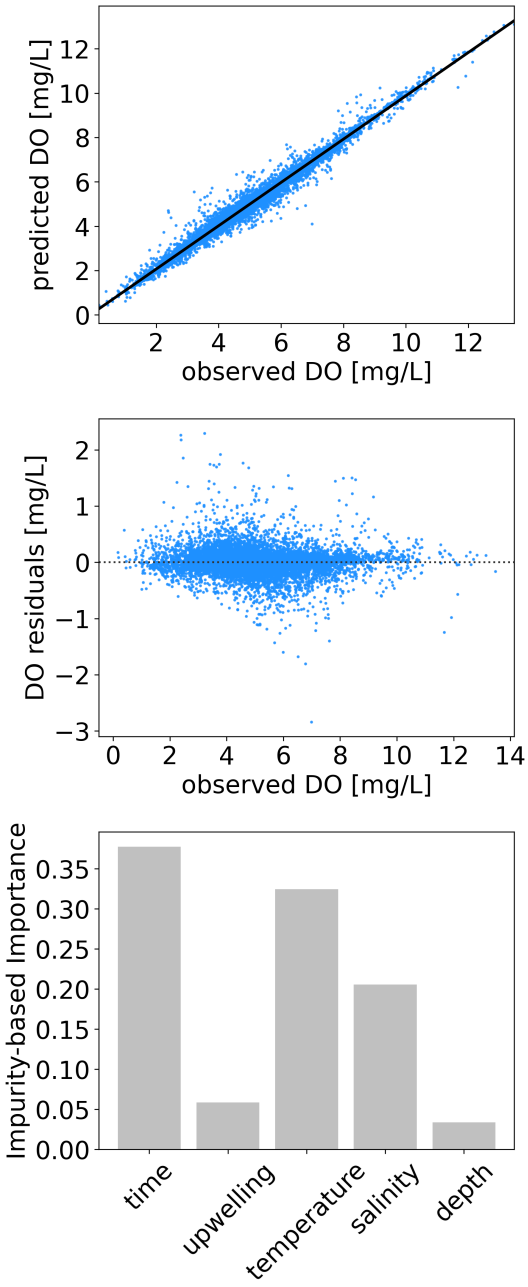


Figure 5.5. Nearshore model relational plot for Random Forest Regression (top) with $R^2 = .987$, along with the residuals plot (center) and relative impurity-based importance (bottom).

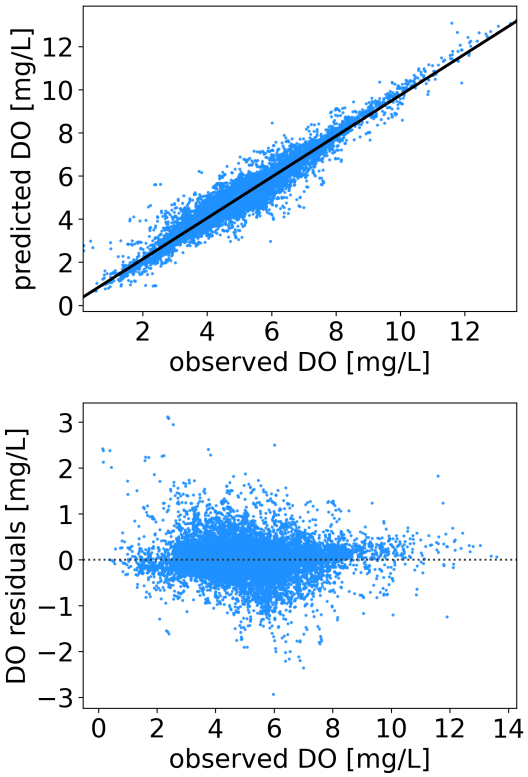


Figure 5.6. Nearshore model relational plot for Support Vector Regression (top) with $R^2 = .945$, along with the residuals plot (bottom).

data to predict SB and NB station data; in this case, some success was obtained for the SB data, but the NB predictions once again fell short, and other methods are suggested to be used in this case. This experiment is interesting because it shines a light on the need to have data from the actual study site to use machine learning appropriately;

otherwise, many physical factors will erase the data validity for a different location. Only good results were obtained in the MB to SB prediction. This is again probably because of physical ocean currents and other natural factors that see a very similar relationship on the two dynamics during the periods of data the model was trained on.

Table 5.5. Station-based RFR error predictions, where the top row shows training the nearshore site used for training and bottom row the nearshore site that is predicted.

Train:	**Self**				**OS**			**MB**	
Predict:	**OS**	**SB**	**NB**	**MB**	**SB**	**NB**	**MB**	**SB**	**NB**
R^2	0.997	0.993	0.995	0.990	0.426	0.567	0.554	0.701	0.438
mae [mg/L]	0.027	0.071	0.040	0.080	0.857	0.493	0.676	0.600	0.632
mse [mg^2/L^2]	0.002	0.015	0.005	0.020	1.351	0.509	0.913	0.704	0.661

Finally, in Figure 5.7 we show a series of curves showing the percentage of training data used for each station's model; the dashed series includes time as a variable, while the solid lines don't. This Figure is fascinating in the sense that for all of the models, using 8-10% of the training data was sufficient to get R^2 higher than 0.90 using time as an additional variable, or 0.85 without it. Thus, this experiment shows that it is sometimes enough to train in only a fraction of the available data for well-tuned models to reconstruct a highly nonlinear quantity like DO in it.

5.3.4 Time Series forecasting results

As described in the previous section, the time series forecasting models here described rely on a re-arranging of time series data in a way that resembles a regression model, using the previous data to predict the next time step, and utilizing lags or differenced periods in the data new, auxiliary features to improve the model. This section will relate how we designed and created time series forecasting models for temperature and dissolved oxygen, using lagged variables and a workflow approach for the DO forecast. We used the forecast of temperature as an additional feature to improve the DO forecasting.

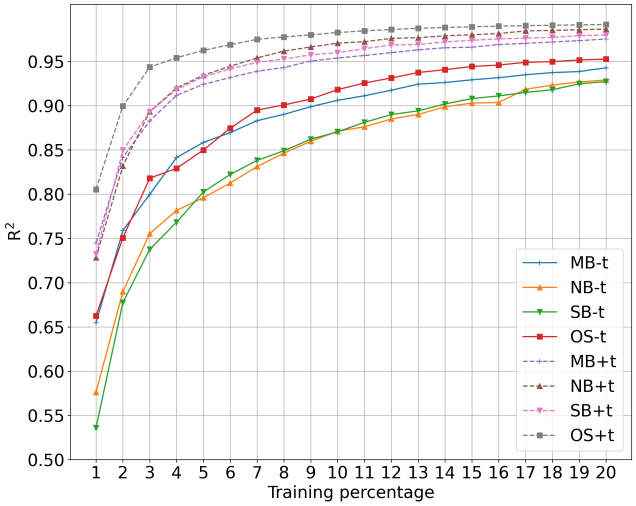


Figure 5.7. Nearshore station-based R^2 scores as a function of percentage of available data used to train the model. "+t" series denote time used as an input in the model.

5.3.4.1 Data Utilized

Tom Connolly provided the original data, comprised of curated records from the Moss Landing sea intake, located in Monterey Bay; the records have a 5 minutes frequency and contain temperature, dissolved oxygen, salinity, fluorescence, Ph, CO2, and other markers. The data starts on 09-03-2010 and ends on 04-03-2021, totaling almost 11 years of high-quality real coastal ocean data. The research team manifested the need to make forecasts long enough to be helpful for the coastal communities that might need the DO information in advance, e.g., fisheries, for which five-days forecasts were defined as the project's goal. Although this requirement forced the time series to be down-sampled to a more manageable amount of data and the number of multi-output points to be obtained, a compromise was found at 3-hours data frequency, which amounts to 40 points of forecast at a time for a five days range.

Auto-correlation function and trend analysis were carried out in the raw data. As a result, a strong seasonal trend as a general increasing trend was identified and corrected, using Augmented Dickey-Fuller (ADH) Test and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test controlled by p-values.

The data is deseasonalized and detrended by differencing it once and then rescaled between -1 and 1 to make it work better with the *tanh* activation function. The lags are created afterward, and an equal number of lags than the desired maximum forecasting period was found to work best, adding 40 lags per independent feature utilized.

5.3.4.2 Model parameters

All of the time series forecasting models described here share the same architecture and parameters. These were tuned manually from the vast array of possible options available, and the number of epochs used was minimized by mapping the loss gains per generation, yielding five epochs as the minimum number of generations to attain good results for us, while at the same time minimizing over-fitting. Whenever possible, the functions chosen were utilized with the default parameters. Tensorflow 2.4.1 was used as a backend, with Keras 2.4 as a frontend for the model's engine.

Specific parameters are listed on Table 5.6 and are as follows: The model was trained using 250 neurons and five generations, with mean squared error as the loss function and root mean squared error as metrics, as well as utilizing the adam optimizer. The network architecture is very simple and can be seeing in Figure 5.8: An input layer equal to the number of lags times the number of features (80 total features for temperature, 120 total features for DO), densely connected to a single LSTM network of 250 neurons, finalized by a dense output layer of size 1, which will become the forecasted value of the next point in time, 3 hours in the future. This configuration is both simple and sufficient to complete the objectives. It is also important to point out that shuffling the model fit must be disabled for time series forecast, and the LSTM must be set to stateful to maintain the time series correlation. Finally, the model states are reset after each generation as a precaution to prevent over-fitting.

Two different models are created, tested, and reported for each interest variable (Temperature and DO), one forecasting for 200 points or 25 days and one forecasting for an entire year (April 2020 - April 2021). The reason was to explore the forecasting performance in a more extended period, especially across different seasons. The first

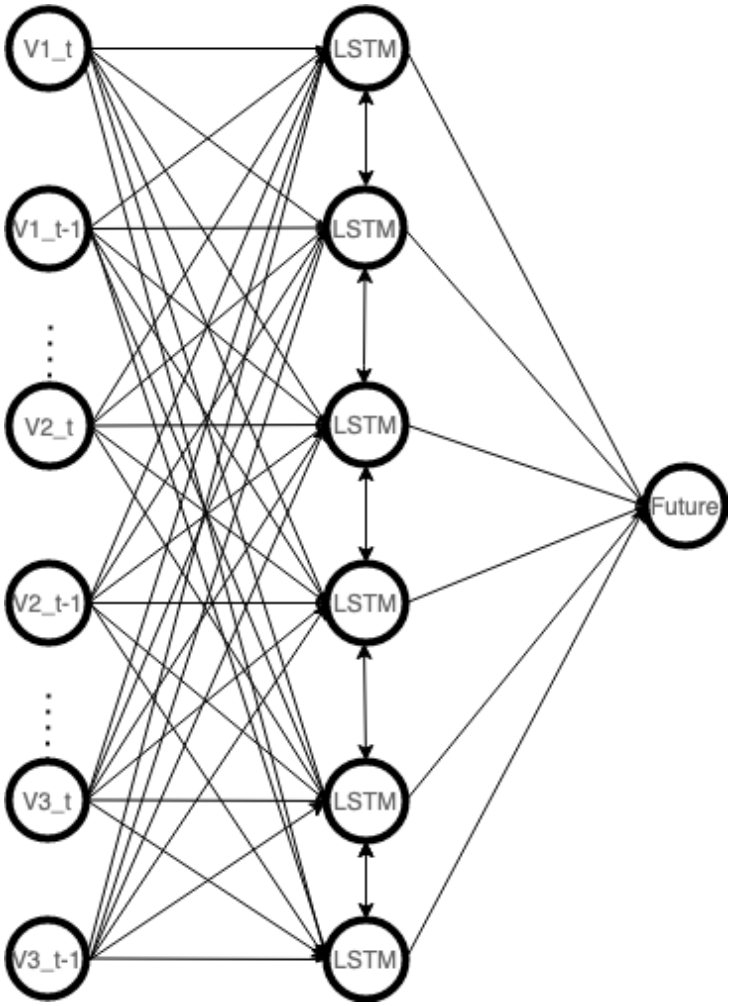


Figure 5.8. Architecture of the LSTM neural networks employed. An input layer of several variables V1_t, V2_t, V3_t, include lagged variables V1_(t-1), V2_(t-1), V3_(t-1) and so on, are densely interconnected to the LSTM neurons and produce a single output for a forecasted reading of the quantity of interest.

Table 5.6. LTSM neural network parameters.

	RFR
Neurons	250
Generations	5
Loss	MSE
Optimizer	Adam
Metrics	RMSE

model splits the data into 97% training and 3% testing, and the second model uses a 90%-10% split between training and testing sets. As we will see, these two models' performances are still similar, but they must be treated as different models since they train and test is different data.

Water level data

The tidal forcing of the site mainly defines the water level in the current context; this is a well-known and highly predictable quantity. As such, it is suited for multi-output iterative forecasts. With this in mind, the water level dataset from NOAA was adapted and merged with the Moss Landing data to include this information as an additional feature of the model. Water level becomes especially useful in the temperature model, where we lack other information to forecast either in one-step or multi-output forecasts. Still, it is also handy on the DO forecast; since we don't have to train and predict an additional variable for the multi-output step, we can assume we have this information beforehand.

In the following sections, we present the model results, four models in total, two for temperature and two for dissolved oxygen, the difference being the amount of training data used, either 97 or 90% for a more comprehensive lecture of the forecasting power across seasons. In both cases, the corresponding temperature models 5-days forecasts are used in the multi-output dissolved oxygen forecast.

5.3.5 Temperature time series forecast

The first model, called T1S1 for temperature one-step, is a function of past temperature and past water level, to produce the following reading for temperature 3 hours ahead, can be seen in Figure 5.9, the plot shows three panels, the top panel over imposes the real temperature data from the testing set and the predicted data from the model, the middle panel shows the residuals of the two series, expressing relative error distribution. In contrast, the bottom panel is a representation of the autocorrelation of the lags utilized with the testing data, autocorrelation being the correlation measurement of past data with current data, the height of the autocorrelation stems describe how much the lagged data resembles the present data, in our models the ideal behavior is to keep autocorrelation at a minimum threshold not to be statistically significant, meaning the lagged data is independent of the present data. This threshold is shown as the shadowed area and is the 5% percentile, indicating the correlation of the lagged data is below this threshold.

Right away, this first model results illustrate the power and limitations of our model; on one side, the general trend of the temperature is captured very well, changing signs and touching minimums and maximums at the same time as the real-time series and overall grasping the behavior needed, on the other hand, the forecast tends to overshoot the values in almost every case. Towards the end of the testing series, the forecast worsens for a short-range; this can also be seen in the residual plot, where the residuals are similarly distributed with higher residuals towards the end. The lag plot shows most of the lags inside the shadowed area, and only some of the first lags being statistically significant for the linear dependence hypothesis. The whole of this figure tells us that this model can probably be improved with additional features, but in its current form is still reliable, and it has not been overfitted, according to the autocorrelation.

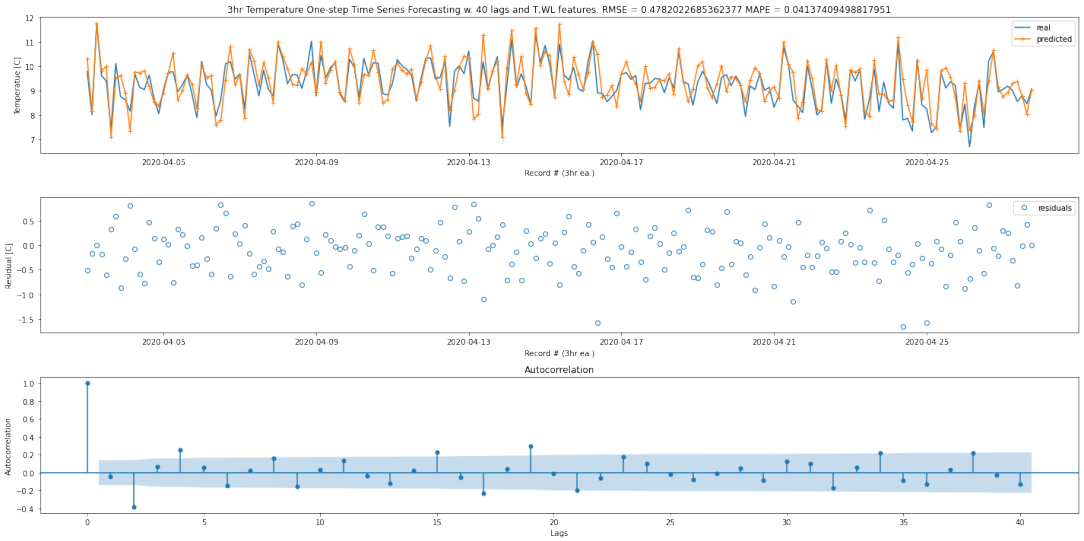


Figure 5.9. Single-output (3hr) forecast of Temperature based on T,WL using 40 lags in 3 hour frequency data, training in 97% of dataset. RMSE $= 0.4782C$**, MAPE** $= 0.0413\%$

The second temperature model, called T1S2, is based on 90% of the training data, which permits a full year of forecasting to be studied. This is particularly important to assess the models' performance across different seasons; the plot for this model can be seen in Figure 5.10, and it follows the same structure as the T1S plot. In this case, the top panel is hard to read. Still, the forecast seems to follow the general

trend of the real data, less so in the middle of the series that corresponds to the months from September to December; this is better seen in the residual plot where most of the residuals at the start of the series are close together, they spread out towards the center of the series and narrow back down towards the end. The lags plot shows weak autocorrelation of the data as well. This model is similar to the previous one in performance and behavior but offers much more robust results after forecasting a full year of data and presents evidence of seasonal variance on the data that becomes harder to grasp for the model.

Figure 5.10. Single-output (3hr) forecast of Temperature based on T,WL using 40 lags in 3 hour frequency data, training in 90% of dataset. RMSE = $0.9982C$**, MAPE =** 0.0617%

The temperature multi-output model performances are described next, in Figure 5.11 we show the results of model T5D1, a sample of ten different forecast ranges are shown, with the over imposed real and forecasted series for comparison, the error measurements are reported for each range and the average errors are seen in the figure description. This model is limited in the forecasting range, but it helps to illustrate the architecture working for our data. The different forecast series seem to follow the temperature trend roughly. Still, the RMSE is at $1.09C$, and the MAPE is about 0.1%, which can be regarded as low. Still, the maximum and minimum agreement left some to be desired in the more abrupt temperature changes, as we will see for the next model,

this might be a localized problem in some parts of the time series but not necessarily a widespread problem, since the errors reported for this model are comparatively high.

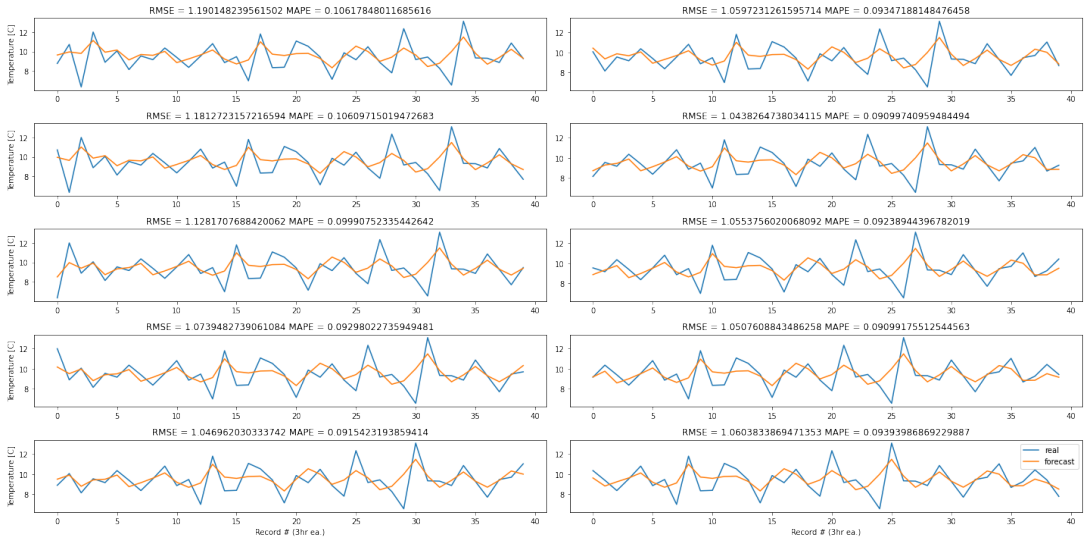


Figure 5.11. 5 days forecast of Temperature based on T,WL using 40 lags in 3 hour frequency data, training in 97% of dataset. RMSE = $1.09C$**, MAPE =** 0.10%

On the other hand, Figure 5.12 shows the results of model T5D2, which has a much more extensive prediction range, one full year, for which the visualization of superimposed time series wouldn't be legible. So instead, this plot reports the error values for every five days forecast on the testing set. The top panel shows RMSE, the middle panel MAPE, and the actual temperature data appears in the bottom panel as a reference. This plot is very interesting since it combines several elements from the previous model in one wider picture, for once, we see a seasonal difference in the errors, again showing the model to have difficulty forecasting the Fall months compared to the rest of the year, next, we have a lower mean error for both RMSE and MAPE than in the previous model, telling us those results are affected by short-range fluctuations in temperature, and finally, this model performance is regarded as very good at RMSE = $0.95C$, MAPE = 0.06%, so it can be used in a workflow chart as a pre-forecasted quantity for more complicated markers, something we will do in the multi-output dissolved oxygen models.

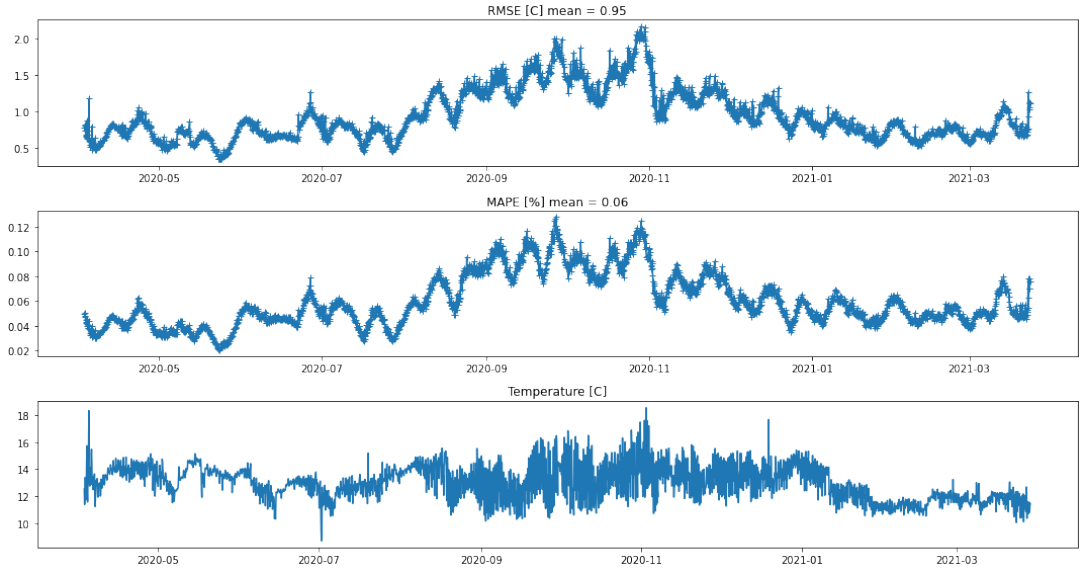


Figure 5.12. Error for one year of 5 days forecast of Temperature based on T,WL using 40 lags in 3 hour frequency data, training in 90% of dataset. RMSE = $0.95C$, MAPE = 0.06%

5.3.6 Dissolved Oxygen time series forecast

The first dissolved oxygen model is shown in Figure 5.13, for the model O1S1, which trains in 97$ of the data. This model uses data from lagged water levels, temperature, and oxygen in the past to predict the next step. As we see, the agreement, in this case, is much better than in the temperature models. This is possible because of the high correlation between temperature and oxygen content, an advantage that the temperature models don't have. The errors are extremely small, RMSE = $0.1284\mu mol/L$ and MAPE = 0.0005%. Like the temperature models, only some of the lowest lags fall above the threshold established for random auto-correlation, which gives us confidence in the model not being overfitted.

In the case of the O1S2 model, training in 90% of the data, the behavior is very similar and can be appreciated in Figure 5.14, only in this case the errors are slightly higher at RMSE = $0.1465\mu mol/L$, MAPE = 0.0006%. This can be thought of as counter-intuitive at first glance, but when we notice the residuals plot (middle panel) we see most of the year, the residuals are minimal and are only around the Fall and less so Spring months than the errors increase. Remembering our data records end during the Spring, this error discrepancy makes sense. The autocorrelation in model O1S2 is also

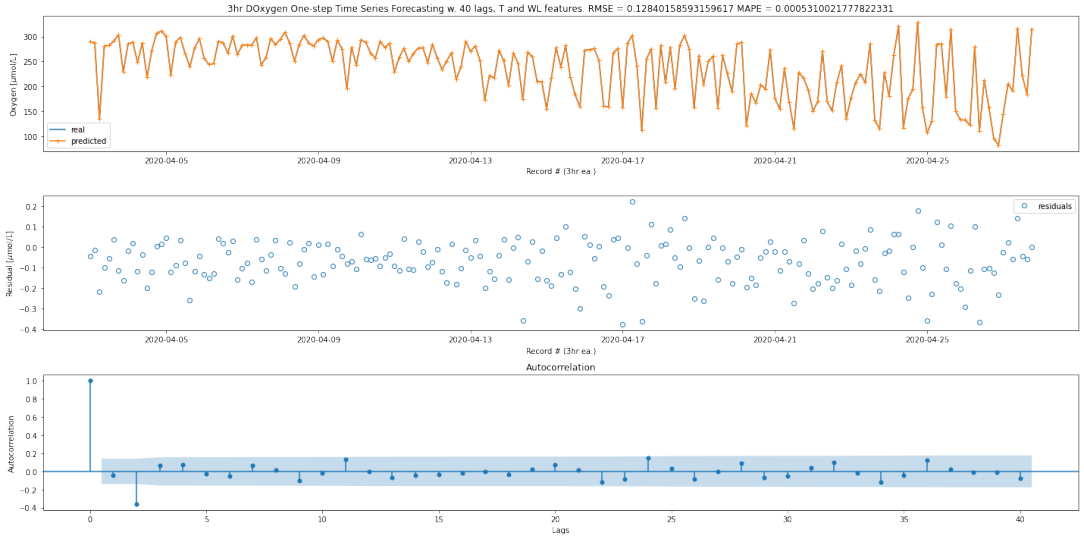


Figure 5.13. Single-output (3hr) forecast of Dissolved Oxygen based on DO,T,WL using 40 lags in 3 hour frequency data, training in 97% of dataset. RMSE $= 0.1284 \mu mol/L$**, MAPE** $= 0.0005\%$

signaling good linear independence between lags. In general, this model shows outstanding performance, and it lays down the ground needed for the multi-output forecast, which is expected to have higher errors associated just because of its design.

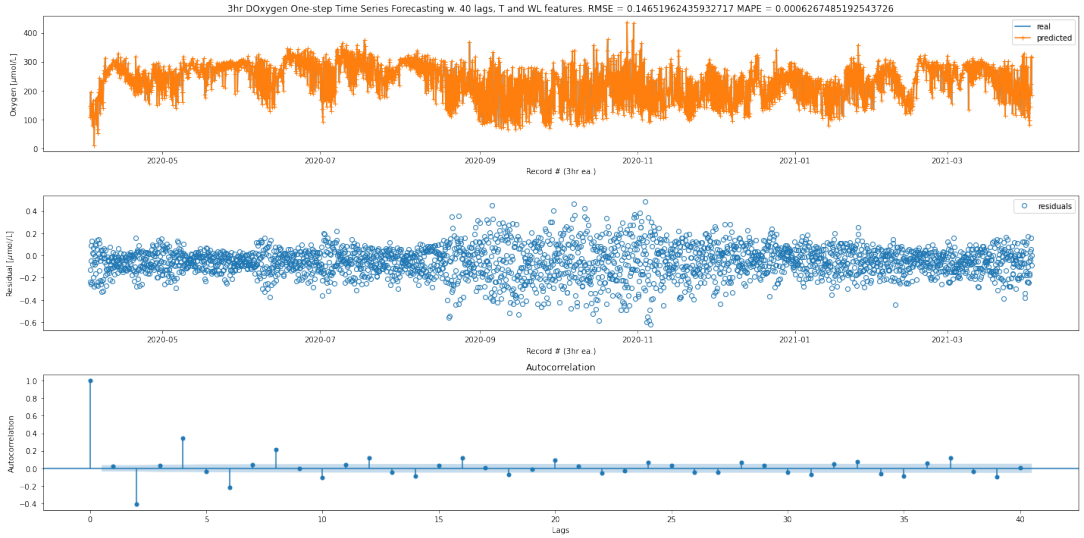


Figure 5.14. Single-output (3hr) forecast of Dissolved Oxygen based on DO,T,WL using 40 lags in 3 hour frequency data, training in 90% of dataset. RMSE $= 0.1465 \mu mol/L$**, MAPE** $= 0.0006\%$

In the following two models, we use a pre-forecasted temperature to complete the iterative multi-output process. This is a necessary step for the model to be practical since an iterative multi-output forecast assumes the provided data is all available.

Unfortunately, there are no reliable forecast models for temperature in the coastal region that the author or the research team knows about. A forecast for the water level, on the other hand, is available. The alternative option was to exclude the temperature as a feature in the dissolved oxygen models, but this would mean poorer forecasts. The presented strategy was employed instead once the temperature forecast was proven to perform well enough.

The first multi-output iterative five days forecast model, trained in 97% of the dataset, called O5D1, can be appreciated in Figure 5.15 and shows a grid view of ten prediction ranges, each one with its errors reported on top. From these plots, it is remarkable how well the oxygen forecast follows the trends in the real data. Although not perfect, most minimums and maximums are synchronous, with some discrepancies in the values. The errors for the set of forecasts in this model are RMSE = $41.36 \mu mol/L$, MAPE = 1.60%, very far from the errors in the one-step models, which is expected, but still, an excellent estimate attained.

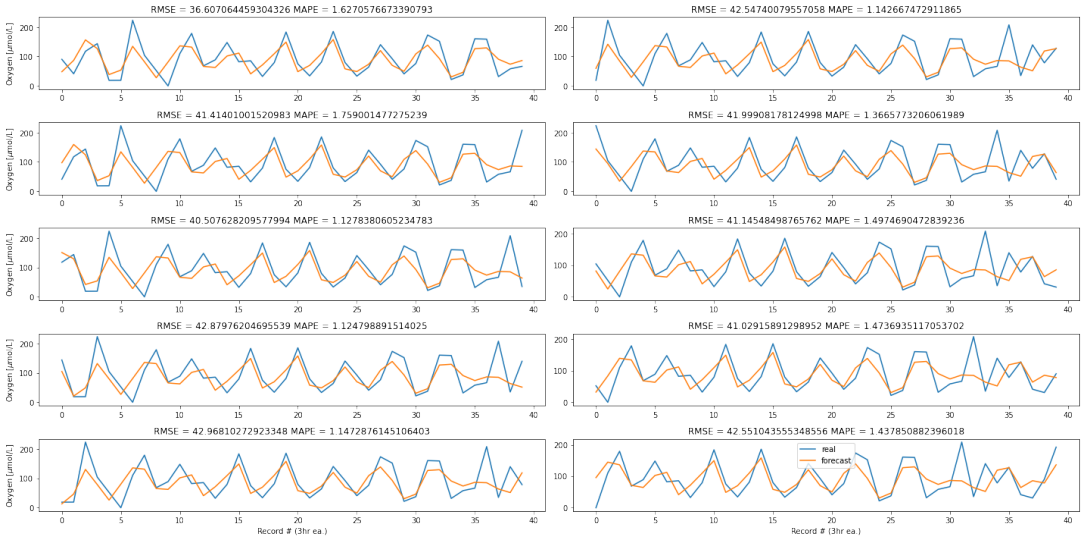


Figure 5.15. 5 days forecast of Dissolved Oxygen based on DO, WL, pre-forecasted T using 40 lags in 3 hour frequency data, training in 97% of dataset. RMSE = $41.36 \mu mol/L$, MAPE = 1.60%

In the case of the O5D2 model, which predicts five days ranges for a full year of data and is trained in 90% of the data, the results are even better and can be seen in Figure 5.16. In this case, we report the errors only as it would be impossible to know

the agreement between forecast and real data in this manner. The errors are RMSE = $43.87 \mu mol/L$, MAPE = 0.18%, and these are still much higher than the O1S1 and O1S2 models, but still relatively low for a model of this kind.

A marked seasonally dependent error increase can be appreciated from both of the full-year prediction models, mainly for the Fall season and into the first half of the winter season. In contrast, the general error is lower for the rest of the year, with minor fluctuations still. In Table 5.7, we summarize the models' performance and some of the characteristics between them

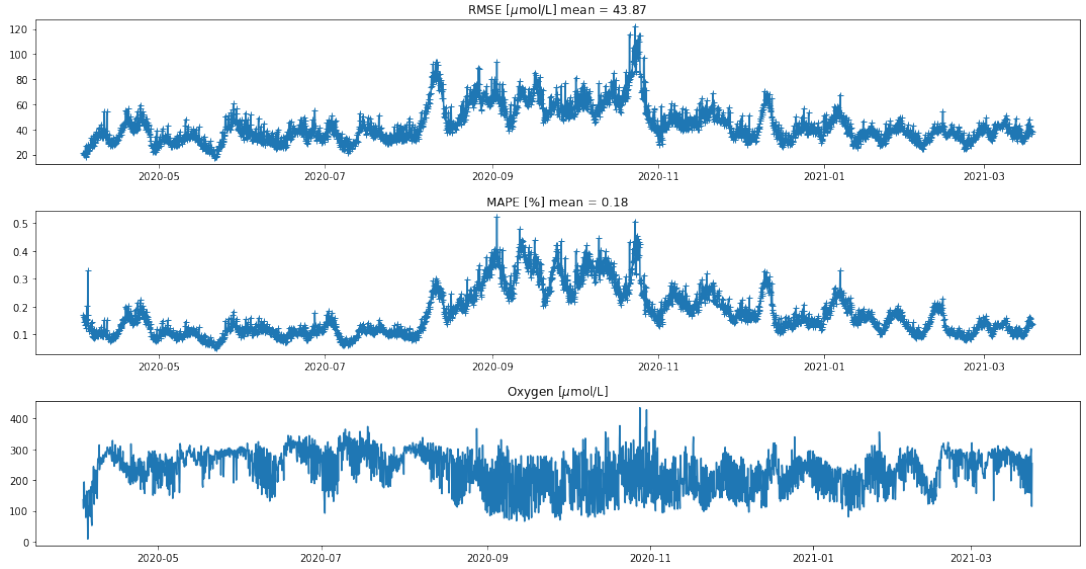


Figure 5.16. Error for one year of 5 days forecast of Dissolved Oxygen based on DO, WL, pre-forecasted T using 40 lags in 3 hour frequency data, training in 90% of dataset. RMSE = $43.87 \mu mol/L$, MAPE = 0.18%

Table 5.7. LTSM models performance results.

	RMSE [C]	MAPE [%]	Training %	Output #
T1S1	0.4782	0.0413	97	1
T1S2	0.9982	0.0617	90	1
T5D1	1.09	0.10	97	40
T5D2	0.95	0.06	90	40
	RMSE [µmol/L]	MAPE [%]	Training %	Output #
O1S1	0.1284	0.0005	97	1
O1S2	0.1465	0.0006	90	1
O5D1	41.36	1.60	97	40
O5D2	43.87	0.18	90	40

5.4 Discussion

Machine learning in environmental sciences is a nascent field showing great promise. The results of this chapter account for this, a highly complicated and important marker to trace such as dissolved oxygen being easily defined by ubiquitous counterparts, such as temperature, pressure, and salinity, is by itself a great application of machine learning, as before this paper, only instrumentation would yield reliable results of the dissolved oxygen in coastal regions. These instruments are usually a thermo-couple or an optic refraction reader of seawater, which in actuality measure DO indirectly from temperature and salinity readings. With these results, we have virtually eliminated the necessity of the specialized equipment to make this variable transformation, which translates into essential savings for the location with enough DO data associated with the training features. As we saw, the matter of how much data is enough data is an open question. Still, the results here presented show that it might be much fewer readings than anticipated. Those ensemble methods could pick up the highly nonlinear relations with only a few weeks of readings spaced during a year time. These applications also open the possibility of creating post-processing modules for CFD models such as the GCCOM, which handles the predictive features very well, to create a layer of information for DO or other similar variables, traditionally modeled using biogeochemical models that might not be accurate for the system. These results seem to bypass those models altogether while providing more substantial results. Additionally, cheaper alternatives to traditional DO reading instruments could be manufactured using pre-trained ML models, and these could be more reliable than the traditional ones.

The second part of the results, corresponding to the time series forecast, is novel results presented in this thesis and unprecedented for coastal ocean problems in its treatment and technique. The results are remarkably good and two-fold significant. First, they describe a method to predict time series of temperature (which arguably could be applied to other fundamental thermodynamic quantities) and a composite technique to feed other models' forecasts into more complicated variable estimates, such

as DO. The most important application of these results is a possible weather-like forecast of the temperature and DO conditions in a coastal location and possibly an early warning system for abrupt changes in these markers, which could be detrimental to fisheries and ecological reservoirs if not managed with enough warning. The other important application would be relating PH level, chlorophyll production, and different critical biogeochemical signatures to the current models for similar treatment and predictability. Second, the correlation of DO/PH is higher than that of temperature/DO here studied, which would make a PH time series forecast feasible in a more robust way than what we show here for DO.

It is important to remember that the results and model performance will be as good as the data used like any other AI or machine learning application. In this case, we were privileged to have extensive data to make these models successful. Still, sadly it is not possible to apply these techniques to other places without good data history.

The application of machine learning in environmental sciences, as well a computational fluid dynamics models and environmental management are showing performance that is hard to ignore and could revolutionize the way we attack traditional problems (instrumentation), as well as accelerating our understanding of the processes of the ocean while making use of large collections of data that already exist.

Finally, we can summarize the achievements of this chapter as follows:

- Implemented supervised ML models to measure DO in coastal waters indirectly
- These indirect measures are based on common thermodynamics markers, also commonly used in CFD.
- This enables a data-driven biogeochemical formulation for numerical modeling, in which pre-trained ML models can be included as part of a CFD simulation for the same conditions as the ML was trained in.
- Additionally, the models can be placed in a new range of instruments which could prove to be cheaper while maintaining accuracy.
- Weather-like (5-day) forecasts have been attained for temperature and DO in coastal waters.
- These forecasts would be beneficial for coastal economic and ecological management activities, being able to mitigate hypoxia or algae bloom events by forecasting them with enough time for meaningful action to occur.

CHAPTER 6

CONCLUSIONS, CONSIDERATIONS AND FUTURE WORK

This thesis represents an ambitious effort to stitch together two different fields, CFD and ML, in a wholesome manner. In contrast, each one has two equally complicated components to work with: Coastal ocean field-scale experiments, mimetic operator Navier-Stokes modeling, supervised general nonlinear machine learning regression, and times series forecasts workflows for biogeochemical markers in coastal seawater are each contributing in crucial ways to each of their fields but presented in this thesis and framework, the possible cross-field impacts of these findings can be appreciated better.

CFD/mimetic presents completely novel formulations tailored for N-S solvers within the mimetic framework, marrying a traditional oceanographic model approach with the mimetic technology, enabling unprecedented energy conservation and physics fidelity in a novel application of mimetic operators, in what it has become the most complex instance of mimetic models to date, and equally contributing to the insight or their working along with computational fluid dynamics models requirements.

The GCCOM has been given a new build from the ground up, basing itself in the mimetic operators' MOLE library for arbitrary order of accuracy, superior energy conservation, and improved validation. The mimetic operators worked exceptionally well for most of the proposed settings. Previously, unknown issues were identified and partially solved to make the mimetic operators work in the specific settings required by the GCCOM model. Future work will expand on these novel implementations, necessary for a higher order of accuracy on the advection step, and in turn, enable finer-detailed phenomena as internal wave beams to be resolved accurately. In any case, the results presented in the CFD portion of this work are remarkable in that they

represent a solid base for future researchers to work from, providing the framework to drive the GCCOM to a fully deployable state.

The ML component presented here is novel in a different way, using techniques that are still uncommon in the environmental sciences, and pushing them to new limits on expected accuracy and time series resolution and forecast. Nevertheless, the results seem to expand the horizon of what can be done in coastal ocean dynamics and possibly in oceanography and atmospheric sciences at large.

The ML models here presented can be applied in a range of different uses; one could be designing a pre-trained (or updated wireless) machine-learning-based DO sensor, that could be self-improving and would be at least 10 times cheaper than current DO sensor commercial models while yielding results at least as good, or even better. Another use would be using the trained models to provide DO layers in CFD simulations of the same sites where the data originated from, this would become an alternative to bio-geochemical models that, in some cases, such as DO in the coastal ocean, are not well understood and therefore yield low accuracy results, giving a data-driven approach here proposed an edge.

The time series forecast models implications are far-reaching, these results open the possibility of weather-like forecasts for biogeochemical markers in coastal regions, or early warning systems for fisheries to correct for sudden drops in oxygen supplies that might be about to happen, similarly for ecological reserves, mitigation strategies might be adopted beforehand to prevent coral bleaching events or massive die-outs, many times attributed to sudden hypoxia. This latter application still needs more research done since the dataset studied here has too few events of hypoxia to be valid in that case. However, the same tools and principles are applicable, and the framework should work similarly.

BIBLIOGRAPHY

[1] M. Abouali, *Curvilinear 2D Grid Poisson, matlab file exchange.*

[2] ——, *Investigating Castillo-Grone's Mimetic Difference Operators in Development of Geophysical Fluid Dynamics Models Implemented on GPGPUs by Mohammad Abouali*, PhD thesis, San Diego State University, 2014.

[3] M. Abouali and J. E. Castillo, *General Curvilinear Ocean Model (GCOM) Next Generation*, Tech. Rep. CSRCR2010-02, Computational Sciences Research Center, San Diego State University, 2010.

[4] M. Abouali and J. E. Castillo, *The Castillo-Grone's Mimetic Difference Operators in 2D and 3D fully Curvilinear Grids Case Study of Poisson's Equation*, tech. rep., 2012.

[5] ——, *Solving Poisson equation with Robin boundary condition on a curvilinear mesh using high order mimetic discretization methods*, Mathematics and Computers in Simulation, 139 (2014), pp. –.

[6] H. Ahmad, *Machine learning applications in oceanography*, Aquatic Reseach, 2 (2016), pp. 161–169.

[7] A. Arakawa, *Design of the UCLA general circulation model*, Tech. Rep. Tech. Report No. 7, Department of Meteorology, University of California, Los Angeles, 1972.

[8] ——, *Computational Design for Long-Term Numerical Integration of the Equations of Fluid Motion: Two-Dimensional Incompressible Flow. Part I*, J. Comput. Phys., 135 (1997), pp. 103–114.

[9] X. W. Bao, J. Yan, and W. X. Sun, *A three-dimensional tidal model in boundary-fitted curvilinear grids*, Estuarine, Coastal and Shelf Science, 50 (2000), pp. 775–788.

[10] L. Bautista, J. Villamizar, G. Calderón, J. C. Carrillo E., J. C. Rueda, and J. Castillo, *Mimetic finite difference methods for restoration of fundus images for automatic glaucoma detection*, in VipIMAGE 2019, J. M. R. S. Tavares and R. M. Natal Jorge, eds., Cham, 2019, Springer International Publishing, pp. 104–113.

[11] C. Bazan, M. Abouali, J. Castillo, and P. Blomgren, *Mimetic finite difference methods in image processing*, Computational and Applied Mathematics, 30 (2011), pp. 701–720.

[12] B. Béjaoui, E. Ottaviani, E. Barelli, B. Ziadi, A. Dhib, M. Lavoie, C. Gianluca, S. Turki, C. Solidoro, and L. Aleya, *Machine learning predictions of trophic status indicators and plankton dynamic in coastal lagoons*, Ecological Indicators, 95 (2018), pp. 765–774.

[13] J. Blanco, O. Rojas, C. Chacón, J. M. Guevara-Jordan, and J. Castill, *Tensor formulation of 3-d mimetic finite differences and applications*

to elliptic problems, Electronic Transactions on Numerical Analysis, 45 (2016), pp. 457–475.

[14] A. Boada Velazco, *High Order Mimetic Finite Differences on Non-Trivial Problems*, PhD thesis, 2021.

[15] A. Boada Velazco, J. Corbino, and J. Castillo, *High order mimetic difference simulation of unsaturated flow using Richards equation*, Mathematics in Applied Sciences and Engineering, 1 (2020), pp. 401–409.

[16] A. Boada Velazco, C. Paolini, and J. E. Castillo, *High-order mimetic finite differences for anisotropic elliptic equations*, Computers and Fluids, 213 (2020), p. 104746.

[17] B. E. Boser, I. M. Guyon, and V. N. Vapnik, *A Training Algorithm Margin for Optimal Classifiers*, in COLT '92: Proceedings of the fifth annual workshop on Computational learning theory, 1992, pp. 144–152.

[18] J. Boussinesq, *Théorie de l'écoulement tourbillonnant et tumultueux des liquides dans les lits rectilignes a grande section*, Paris, Gauthier-Villars et fils, (1897), p. 94.

[19] L. Breiman, *Random Forests*, Random Forests, 45 (2001), pp. 5–32.

[20] D. L. Brown, R. Cortez, and M. L. Minion, *Accurate Projection Methods for the Incompressible Navier-Stokes Equations*, Journal of Computational Physics, 168 (2001), pp. 464–499.

[21] J. Brzenski, M. Valera, and J. E. Castillo, *Coupling GCCOM, a curvilinear ocean model rigid lid simulation with SWASH for analysis of free surface conditions*, OCEANS 2019 MTS/IEEE Seattle, OCEANS 2019, (2019).

[22] M. H. Carr, C. B. Woodson, O. M. Cheriton, D. Malone, M. A. McManus, and P. T. Raimondi, *Knowledge through partnerships: Integrating marine protected area monitoring and ocean observing systems*, Frontiers in Ecology and the Environment, 9 (2011), pp. 342–350.

[23] J. E. Castillo and R. D. R. Grone, *A Matrix Analysis Approach to Higher-Order Approximations for Divergence and Gradients Satisfying a Global Conservation Law*, SIAM Journal on Matrix Analysis and Applications, 25 (2003), pp. 128–142.

[24] J. E. Castillo and G. F. Miranda, *Mimetic Discretization Methods*, Chapman and Hall/CRC, 2013.

[25] V. Casulli, *A semi-implicit finite difference method for non-hydrostatic, free-surface flows*, International Journal for Numerical Methods in Fluids, 30 (1999), pp. 425–440.

[26] T. Cavazos, A. C. Comrie, and D. M. Liverman, *Intraseasonal variability associated with wet monsoons in southeast Arizona*, Journal of Climate, 15 (2002), pp. 2477–2490.

[27] V. K. Chalamalla, E. Santilli, A. Scotti, M. Jalali, and S. Sarkar, *SOMAR-LES: A framework for multi-scale modeling of turbulent stratified oceanic flows*, Ocean Modelling, 120 (2017), pp. 101–119.

[28] S. Chen, C. Hu, B. B. Barnes, R. Wanninkhof, W. J. Cai, L. Barbero, and D. Pierrot, *A machine learning approach to estimate surface ocean pCO2 from satellite measurements*, Remote Sensing of Environment, 228 (2019), pp. 203–226.

[29] P. Choboter, *Equations and Notes for GCCOM Details on Spline and Extrapolation to Ghost Points*, 2021.

[30] J. Corbino, *MOLE: Mimetic Operators Library Enhanced*, 2019.

[31] J. Corbino and J. E. Castillo, *High-order mimetic finite-difference operators satisfying the extended Gauss divergence theorem*, Journal of Computational and Applied Mathematics, 364 (2020), p. 112326.

[32] L. J. Córdova, O. Rojas, B. Otero, and J. Castillo, *Compact finite difference modeling of 2-D acoustic wave propagation*, Journal of Computational and Applied Mathematics, 295 (2016), pp. 83–91.

[33] C. Cortes and V. Vapnik, *Support-Vector Networks*, Machine Learning, 20 (1995), pp. 273–297.

[34] D. T. Cox, P. Tissot, and P. Michaud, *Water level observations and short-term predictions including meteorological events for entrance of Galveston Bay, Texas*, Journal of Waterway, Port, Coastal and Ocean Engineering, 128 (2002), pp. 21–29.

[35] B. Cushman-Roisin and J.-M. Beckers, *Introduction to Geophysical Fluid Dynamics. Physical and Numerical Aspects*, vol. 101, Academic Press, 2010.

[36] J. De la Puente, M. Ferrer, M. Hanzich, J. E. Castillo, and J. M. Cela, *Mimetic seismic wave modeling including topography on deformed staggered grids*, Geophysics, 79 (2014), pp. T125–T141.

[37] S. Dzeroski, *Applications of symbolic machine learning to ecological modelling*, Ecological Modelling, (2001), pp. 263–273.

[38] O. B. Fringer, *Towards Nonhydrostatic Ocean Modeling with Large-eddy Simulation*, in Natl. Res. Counc. Oceanogr. 2025 Proc. a Work., 2009, pp. 81–83.

[39] O. B. Fringer, J. McWilliams, R. L. Street, J. C. McWillimas, and R. L. Street, *A New Hybrid Model for Coastal Simulations*, Oceanography, 19 (2006), pp. 64–77.

[40] M. Garcia, P. F. Choboter, R. K. Walter, and J. E. Castillo, *Validation of the nonhydrostatic General Curvilinear Coastal Ocean Model (GCCOM) for stratified flows.*, Journal of Computational Science, 30 (2019), pp. 143–156.

[41] M. Garcia, T. Hoar, M. Thomas, B. Bailey, and J. E. Castillo, *Interfacing an ensemble Data Assimilation system with a 3D nonhydrostatic Coastal Ocean Model, an OSSE experiment*, in Oceans 2016 MTS/IEEE Monterey, sep 2016, pp. 1–11.

[42] B. Gokaraju, S. S. Durbha, R. L. King, and S. Member, *A Machine Learning Based Spatio-Temporal Data Mining Approach for Detection of Harmful Algal Blooms in the Gulf of Mexico*, Journal of Selected Topics in Applied Earth

Observations and Remote Sensing, 4 (2011).

[43] E. B. Goldstein, G. Coco, and N. G. Plant, *A review of machine learning applications to coastal sediment transport and morphodynamics*, Earth-Science Reviews, 194 (2019), pp. 97–108.

[44] J. M. Guevara-Jordan, S. Rojas, M. Freites-Villegas, and J. E. Castillo, *Convergence of a mimetic finite difference method for static diffusion equation*, Advances in Difference Equations, 2007 (2007), pp. 1–12.

[45] J. Gum, M. Valera, B. Larson, K. Jackson, S. Sherk, and A. Et, *"OMLET" - Ocean Machine LEarning Toolkit*, in OceanHackWeek 2018, 2018.

[46] C. Härtel, E. Meiburg, and F. Necker, *Analysis and direct numerical simulation of the flow at a gravity-current head. Part 1. Flow topology and front speed for slip and no-slip boundaries*, Journal of Fluid Mechanics, 418 (2000), pp. 189–212.

[47] K. a. Hoffmann and S. T. Chiang, *Computational Fluid Dynamics Vol.I - Hoffmann.pdf*, vol. 126, Engineering Education System, 2000.

[48] W. Hsieh, *Machine Learning in the Enviromental Sciences*, Cambridge University Press, 2009.

[49] J. Hyman, J. Morel, M. Shashkov, and S. Steinberg, *Mimetic finite difference methods for diffusion equations*, Computational Geosciences, 6 (2002), pp. 333–352.

[50] C. Jamet, S. Thiria, C. Moulin, and M. Crepon, *Use of a neurovariational inversion for retrieving oceanic and atmospheric constituents from ocean colar imagery: A feasibility study*, Journal of Atmospheric and Oceanic Technology, 22 (2005), pp. 460–475.

[51] X. Ji, X. Shang, R. A. Dahlgren, and M. Zhang, *Prediction of dissolved oxygen concentration in hypoxic river systems using support vector machine: a case study of Wen-Rui Tang River, China*, Environmental Science and Pollution Research, 24 (2017), pp. 16062–16076.

[52] P. Jimeno-Sáez, J. Senent-Aparicio, J. M. Cecilia, and J. Pérez-Sánchez, *Using machine-learning algorithms for eutrophication modeling: Case study of mar menor lagoon (spain)*, International Journal of Environmental Research and Public Health, 17 (2020).

[53] D. Kamykowski and S.-j. Zentara, *Predicting plant nutrient concentrations from temperature and sigma- £ in the upper kilometer of the world ocean*, Deep-Sea Research, 33 (1986), pp. 89–105.

[54] T. Kawamura, H. Takami, and K. Kuwahara, *Computation of high Reynolds number flow around a circular cylinder with surface roughness*, Fluid Dyn. Res., 1 (1986), pp. 145–162.

[55] L. E. Keiner and X. H. Yan, *A neural network model for estimating sea surface chlorophyll and sediments from thematic mapper imagery*, Remote Sensing of Environment, 66 (1998), pp. 153–165.

[56] C. A. Kennedy, M. H. Carpenter, and R. M. Lewis, *Low-storage, explicit*

Runge-Kutta schemes for the compressible Navier-Stokes equations, Applied Numerical Mathematics, 35 (2000), pp. 177–219.

[57] D. I. Ketcheson, *Highly efficient strong stability-preserving Runge-Kutta methods with low-storage implementations*, SIAM Journal on Scientific Computing, 30 (2007), pp. 2113–2136.

[58] H.-O. KREISS and G. SCHERER, *Finite Element and Finite Difference Methods for Hyperbolic Partial Differential Equations*, in Mathematical Aspects of Finite Elements in Partial Differential Equations, C. de Boor, ed., Academic Press, 1974, pp. 195–212.

[59] Z. Lai, C. Chen, G. W. Cowles, and R. C. Beardsley, *A nonhydrostatic version of FVCOM: 1. Validation experiments*, Journal of Geophysical Research: Oceans, 115 (2010), pp. 1–23.

[60] F. Lemarié, J. Kurian, A. F. Shchepetkin, M. Jeroen Molemaker, F. Colas, and J. C. McWilliams, *Are there inescapable issues prohibiting the use of terrain-following coordinates in climate models?*, Ocean Modelling, 42 (2012), pp. 57–79.

[61] Z. Liu, L. Lin, L. Xie, and H. Gao, *Partially implicit finite difference scheme for calculating dynamic pressure in a terrain-following coordinate non-hydrostatic ocean model*, Ocean Modelling, 106 (2016), pp. 44–57.

[62] S. G. Llewellyn Smith and W. R. Young, *Conversion of the barotropic tide*, Journal of Physical Oceanography, 32 (2002), pp. 1554–1566.

[63] W. S. McCullough and W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics, 5 (1943).

[64] N. Nangia, B. E. Griffith, N. A. Patankar, and A. P. S. Bhalla, *A robust incompressible Navier-Stokes solver for high density ratio multiphase flows*, Journal of Computational Physics, 390 (2019), pp. 548–594.

[65] F. Necker, C. Härtel, L. Kleiser, and E. Meiburg, *High-resolution simulations of particle-driven gravity currents*, International Journal of Multiphase Flow, 28 (2002), pp. 279–300.

[66] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, 12 (2011), pp. 2825–2830.

[67] K. Qu, H. S. Tang, A. Agrawal, C. B. Jiang, and B. Deng, *Evaluation of SIFOM-FVCOM system for high-fidelity simulation of small-scale coastal ocean flows*, Journal of Hydrodynamics, Ser. B, 28 (2016), pp. 994–1002.

[68] D. A. Randall, R. A. Wood, S. Bony, R. Colman, T. Fichefet, J. Fyfe, V. Kattsov, A. Pitman, J. Shukla, J. Srinivasan, R. J. Stouffer, A. Sumi, and K. E. Taylor, *Climate Models and Their Evaluation*, in Contrib. Work. Gr. I to Fourth Assess. Rep. Intergov. Panel Clim. Chang., 2007, ch. 8.

[69] O. Rojas, S. Day, J. Castillo, and L. A. Dalguer, *Modelling of rupture propagation using high-order mimetic finite differences*, Geophysical Journal

International, 172 (2008), pp. 631–650.

[70] F. Rosenblatt, *The perceptron: A probabilistic model for information storage and organization in the brain*, Psychological Review, 65 (1958), pp. 386–408.

[71] A. C. Ross and C. A. Stock, *An assessment of the predictability of column minimum dissolved oxygen concentrations in Chesapeake Bay using a machine learning model*, Estuarine, Coastal and Shelf Science, 221 (2019), pp. 53–65.

[72] E. Santilli and A. Scotti, *The stratified ocean model with adaptive refinement (SOMAR)*, Journal of Computational Physics, 291 (2015), pp. 60–81.

[73] H. Schiller and R. Doerffer, *Improved determination of coastal water constituent concentrations from MERIS data*, IEEE Transactions on Geoscience and Remote Sensing, 43 (2005), pp. 1585–1591.

[74] A. F. Shchepetkin and J. C. McWilliams, *A method for computing horizontal pressure-gradient force in an oceanic model with a nonaligned vertical coordinate*, J. Geophys. Res., 108 (2003), pp. n/a—-n/a.

[75] ——, *The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model*, Ocean Model., 9 (2005), pp. 347–404.

[76] J. E. Sherwood, F. Stagnitti, M. J. Kokkinn, and W. D. Williams, *Dissolved oxygen concentrations in hypersaline waters*, Limnology and Oceanography, 36 (1991), pp. 235–250.

[77] C.-W. Shu and S. Osher, *Efficient Implementation of essentially non-oscillatory shock capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471.

[78] J. Smagorinsky, *General Circulation Experiments with the Primitive Equations*, Monthly Weather Review, 91 (1963), p. 99.

[79] Y. Song and D. B. Haidvogel, *A semi-implicit primitive equation ocean circulation model using a generalized topography-following coordinate system*, J. Comput. Phys., 115 (1994), pp. 228–244.

[80] M. Thomas and J. Castillo, *A Cyberinfrastructure-Based Computational Environment for Unified Curvilinear Ocean Atmospheric Model (UCOAM)*, in 4th Int. Congr. Comput. Eng. Sci., 2013.

[81] H. L. Tolman, V. M. Krasnopolsky, and D. V. Chalikov, *Neural network approximations for nonlinear interactions in wind wave spectra: Direct mapping for wind seas in deep water*, Ocean Modelling, 8 (2005), pp. 253–278.

[82] C. Torres and J. E. Castillo, *Modeling Ocean Circulation in Boundary-fitted Coordinates: The GCOM Project.*, in III Pan-American Adv. Stud. Inst. Comput. Sci. Eng., 2006, pp. 1–37.

[83] C. R. Torres and J. E. Castillo, *Stratified Rotating Flow Over Complex Terrain*, Applied Numerical Mathematics, 47 (2003), pp. 531–541.

[84] M. Valera, M. Garcia, M. Walter, R. Choboter, and J. E. Castillo, *Modeling nearshore internal bores and waves in Monterey bay using the General Curvilinear Coastal Ocean Dynamics Model , GCCOM*, in 2018 Joint AMS-SIAM Meeting, Minisymposium on Mimetic Multiphase Subsurface and Oceanic

Transport., 2018.

[85] M. Valera, Z. Guo, P. Kelly, S. Matz, V. A. Cantu, A. G. Percus, J. D. Hyman, G. Srinivasan, and H. S. Viswanathan, *Machine learning for graph-based representations of three-dimensional discrete fracture networks*, Computational Geosciences, 22 (2018), pp. 695–710.

[86] M. Valera, N. Patel, and J. Castillo, *PETSc-based Parallelization of the fully 3D-curvilinear Non-hydrostatic Coastal Ocean Dynamics Model, GCCOM*, (2017).

[87] M. Valera, M. P. Thomas, M. Garcia, and J. E. Castillo, *Parallel implementation of a PETSc-based framework for the General Curvilinear Coastal Ocean Model*, Journal of Marine Science and Engineering, 7 (2019).

[88] M. Valera, R. K. Walter, B. A. Bailey, and J. E. Castillo, *Machine Learning Based Predictions of Dissolved Oxygen in a Small Coastal Embayment*, Journal of Marine Science and Engineering, 8 (2020), p. 16.

[89] E. A. Virtanen, A. Norkko, A. Nyström Sandman, and M. Viitasalo, *Identifying areas prone to coastal hypoxia - The role of topography*, Biogeosciences, 16 (2019), pp. 3183–3195.

[90] S. Vitousek and O. B. Fringer, *Physical vs. numerical dispersion in nonhydrostatic ocean modeling*, Ocean Model., 40 (2011), pp. 72–86.

[91] S. Vitousek and O. B. Fringer, *A nonhydrostatic, isopycnal-coordinate ocean model for internal waves*, Ocean Modelling, 83 (2014), pp. 118–144.

[92] R. K. Walter, C. B. Woodson, R. S. Arthur, O. B. Fringer, and S. G. Monismith, *Nearshore internal bores and turbulent mixing in southern Monterey Bay*, J. Geophys. Res., 117 (2012), p. C07017.

[93] R. K. Walter, C. B. Woodson, P. R. Leary, and S. G. Monismith, *Connecting wind-driven upwelling and offshore stratification to nearshore internal bores and oxygen variability*, Journal of Geophysical Research: Oceans, 119 (2014), pp. 3517–3534.

[94] T. Weber, N. A. Wiseman, and A. Kock, *Global ocean methane emissions dominated by shallow coastal waters*, Nature Communications, 10 (2019), pp. 1–10.

[95] R. F. Weiss, *The solubility of nitrogen, oxygen and argon in water and seawater*, in Deep Sea Research and Oceanographic Abstracts, vol. 17, Elsevier, 1970, pp. 721–735.

[96] J. Williamson, *Low-storage Runge-Kutta schemes*, J. Comput. Phys., 35 (1980), pp. 48–56.

[97] C. B. Woodson, *The Fate and Impact of Internal Waves in Nearshore Ecosystems*, Annual Review of Marine Science, 10 (2018), pp. 421–441.

[98] Z. M. Yaseen, O. Jaafar, R. C. Deo, O. Kisi, J. Adamowski, J. Quilty, and A. El-Shafie, *Stream-flow forecasting using extreme learning machines: A case study in a semi-arid region in Iraq*, Journal of Hydrology, 542 (2016), pp. 603–614.

[99] X. Yu, J. Shen, and J. Du, *A Machine-Learning-Based Model for Water*

Quality in Coastal Waters, Taking Dissolved Oxygen and Hypoxia in Chesapeake Bay as an Example, Water Resources Research, 56 (2020), p. e2020WR027227.