Scientific Machine Learning for Dynamical Systems: Theory and Applications to Fluid Flow and Ocean Ecosystem Modeling

by

Abhinav Gupta

B.Tech.-M.Tech., Indian Institute of Technology, Kanpur (2016)

Submitted to the Department of Mechanical Engineering and the
Center for Computational Science and Engineering
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Mechanical Engineering and Computation
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
September 2022

Author .....
Department of Mechanical Engineering and the Center for
Computational Science and Engineering
Friday 29th July, 2022

Certified by .....
Pierre F.J. Lermusiaux
Professor, Department of Mechanical Engineering
Thesis Supervisor

Accepted by .....
Nicolas Hadjiconstantinou
Professor, Department of Mechanical Engineering
Chairman, Department Committee on Graduate Theses

Accepted by .....
Youssef Marzouk
Professor, Department of Aeronautics and Astronautics
Co-Director, Center for Computational Science and Engineering

Scientific Machine Learning for Dynamical Systems: Theory and Applications to Fluid Flow and Ocean Ecosystem Modeling

by

Abhinav Gupta

Submitted to the Department of Mechanical Engineering and the Center for Computational Science and Engineering
on Friday 29th July, 2022, in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Mechanical Engineering and Computation

Abstract

Complex dynamical models are used for prediction in many domains, and are useful to mitigate many of the grand challenges being faced by humanity, such as climate change, food security, and sustainability. However, because of computational costs, complexity of real-world phenomena, and limited understanding of the underlying processes involved, models are invariably approximate. The missing dynamics can manifest in the form of unresolved scales, inexact processes, or omitted variables; as the neglected and unresolved terms become important, the utility of model predictions diminishes. To address these challenges, we develop and apply novel scientific machine learning methods to learn unknown and discover missing dynamics in models of dynamical systems.

In our Bayesian approach, we develop an innovative stochastic partial differential equation (PDE) - based model learning theory and framework for high-dimensional coupled biogeochemical-physical models. The framework only uses sparse observations to learn rigorously within and outside of the model space as well as in that of the states and parameters. It employs Dynamically Orthogonal (DO) differential equations for adaptive reduced-order stochastic evolution, and the Gaussian Mixture Model-DO (GMM-DO) filter for simultaneous nonlinear inference in the augmented space of state variables, parameters, and model equations. A first novelty is the Bayesian learning among compatible and embedded candidate models enabled by parameter estimation with special stochastic parameters. A second is the principled Bayesian discovery of new model functions empowered by stochastic piecewise polynomial approximation theory. Our new methodology not only seamlessly and rigorously discriminates between existing models, but also extrapolates out of the space of models to discover newer ones. In all cases, the results are generalizable and interpretable, and associated with probability distributions for all learned quantities. To showcase and quantify the learning performance, we complete both identical-twin

and real-world data experiments in a multidisciplinary setting, for both filtering forward and smoothing backward in time. Motivated by active coastal ecosystems and fisheries, our identical-twin experiments consist of lower-trophic-level marine ecosystem and fish models in a two-dimensional idealized domain with flow past a seamount representing upwelling due to a sill or strait. Experiments have varying levels of complexities due to different learning objectives and flow and ecosystem dynamics. We find that even when the advection is chaotic or stochastic from uncertain nonhydrostatic variable-density Boussinesq flows, our framework successfully discriminates among existing ecosystem candidate models and discovers new ones in the absence of prior knowledge, along with simultaneous state and parameter estimation. Our framework demonstrates interdisciplinary learning and crucially provides probability distributions for each learned quantity including the learned model functions. In the real-world data experiments, we configure a one-dimensional coupled physical-biological-carbonate model to simulate the state conditions encountered by a research cruise in the Gulf of Maine region in August, 2012. Using the observed ocean acidification data, we learn and discover a salinity based forcing term for the total alkalinity (TA) equation to account for changes in TA due to advection of water masses of different salinity caused by precipitation, riverine input, and other oceanographic processes. Simultaneously, we also estimate the multidisciplinary states and an uncertain parameter. Additionally, we develop new theory and techniques to improve uncertainty quantification using the DO methodology in multidisciplinary settings, so as to accurately handle stochastic boundary conditions, complex geometries, and the advection terms, and to augment the DO subspace as and when needed to capture the effects of the truncated modes accurately. Further, we discuss mutual-information-based observation planning to determine what, when, and where to measure to best achieve our learning objectives in resource-constrained environments.

Next, motivated by the presence of inherent delays in real-world systems and the Mori-Zwanzig formulation, we develop a novel delay-differential-equations-based deep learning framework to learn time-delayed closure parameterizations for missing dynamics. We find that our neural closure models increase the long-term predictive capabilities of existing models, and require smaller networks when using non-Markovian over Markovian closures. They efficiently represent truncated modes in reduced-order-models, capture effects of subgrid-scale processes, and augment the simplification of complex physical-biogeochemical models. To empower our neural closure models framework with generalizability and interpretability, we further develop neural partial delay differential equations theory that augments low-fidelity models in their original PDE forms with both Markovian and non-Markovian closure terms parameterized with neural networks (NNs). For the first time, the melding of low-fidelity model and NNs with time-delays in the continuous spatiotemporal space followed by numerical discretization automatically provides interpretability and allows for generalizability to computational grid resolution, boundary conditions, initial conditions, and problem specific parameters. We derive the adjoint equations in the continuous form, thus, allowing implementation of our new methods across differentiable and non-differentiable computational physics codes, different machine learning frameworks, and also non-uniformly-spaced spatiotemporal training data. We also show

that there exists an optimal amount of past information to incorporate, and provide methodology to learn it from data during the training process. Computational advantages associated with our frameworks are analyzed and discussed. Applications of our new Bayesian learning and neural closure modeling are not limited to the shown fluid and ocean experiments, but can be extended to other fields such as control theory, robotics, pharmacokinetic-pharmacodynamics, chemistry, economics, and biological regulatory systems.

Thesis Supervisor: Pierre F.J. Lermusiaux
Title: Professor, Department of Mechanical Engineering

Acknowledgments

Oṃ Bhūr Bhuvaḥ Svaḥ
Tat Saviturvareñyaṃ
Bhargo Devasya Dhīmahi
Dhiyo Yonaḥ Prachodayāt

Gayatri Mantra
Rig Veda 3.62.10

I would like to, first of all, thank my advisor, Prof. Pierre F.J. Lermusiaux. We have come a long way, starting with our first ever correspondence on October 1st, 2013, when as an undergraduate student I wrote an email expressing my desire to conduct research with him in the coming summer. Thank you for replying to me and initiating a series of events that lead to the eventual completion of this PhD. The two summer internships at MSEAS greatly helped me shape as a researcher and prepare me for graduate school. Over the years, there have been so many ups and downs, however, you have always supported and shown confidence in me. Also, thank you for financially supporting me throughout the years from various research grants, and pushing me to apply for competitive fellowships, and supporting my applications. Further, I really appreciate the countless hours you have spent polishing my various slide decks, papers, and thesis text. I would like to acknowledge the following funding sources which supported me during my PhD: • MIT-Tata Center for Technology & Design Fellowship 2018-20 (Dr. Robert Stoner, Dr. Angeliki Diane Rigos, and Dr. Chintan Vaishnav for their guidance and help) • MathWorks Mechanical Engineering Fellowship 2020-21 • Office of Naval Research N00014-19-1-2693 (IN-BDA) and N00014-20-1-2023 (MURI ML-SCOPE) • Sea Grant and NOAA NA18OAR4170105 (BIOMAPS). Additionally, I would also like to thank Indo-US Science and Technology Forum (IUSSTF) for funding my summer internship through the S.N. Bose Scholars Program 2014. Pierre, thank you for treating me as a friend and younger brother. I wish you the best of luck in your future endeavors and look forward to staying in touch forever.

I would also like to thank my committee members, Prof. Harry Asada, Prof. Avijit Gangopadhyay, and Prof. Tamara Broderick for all their inputs and help during the numerous committee meetings and the defense. Thank you for asking important questions and for your constant encouragement. It has also been a pleasure to attend 2.160 taught by Prof. Asada and 6.435 taught by Prof. Broderick. Both the courses were a great learning experience for me.

Next, I would like to thank the senior members of the MSEAS group, Dr. Patrick Haley and Dr. Chris Mirabito. It has been a great pleasure working with both of you, and I am always grateful for all the help over the years. Thank you for making all the countless hours working for sea-exercises bearable with your companionship, and coming to check-in in the morning after all-nighters. Pat, I really enjoy your puns. Chris, don't forget to keep yourself up to date with all the Nantucket bottle cap facts. You have to keep on beating both Pat and Pierre!

I am also thankful to the administrative staff who have offered me great help over the years, Leslie Regan (MechE), Saana J McDaniel (MechE), Una M Sheehan (MechE), Kate Nelson (CCSE), Marcia Munger (MSEAS), and Lisa Mayer (MSEAS). They have solutions to all our problems and hats off to their commitment to helping students selflessly.

Next comes the amazing labmates who made my grad school experience awesome and completely different from what I expected. I will first start with the MSEAS students who I explicitly only overlapped with during my summer internships, Sydney Saroka, John Aoussou, Jen Landry, and Tapovan Lolla. Thank you, Sydney, for sharing all the 2.29 FV framework struggles and for keeping bumping into me in Sid-Pac. John, I so clearly remember your love for Japan, and I am happy that you pursued your dream and moved there after graduation. Tapovan, thank you for all the guidance, letting me stay in your apartment, and including me in the badminton group. Followed are the people who overlapped during both internship and grad school, Jing Lin and Deepak Subramani. Jing, any amount of words will not be able to do justice for all the respect and gratitude I want to express. You

are the most intelligent human I have ever encountered and was I lucky enough to work with you. Thank you for so patiently answering all my questions over the years, for some really nice chats, and for all the work we did together on Bayesian learning. I really cherish each and every conversation I have shared with you, and also the only all-nighter we ever did together. Deepak, thank you for all your help and advice over the years, reviewing my statement of purpose, for inviting me to give seminars at IISc, and for countless other things. You are a great orator when it comes to selling your research, keep it up. Finally comes the long list of labmates who I only met after starting grad school, Arkopal Dutt, Chinmay Kulkarni, Johnathan Vo, Michael Humara, Jacob Heuss, Kyprianos Gkirgkis, Aditya Ghodgaonkar, Zach Duguid, Manmeet Singh Bhabra, Tony Ryu, Aaron Charous, Manan Doshi, Corbin Foucart, Wael H. Ali, Aman Jalan, Clara Dahill, Alonso V. Rodriguez, Anantha Narayanan Suresh Babu, and Aditya K. Saravanakumar. Arkopal and Chinmay, thank you all the help during the years and especially during the sea-exercises and various presentations. JVo, our overlap as labmates was hardly 6 months, however, I feel like I have spent a lifetime with you. I will never forget the India trip and thank you for keeping on visiting us over the years. Thank you for numerous meals where you have footed the bill. Mike, I will always be indebted for teaching me to drive, thank you for risking your life and also hosting us at your house on various occasions. Do send the submarine to pick me up when the apocalypse comes. Jacob, thank you for reminding me that there is no place for veggies in American cuisine. Akis, thank you for all the Greece travel advice, and I definitely look forward to visiting. Aditya and Zach, though your time in MSEAS was cut short, however, it was a lot of fun to have you around. Manny, I am still waiting for my postcard! There is no doubt that you write the most beautiful codes, and they read like a poem. Thank you for being a wonderful friend. Tony and Aaron, thank you for always being there and smiling. It is so much fun hanging out with you guys. Manan, thank you for always offering a helping hand and being a good sport. Corbin, we share a special bond of love and thank you for marrying me off. Wael, thank you for all the Baklava over the years, and for bringing calmness to our lab. JVo, Manny, Tony, Aaron, Manan, Corbin, and

Wael, any amount of text in this acknowledgment cannot do justice to the friendship we share, i.e. why I am keeping it short. Aman, Clara, Alonso, Anantha, and Aditya, it was my pleasure knowing and working with you guys in the final two years. Over the years, I also had the chance to work with and guide a number of UROPs, RSI kids, and other interns in our lab. The list is long, but some notable mentions are Ali Daher, Winston Fu, Jade, Stefano, Flavia, Chance Emerson, and Jani. *I would like to thank all the current members of the MSEAS group for their contribution towards preparation of this thesis document, polishing my defense slides, and helping me nail the delivery.*

Over the years, I have also made some really strong friendships with people in the department, through the Tata center fellowship program, Sid-Pac, and social groups like Sangam, GAME, etc. Saviz, Sid, Kriti, Marc, Sujay, Nithin, Shashank, Robin, Prashant, Riddhi, Ranu, Esha, Madhav, Emma, Qin, Hoa, Sanjana, and Pushpa, you all have played a very crucial role by providing me with a community outside my lab. You guys have consistently spent time with me and helped me cope with so many things, especially covid. I also had a number of apartment-mates in Sid-Pac, which made my stay in Apt. 333 very memorable. Afshine, Corban, and Palash thank you for continuing our friendship even after you left. Anzo, you are the only one who has outlived me in that apartment, thank you for making my last year pleasant. Swathi, thank you for being my badminton partner for the last one-and-a-half years. I would like to also mention Tanmayee, whom I met during the summer of 2015 for many fun trips, staying in touch over all these years, and also for all the nice postcards.

Going to my pre-MIT life, I would also like to thank all my friends, teachers, and advisors from IIT Kanpur, Bharti Public School, and IIT-JEE coaching. All my wingies (Kanhaiya, Aman Sunderka, Nikhil, Vikash, Amandeep, Sanyam, Dhrupal, Rachit, Sarthak, Kishlay, Saket, Pranjul, Suraj, Hari Shankar, Sandip, Aman Gupta, and Neelesh), Prof. Arun K. Saha, Gaurav Saxena, Basheer, Sachidananda Behera, Ankit Shrivastava, Ayush Jain, Jayant, Eeshit, Deepansh, Abhishek Gupta, Shelvi, Tawleen, Farman Menon, Kalyani, Ayush Singh, Piyush, Aman Maheshwari, Nibesh,

Puneet, Shahdara neighbors, and soo many more people have played such an important role in making me who I am today and contributing to my success. Thank you all for your unwavering confidence in me, love, friendship, and guidance.

Finally, I would like to thank my family for their love, support, and sacrifice, Bua ji, Tau ji, Tai ji, Ratna behen, Ajay jiju, Deepa behen, Sudershan jiju, Amit bhai, Anu bhabhi, Papa, Mummy, Anu behen (and your in-laws), Abhishek jiju, Guddu behen (and your in-laws), Manan jiju, Abhay, Shubham, Gauri, Sunny, Dhruv, Aanya, Hiyansh, Ravinder mama, and Ankit bhai. I wish Bua and Mummy were here to see the completion of my PhD. I will always be indebted to all my family members. I would also like to thank my wife Ria for her love and support, and for making life fun. Our relationship started in my 3rd year of PhD, blossomed in the final years, and graduated to being life partners simultaneously as I finish this PhD.

Thank you, god, for maintaining your grace and giving me all the opportunities[1].

[1]If in case you are reading this acknowledgment expecting to see your name and don't find it, I am really sorry to disappoint you. I would like to thank you whole-heartily for your contributions to my growth, and please forgive me for the unintentional omission.

Dedicated to my family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Background

Mathematical modeling provides humans the facility to use the language of mathematics to examine, understand, explain, and predict real-world phenomena. Mathematical models are omnipresent in every discipline, ranging from natural sciences, engineering, as well as in the social sciences, and are used for a variety of research and societal needs, including applications in energy, food, climate, and sustainability.

In this thesis, our main focus will be on mathematical models used to describe dynamical systems. Dynamical systems are phenomena whose state evolve in time, such as chemical reactions, biological processes, fluid flows, etc. They are often modeled as differential equations, and the simplest ones formulated using ordinary differential equations (ODEs) of the form,

$$\frac{d\boldsymbol{u}(t)}{dt} = f(\boldsymbol{u}(t), \theta), \qquad t \in [0, \infty) \ , \tag{1.1}$$

where $\boldsymbol{u}$ is the state vector containing the variables being modeled, $f(\bullet)$ can be any non-linear function, and θ some associated parameters. An initial condition, $\boldsymbol{u}(0) = \boldsymbol{u}_0$, is commonly required to obtain an unique solution at some later time, t. The model is integrated forward in time,

$$\boldsymbol{u}(t) = \boldsymbol{u}_0 + \int_0^t f(\boldsymbol{u}(s), \theta) ds, \qquad t \in [0, \infty) \ , \tag{1.2}$$

to obtain the unique solution, $\boldsymbol{u}(t)$. However, for phenomena where both spatial variation and time evolution are important, such as fluid and ocean flows, the dynamical system models are commonly formulated using partial differential equations (PDEs) of the form,

$$\frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial t} = \mathcal{L}(\boldsymbol{u}(\boldsymbol{x},t),\theta), \qquad \boldsymbol{x} \in \mathcal{D} \subset \mathbb{R}^n,\ t \in [0,\infty)\ , \tag{1.3}$$

where $\mathcal{L}$ is any non-linear function containing spatial derivatives. At any given time, t, the state variable $\boldsymbol{u}(\cdot,t)$ is a vector field defined over a spatial domain $\mathcal{D} \subset \mathbb{R}^n$ in one, two, or three dimensions (n). A unique solution to the above system exists given some initial condition ($\boldsymbol{u}(\boldsymbol{x},0) = \boldsymbol{u}_0(\boldsymbol{x}),\ \boldsymbol{x} \in \mathcal{D}$) and well-defined boundary conditions ($\mathcal{B}(\boldsymbol{u}(\boldsymbol{x},t)) = \boldsymbol{u}_b(\boldsymbol{x},t),\ \boldsymbol{x} \in \partial\mathcal{D}$). A PDE can also be interpreted as an infinite-dimensional ODE system. Unfortunately, solving PDEs is non-trivial. However, a variety of numerical techniques exists under the umbrella of computational physics to cater to them [5]. Computational physics is a rapidly-growing and interdisciplinary area with vast amounts of scientific knowledge already in existence.

In the remainder of the thesis, the use of the term "models" will refer to "dynamical system models".

1.1 Missing Dynamics in Existing Models

Often, a lot of scientific rigour in the form of knowledge of conservation laws, complex mathematical tools, and carefully-obtained experimental data goes into the formulation of dynamical system models (and mathematical models, in general). However, it is not often possible to derive the 'perfect model' which describes the phenomenon of interest exactly. This is because real-world phenomena are highly complex and it is not possible to account for every atom or molecule. Most models only resolve spatiotemporal scales, processes, and field variables to a certain level of accuracy because of the high computational costs. In many cases, due to incomplete or even poor understanding, there exists multiple competing model hypotheses, and there

are interactions and processes completely unknown to scientists. We refer to such truncated, unresolved, or unmodeled scales, processes, and variables as "missing dynamics", which often limit the reliability and usefulness of simulations, especially for scientific, engineering, and societal applications where longer-term model predictions are needed to guide decisions. A variety of modeling techniques have been developed to represent the missing dynamics. Techniques that express these missing dynamics as functions of modeled state variables and parameters are referred to as closure models. Turbulence closure [6, 7] is a classic example of this approach, however, deriving these closure models even for relatively simple systems could constitute a PhD thesis in its own right.

1.2 Bayesian Model Learning for Dynamical Systems

Due to the inability to account for the missing dynamics, there exists an inherent model uncertainty, which could manifest in many different forms. These might include uncertainty in the initial conditions, boundary conditions, parameters, the option to choose from a set of candidate functions or model complexity, or the functional form of the model being completely unknown. A Bayesian approach is then useful because it allows to take into account prior information in accord with their uncertainties and updates uncertainty estimates when data becomes available. The results of Bayesian learning are often easy to interpret and provide quantifiable uncertainty on the answers. Bayesian approaches are already widely used in variety of disciplines such as life sciences, finance, social-sciences, etc. However, applications to dynamical systems have been limited in their scope. Due to the inherent high-dimensionality of PDE-based systems (dimensions from $\mathcal{O}(10^3)$, to above $\mathcal{O}(10^8)andmore$; [8]), they include simplifying assumptions such as linearity and Gaussianity, and often are only limited to parameter estimation or do not provide full posterior probability distributions. The reader is referred to Lin, 2020 [9] section 1.3 for a comprehensive review of different methods.

Recent developments in our group by Lu and Lermusiaux, 2014 & 2021 [10, 8] and

Lin, 2020 [9] have for the first time made it feasible to perform joint inference of state variables, parameters, and the dynamical model itself for realistic high-dimensional dynamical systems with non-Gaussian statistics and nonlinear dynamics. Their theory and computational framework propagate uncertainty for each candidate model in a reduced subspace by the dynamically orthogonal (DO) equations [11, 12, 13, 14, 15], and jointly infer state variables and model parameters via data assimilation by the Gaussian mixture model (GMM) based Kalman filter (GMM-DO filter; [16, 17]). Based on observations and model predictions, the framework learns the underlying dynamical models in a principled hierarchical Bayesian way. This formulation has been very successful for cases when the true model is exactly equal to one of the candidate models, or when there are a limited number of candidate models to choose from even if none is the exact one. However, it might be the case that none of the candidate models is exactly equal to the true model, or there are too many candidates, or the functional form is elusive to the scientists, in which case the candidate model space becomes infinite. Thus, one of the goals of this thesis is to extend and generalize the discrimination based model learning developed by Lu and Lermusiaux, 2014 & 2021 [10, 8] and Lin, 2020 [9] to allow for interpolation in the space of known candidate models, and also the discovery of new models in an efficient but rigorous fashion.

1.3 Deep Learning for Dynamical Systems

We are in the midst of the "big data" revolution, and the advances in deep learning have revolutionized the way we analyse and utilize data. Learning models solely from data works very well for applications where a vast multitude of data is readily available and useful prior models are not available, but struggles in situations where data is scarce both in space and time. Thus, using deep learning techniques which were found to be successful with image and language tasks, and applying them out-of-the box to dynamical systems such as fluid and ocean flows fails to generalize and compete with existing models which were derived using sound scientific analysis over centuries. Deep

learning, existing dynamical system models, and computational physics techniques are in themselves very powerful tools in the arsenal of mathematical modeling, and have their own strengths and weaknesses which can potentially complement each other. For example, an existing dynamical system model could act as a regularizer for the deep learning model when data is scarce, or deep learning could specifically switch out tasks in computational physics that are either computationally very expensive or use heuristics. Thus, it becomes imperative to use them in consonance and leverage existing scientific knowledge to come up with new hybrid methods which will help push the boundaries of computational science.

Over the last 5 years or so, many novel methods have been developed which meld deep learning and dynamical systems in interesting ways. We will next outline some of the significant methods in the literature. Physics informed neural networks (PINNs; [18, 19]) were originally developed to learn a direct map from a point in space and time, $(\boldsymbol{x}, t)$, to the state of the dynamical system, $\boldsymbol{u}(\boldsymbol{x}, t)$, with the training loss constrained using the known model equations, boundary and initial conditions, thus, circumventing the need for any training data. Nowadays, the term "PINNs" is used more colloquially to refer to incorporation of any insight about the dynamical system at hand into the learning process, for example, custom architectures, loss functions, etc. Another popular approach is to use recurrent networks such as long-short term memory networks (LSTMs), gated recurrent units (GRUs) etc. as surrogates for the discretized time-integration (equation 1.2) step [20, 21, 22, 23]. Sparse regression-based methods (SINDy; [24, 25]) have also been developed for the discovery of model equations, which are promising as they do not require prior knowledge, however, they often require large data sets. Variations of SINDy, such as weak SINDy, have been developed to learn PDEs [26], adaptively generate features to increase the library of models [27], and extend to Bayesian identification [28]. Some methods use Genetic algorithms [29] and reinforcement learning [30, 31, 32] to perform searches in the space of potential models. Next is the neural ODE framework [33], which parameterizes the temporal derivative of the state variable ($d\boldsymbol{u}(t)/dt$ in equation 1.1) using neural networks to learn the system dynamics. A significant extension of this method is

the universal differential equations framework [34], which extends it to any class of differential equations. However, there is still need to study behaviour and develop innovative architectures, efficient training methods, etc., for the different classes of differential equations. Finally, we mention DeepONet [35] and the neural operators [36] framework which leverages the Green's function corresponding to a PDE and universal approximation theorem for operators to justify learning the operator-mapping between infinite dimensional input-output function spaces, and generalizes well.

Overall, the field of Scientific Machine Learning (SciML; [37]) is burgeoning with innovative methods that combine existing scientifically-derived differential equation models and computational physics techniques with machine learning to make realistic simulations of real-world phenomena feasible, given current computational capabilities. This thesis is a new contribution to the same.

1.4 Contributions and Structure of this Thesis

The overarching goal of this thesis is to develop novel machine learning methods to learn and discover missing dynamics in existing dynamical system models. We develop both new Bayesian learning and new deep learning methodologies, that enable leveraging all the existing scientific knowledge in the learning process. For the Bayesian learning part of the thesis, we build on the existing theory and computational framework of Lu and Lermusiaux, 2014 & 2021 [10, 8] and Lin, 2020 [9], and equip it to interpolate in the space of known candidate models, and also discover new models in an efficient fashion. We also develop and adapt techniques for improving uncertainty quantification using the DO equations for multidisciplinary dynamics, and observation planning in resource constraint environments to best achieve the learning objectives. For the deep learning part of the thesis, we build a neural ODE style framework that allows learning non-Markovian closure models. We also ensure it is generalizable and interpretable. Our primary applications of interest include fluid and ocean flows, marine ecosystems, and ocean acidification, all of which play a vital role in climate change, fisheries management, food security, and require immediate understanding,

monitoring, forecasting, and intervention. However, the developed frameworks are application-agnostic and could be widely extended to other fields such as control theory, robotics, pharmacokinetic-pharmacodynamics, chemistry, economics, biological regulatory systems, etc.

Before concluding this chapter, we provide a brief exposition of this thesis. In chapter 2, we will derive the developed Bayesian model learning methodology for simultaneous estimation of states, parameters, and model discovery, with applications to lower-trophic-level models for the marine ecosystem. In chapter 3, we will apply our Bayesian model learning framework to fish models. In chapter 4, we will use it to discover missing dynamics in an existing ocean acidification model from real-world data collected in the Gulf of Maine. In chapter 5, we will develop schemes for handling stochastic boundary conditions, numerics, data-driven subspace augmentation, and observation planning to answer what, when, and where to measure. In chapter 6, we will present our new neural closure models framework to learn non-Markovian closure parameterizations for known-physics/low-fidelity models using data from high-fidelity simulation data. Applications will include accounting for truncated modes in reduced-order-models, capturing the effects of subgrid-scale processes in coarse models, and augmenting the simplification of complex biological and physical-biogeochemical models. In chapter 7, we will develop the novel generalized and interpretable extension of our neural closure models framework, and demonstrate its performance. Finally, in chapter 8, we will make concluding remarks and discuss some future directions.

Chapter 2

Bayesian Learning Machines for Coupled Biogeochemical-Physical Models

The ability to predict and understand marine ecosystems is essential for addressing many of the grand challenges faced by humanity, such as, climate change, food security, and sustainability. In broad terms, marine ecosystems can be seen as food webs, or flow of food/energy from nutrients, to phytoplanktons, to zooplanktons, to fish, and finally recycling back to the nutrients [38, 39]. However, there does not yet exist a single generic model that accurately represents all the components in marine food webs due to the presence of highly complex biological processes with many unknown interactions, as well as of nonlinear physical forcing. Many approximations are thus made and only parts of a food web are commonly modeled. The interactions of what is modeled with other portions of the food web then need to be parameterized. The biology is also forced by the dynamic physical state of the ocean. The overall result are biogeochemical-physical modeling systems. In these systems, the nutrients and individual species (plankton, fish, etc.) are broadly categorized and represented as components or state variables, defined as a concentration of nutrients, biomass, or number of organisms per unit volume of water. The dynamics of these state variable fields thus consists of reaction terms representing biogeochemical processes such as

nutrient uptake, grazing, death, etc., and of forcing by physical processes such as advection, diffusion, and sun light.

A plethora of biogeochemical models have been proposed by scientists which differ in their ability to resolve different biological processes. This ability is determined by the complexity of the model. Models of higher complexity have more biological components, functional terms, and parameters. However, process terms and parameters are often poorly known, which hampers the utility of highly complex models [40, 41]. Some prominent biogeochemical models of varying complexities are listed next. The simplest and the most popular are 3-component nutrient-phytoplankton-zooplankton (NPZ) models [42, 43]. NPZ models are easily explored and understood, thus serving as an important tool in oceanographic research. Also modeling the intermediate state of detritus leads to four component NPZ-Detritus biological models [44]. An intermediate complexity model is the 7-component additionally accounting for bacteria, nitrate, ammonium, and dissolved organic nitrogen proposed in [45]. One of the most complex lower-trophic level marine ecosystem models is the European Regional Seas Ecosystem Model (ERSEM, [46, 47, 48]) which was developed for the North Sea. Many different choices of functional forms exists for each of the biological processes [40] which helps develop application specific variants of the above models.

Biogeochemical models are developed using semi-empirical methodologies, which leads to a lot of uncertainty associated with the parameters, functional form, and the level of complexity of these models. A set of parameter values working in a particular part of the ocean, might not work anywhere else, or there may be seasonal variability in parameter values. This also leads to uncertainty in the state variables being predicted using these models. Observations are already an integral part of the formation of these models, however, are in general only used for data fitting in order to find appropriate parameter values or functional forms of these models in offline mode. With the availability of state-of-the-art data assimilation techniques, we should instead use these observations in a Bayesian sense to learn state variables, parameter values, and discriminate/discover functional forms of biogeochemical models with quantifiable uncertainty for better estimation and prediction of ocean biology. A variety

of data assimilation techniques are being applied to biogeochemical models and can be categorized broadly into two categories. First is parameter optimization, where model parameters are calibrated by minimizing misfits between model output and independent observations [49, 50, 41]. The second is sequential data assimilation, which helps to estimate model states taking into account the observations available while integrating the model forward in time [51, 52, 53]. However, very few studies deal with the simultaneous estimation of parameters, state variables, and model equations. Some notable examples include Dowd et al., 2011 ([54]) using a Monte Carlo approach with an ensemble of 200 simulations lasting 30-days during the spring bloom in the North Atlantic. A twin experiment was conducted in an idealized framework, with surface observations of phytoplankton concentration to perform a Kalman filter based uncertain parameter and state estimation using the technique of state augmentation. Similarly, Julier and Uhlmann, 2010 ([55]) performed state and parameter estimation in a non-linear phytoplankton-zooplankton model using two different Markov Chain Monte Carlo (MCMC) algorithms in an identical-twin setting. Lately, along with state and parameter estimation, the selection of optimal complexity of biogeochemical models has become an active area of research [41]. With the recent advancement and popularization of machine learning, several machine learning methods have been developed for the discovery of model equations. The sparse regression-based methods (SINDy; [24, 25]) are promising as they do not require prior knowledge, however, they often require large data sets. Variations of SINDy have been developed such as weak SINDy to learn PDEs [26], the adaptive generation of features to increase the library of models [27], and extensions to Bayesian identification [28]. Some methods use genetic algorithms [29] and reinforcement learning [30, 31, 32] to perform searches in the space of potential models. However, most of these approaches do not provide uncertainty estimates for the discovered models. Methods have also combined prior knowledge about underlying governing equations for model recovery and refinement. For example, Raissi and Karniadakis, 2018 [18] successfully used Gaussian processes to learn the values of the parametric response of partially-known nonlinear differential equations. Unfortunately, data and knowledge of governing laws is a luxury in the

case of biogeochemical models.

There are as many biogeochemical models as biologists. Hence, the selection of models which best explains the data in a principled way is much needed. The rigorous Bayesian learning approach developed by Lu and Lermusiaux, 2014 & 2021 [10, 8] addresses most of the above needs and drawbacks, allowing for simultaneous estimation of states and parameters along with discrimination among candidate models using sparse observations. However, what to do in the case when none of the candidate models is exactly equal to the true model? Or the functional form is yet completely elusive to scientists? Could we interpolate within and extrapolate out of known model spaces, while providing accurate joint probability distributions for model states, parameters, and formulations? Could such Bayesian learning be efficient and successful with high-dimensional and multidisciplinary stochastic PDEs? Thus, the goal of the present paper is to extend and generalize the discrimination-based model learning developed in [10, 8], to allow for interpolation in the space of known candidate models and the discovery of new models in an efficient fashion. Our novel model learning and discovery is achieved by introducing special stochastic parameters and stochastic linear piece-wise function approximations. We address the challenges of multidisciplinary dynamics and develop a rigorous PDE-based Bayesian learning framework by combining the Dynamically Orthogonal (DO) methodology [11, 12, 14, 13, 15] for reduced dimension stochastic evolution, and Gaussian Mixture Model (GMM)-DO filtering algorithm [16, 17]. In Sect. 2.1, we present the problem statement. In Sect. 2.2, we develop the special parameters for formulating model uncertainty and obtain novel Bayesian methods for model learning and discovery. In Sect. 2.3, details of the biogeochemical models used in this study, governing equations, modeling domain, true solution generation, etc. are presented. Finally, in Sect. 2.4 we show the application of our algorithms using four experiments of varying complexities and learning objectives. The conclusions of this study are provided in Sect. 2.5.

2.1 Problem Statement

A single mathematical model that exactly captures all the physical and biological processes occurring in the real-world does not yet exist. Hence, there is inherent model uncertainty that manifests in many forms, including: initial and boundary condition uncertainties; unreliable parameter values; multiple competing candidate model functions; unknown functional forms; missing model terms; and, debatable complexity of the model. In this work, we consider discriminating among candidate models as well as learning among compatible models and discovering new model formulations. Compatible models are models that can be related to a single modeling system theoretically and that can also be combined numerically. Compatible models can nonetheless represent different dynamics, e.g., our goals include learning which dynamics is or is not present based on observations.

In general, we consider a stochastic dynamical modeling system defined on a domain $\mathcal{D}$, governing the dynamics of $\boldsymbol{u}(\boldsymbol{x}, t; \omega) : \mathbb{R}^n \times [0, T] \to \mathbb{R}^{N_v}$, the stochastic state vector comprising N_v dynamical state variable fields. The realization index ω belongs to a measurable sample space Ω and the model depends on a vector $\boldsymbol{\theta}(\omega)$ of N_θ uncertain parameters. To encompass the majority of scenarios, we write the general form of the uncertain dynamical modeling system as follows,

$$\begin{aligned} \frac{\partial \boldsymbol{u}(\boldsymbol{x}, t; \omega)}{\partial t} &= \mathcal{L}[\boldsymbol{u}(\boldsymbol{x}, t; \omega), \boldsymbol{\theta}(\omega), \boldsymbol{x}, t] + \widehat{\mathcal{L}}[\boldsymbol{u}(\boldsymbol{x}, t; \omega); \omega] + \widetilde{\mathcal{L}}[\boldsymbol{u}(\boldsymbol{x}, t; \omega); \omega]\,, \\ &\qquad\qquad \boldsymbol{x} \in \mathcal{D},\ t \in [0, T],\ \omega \in \Omega\,, \\ \text{with} \quad & \boldsymbol{u}(\boldsymbol{x}, 0; \omega) = \boldsymbol{u}_o(\boldsymbol{x}; \omega)\,, \\ \text{and} \quad & \mathcal{B}[\boldsymbol{u}(\boldsymbol{x}, t; \omega)] = \boldsymbol{b}(\boldsymbol{x}, t; \omega),\ \boldsymbol{x} \in \partial\mathcal{D},\ t \in [0, T],\ \omega \in \Omega\,, \end{aligned} \tag{2.1}$$

where $\boldsymbol{u}_o(\boldsymbol{x}; \omega)$, $\mathcal{B}$, and $\boldsymbol{b}(\boldsymbol{x}, t; \omega)$ are the stochastic initial conditions, boundary condition operators, and boundary values respectively. The functional form of the first dynamics term $\mathcal{L}[\boldsymbol{u}(\boldsymbol{x}, t; \omega), \boldsymbol{\theta}(\omega), \boldsymbol{x}, t]$ is assumed to be known, but with uncertain parameters. The second term $\widehat{\mathcal{L}}[\boldsymbol{u}(\boldsymbol{x}, t; \omega); \omega] \in \{\widehat{\mathcal{L}}_1[\boldsymbol{u}(\boldsymbol{x}, t; \omega); \omega], ..., \widehat{\mathcal{L}}_{N_m}[\boldsymbol{u}(\boldsymbol{x}, t; \omega); \omega]\}$,

represents a set of compatible candidate functional forms, where N_m is the number of candidates. For example, for reaction terms, model functions are often from the polynomial, exponential, and/or sinusoidal families, and can be rational or irrational functions. The third term $\widetilde{\mathcal{L}}[\boldsymbol{u}(\boldsymbol{x},t;\omega);\omega]$ has a functional form completely unknown. The stochastic initial and boundary condition formulations can also have uncertain function forms, similar to the dynamical modeling system itself, e.g. they can be known, belonging to a family, or unknown.

In some cases, candidate models have different complexities,

$$\mathcal{M}_i: \quad \begin{cases} \frac{\partial u_1^i(\boldsymbol{x},t;\omega)}{\partial t} = & \mathcal{L}_1^i[u_1^i(\boldsymbol{x},t;\omega),...,u_{N_v(i)}^i(\boldsymbol{x},t;\omega),\boldsymbol{\theta}^i(\omega),\boldsymbol{x},t;\omega] \\ & \vdots \\ \frac{\partial u_{N_v(i)}^i(\boldsymbol{x},t;\omega)}{\partial t} = & \mathcal{L}_{N_v(i)}^i[u_1^i(\boldsymbol{x},t;\omega),...,u_{N_v(i)}^i(\boldsymbol{x},t;\omega),\boldsymbol{\theta}^i(\omega),\boldsymbol{x},t;\omega] \\ & i=1,...,N_m \end{cases}, \quad (2.2)$$

where each model, $\mathcal{M}_i$, has $N_v(i)$ number of state variables ($\{u_1^i,...,u_{N_v(i)}^i\}$) from a pool of candidates, and their aggregates. In such situations, the candidate models can often remain compatible with each other, for example low complexity models are embedded in higher complexity ones. We refer to such classes of candidate models as, *compatible-embedded models*. Of course, in general, uncertainty in parameter values, functional forms, and complexities occur simultaneously.

Let $\boldsymbol{U}(t;\omega) \in \mathbb{R}^{N_v N_x}$ denote the spatially discretized state vector of the continuous field $\boldsymbol{u}(\boldsymbol{x},t;\omega)$. where N_x denotes the dimension of the discretized state space. Next, we assume that the observations ($\boldsymbol{\mathcal{Y}}(t;\omega)$) are indirect, noisy, and related to $\boldsymbol{U}(t;\omega)$ according to the linear model from the state to the data space,

$$\boldsymbol{\mathcal{Y}}(t;\omega) = \boldsymbol{H}\boldsymbol{U}(t;\omega) + \boldsymbol{V}(t;\omega), \qquad \boldsymbol{V}(t;\omega) \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{R}) \quad (2.3)$$

where N_y is the number of available observations; $\boldsymbol{H} \in \mathbb{R}^{N_y \times N_v N_x}$ the observation matrix; and $\boldsymbol{V} \in \mathbb{R}^{N_y}$ a zero-mean, uncorrelated Gaussian measurement noise with covariance matrix $\boldsymbol{R} \in \mathbb{R}^{N_y \times N_y}$. Observations are assumed to be available only at

discrete time-instants, t_k for $k = 1, 2, ..., K$.

In summary, our specific objectives are thus two-folds, first to solve the stochastic forward-modeling system (Eqs. 2.1 & 2.2), taking into account all the associated uncertainties including compatible, compatible-embedded, and unknown model terms; and second to simultaneously learn, in the Bayesian sense, the state fields, parameters, and model equations based on the observation model (Eq. 2.3). Our Bayesian learning thus need to evolve the prior and posterior probabilities of state fields, parameters, and model formulations, given the observations available and all uncertainties. The overall goal is to accurately represent these probability density functions (pdfs), including the marginal probabilities of known, uncertain, and unknown model formulations. It is only if the observations are sufficiently informative about either the state fields, parameters, and/or model formulations, that the Bayesian machine will identify the true state variables, true parameters, and/or true model. If the observations are not sufficiently informative, the perfect Bayesian machine will not lead to prefect identification, but provide the exact posterior probabilities of the models, parameter values and/or state variable fields.

2.2 General Bayesian Learning Methodology

In this work, we start from Bayesian learning for rigorous discrimination among candidate models [10, 8]. Each candidate model then evolves the joint pdf of its state variables and parameters, independently from other models, and provides probability distributions that are conditional on the candidate model. When observations are made, both the model-conditional state variables and parameters, and the model pdfs are updated using Bayes' rules [56],

$$p_{\boldsymbol{U}|\boldsymbol{y},\mathcal{M}}(\boldsymbol{U}|\boldsymbol{y},\mathcal{M}_i) = \frac{p_{\boldsymbol{y}|\boldsymbol{U},\mathcal{M}}(\boldsymbol{y}|\boldsymbol{U},\mathcal{M}_i)}{p_{\boldsymbol{y}|\mathcal{M}}(\boldsymbol{y}|\mathcal{M}_i)} p_{\boldsymbol{U}|\mathcal{M}}(\boldsymbol{U}|\mathcal{M}_i) \ , \quad \forall\, \boldsymbol{U} \in \mathbb{R}^{N_v N_x}, \forall\, i \in \{1, ..., N_m\} \ , \tag{2.4}$$

$$p_{\mathcal{M}|\boldsymbol{y}}(\mathcal{M}_n|\boldsymbol{y}) = \frac{p_{\boldsymbol{y}|\mathcal{M}}(\boldsymbol{y}|\mathcal{M}_i)}{p_{\boldsymbol{y}}(\boldsymbol{y})} p_{\mathcal{M}}(\mathcal{M}_i) \ , \quad \forall\, i \in \{1, ..., N_m\} \ ,$$

where $\mathcal{M}_i$ is the i^{th} model candidate and the distributions $p_{\boldsymbol{U}|\mathcal{M}}(\boldsymbol{U}|\mathcal{M}_i)$ and $p_{\boldsymbol{U}|\boldsymbol{\mathcal{Y}},\mathcal{M}}(\boldsymbol{U}|\boldsymbol{y},\mathcal{M}_i)$ are the prior and posterior model-conditional state variable distributions, respectively. The model distribution $p_{\mathcal{M}}(\bullet)$ is the prior probability for each of the candidates being the true model and $p_{\mathcal{M}|\boldsymbol{\mathcal{Y}}}(\bullet|\boldsymbol{y})$ is the corresponding posterior model distribution. This pdf $p_{\mathcal{M}|\boldsymbol{\mathcal{Y}}}(\bullet|\boldsymbol{y})$ allows learning by exact Bayesian discrimination among candidate models. In particular, when observations are not sufficient to achieve unequivocally the ultimate learning objective, this posterior pdfs will correctly represent the ambiguity including possible multimodal distributions and the effects of biases in the candidate models [10, 8].

The above Bayesian learning evolves each stochastic candidate model separately. To increase efficiency, this can be circumvented, for example, when models are compatible or compatible-embedded. Next, we thus develop new stochastic parameterizations that unify all such candidate models into a single general modeling system. We recast the model learning into new parameter estimation problems, using special stochastic parameters (Sect. 2.2.1) and stochastic piece-wise function approximation theory (Sect. 2.2.2). We then evolve the joint probabilities of the state fields and of the regular and special parameters using new stochastic DO equations (Appendix A and Sect. 2.3.3). Finally, at each observation time, we perform Bayesian learning using the GMM-DO filter (Appendix B) with state augmentation (Appendix C). The overall methodology (Sect. 2.2) avoids the computation of the discrete marginal likelihoods, $p_{\boldsymbol{\mathcal{Y}}|\mathcal{M}}(\boldsymbol{y}|\mathcal{M}_i)$, and instead learns in a parameterized continuous model space. It thus extends learning among discrete model formulations to learning within a continuous infinite range of formulations as well as across models of different complexities and into unknown models. In other words, we remain able to discriminate among existing models, but we can now also interpolate or even extrapolate in the space of models to discover new ones.

2.2.1 Special Stochastic Parameters: Compatible and Compatible-embedded Models

Let us first consider the case when according to prior scientific knowledge, the uncertain model belongs to a set of compatible candidate functional forms ($\hat{\mathcal{L}}[\bullet]$; Eq. 2.1). In order to recast this learning problem with multiple models into a learning problem with a single model and parameter estimation, the compatible candidate model functions are added to each other but only after being multiplied with novel stochastic parameters. Each of the candidates is thus assigned a special stochastic parameter that can take discrete or continuous values depending on the learning objectives and prior knowledge. For example, binary values would be utilized to discriminate between presence or absence of certain functions, while other values would be utilized to allow some linear interpolation within the space defined by the compatible candidate models. To complete Bayesian learning, when observations are collected, the probability distributions of these uncertain special parameters ($\alpha_k(\omega)$'s, $k = 1, ..., N_m$) are updated and their mean values estimated alongside these of other regular parameters ($\boldsymbol{\theta}(t;\omega)$), using state augmentation. Summarizing, the general model can thus be written as a stochastic linear combination of the candidates,

$$\hat{\mathcal{L}}[\boldsymbol{u}(\boldsymbol{x},t;\omega),t;\omega] = \sum_{k=1}^{N_m} \alpha_k(\omega)\mathcal{L}_k[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{x},t;\omega] \; . \tag{2.5}$$

where the distribution of the $\alpha_k(\omega)$ is updated at each observation time. This new formulation can thus both help select active candidate functions and identify their linear combinations. It allows interpolating in the space of known candidate functions.

Next, we extend this approach to learn model complexity (Eq. 2.2). This is achieved by defining new states multiplied with special stochastic parameters, $\boldsymbol{u}'_k = \beta_k(\omega)\boldsymbol{u}_k$, and a general model, $\mathcal{L}'_k$, which encompasses all the candidates in the class

of compatible-embedded models,

$$\frac{\partial \boldsymbol{u}'_k(\boldsymbol{x},t;\omega)}{\partial t} = \mathcal{L}'_k[\boldsymbol{u}'_1(\boldsymbol{x},t;\omega),...,\boldsymbol{u}'_{N_v}(\boldsymbol{x},t;\omega),\boldsymbol{\theta}(t;\omega),\boldsymbol{\beta}(\omega),\boldsymbol{x},t;\omega],\ k=1,...,N_v \tag{2.6}$$

where $N_v = \max\{N_v(i)\}_{i=1}^{N_m}$. By learning these special parameters, we can eliminate certain state variables or aggregate them to form new states, and determine the model of appropriate complexity that best explains the observed data. To illustrate such combinations of compatible-embedded models into a general model, let us consider a case with only two candidate models ($N_m = 2$ in Eq. 2.2). Let us further assume that the set of states of the first model ($\{u_1,...,u_{N_v(1)}\}$) are fully contained within the set of states of the second model ($\{u_1,...,u_{N_v(1)},...,u_{N_v(2)}\}$), and the goal is to discriminate between the presence or absence of either of the model. Using a special stochastic parameter $\beta(\omega)$ that is allowed to take only binary values and new states $\boldsymbol{u}'_{N_v(1)+1} = \beta(\omega)\boldsymbol{u}_{N_v(1)+1}$, ..., $\boldsymbol{u}'_{N_v(2)} = \beta(\omega)\boldsymbol{u}_{N_v(2)}$, the general model can be written as (based on Eq. 2.2 and omitting explicit dependence on $\boldsymbol{x}$, t, & ω for brevity),

$$\begin{aligned}
\frac{\partial u_1}{\partial t} =& (1-\beta)\mathcal{L}^1_1[u_1,...,u_{N_v(1)},\boldsymbol{\theta}^1] + \beta\mathcal{L}^2_1[u_1,...,u_{N_v(1)},u'_{N_v(1)+1},...,u'_{N_v(2)},\boldsymbol{\theta}^2]\ ,\\
&\vdots\\
\frac{\partial u_{N_v(1)}}{\partial t} =& (1-\beta)\mathcal{L}^1_{N_v(1)}[u_1,...,u_{N_v(1)},\boldsymbol{\theta}^1] + \beta\mathcal{L}^2_{N_v(1)}[u_1,...,u_{N_v(1)},u'_{N_v(1)+1},...,u'_{N_v(2)},\boldsymbol{\theta}^2]\ ,\\
\frac{\partial u'_{N_v(1)+1}}{\partial t} =& \beta\mathcal{L}^2_{N_v(1)+1}[u_1,...,u_{N_v(1)},u'_{N_v(1)+1},...,u'_{N_v(2)},\boldsymbol{\theta}^2]\ ,\\
&\vdots\\
\frac{\partial u'_{N_v(2)}}{\partial t} =& \beta\mathcal{L}^2_{N_v(2)}[u_1,...,u_{N_v(1)},u'_{N_v(1)+1},...,u'_{N_v(2)},\boldsymbol{\theta}^2]\ ,
\end{aligned} \tag{2.7}$$

where $\beta(\omega) = 0$ leads to the first candidate model, and $\beta(\omega) = 1$ to the second candidate model. In similar fashion, we can derive the general model for cases with more than two candidate models, with states in one model being aggregate of states in other models, etc.

2.2.2 Stochastic Piece-wise Linear Function Approximations: Unknown Models

The above two new uses of special stochastic parameters require a set of candidate functional forms to choose from. However, in some cases, there might be no such prior information / candidates available, hence the unknown part $\widetilde{\mathcal{L}}$ of the model (Eq. 2.1). These model functions then need to be discovered. We thus propose to parameterize such an unknown function space using stochastic piece-wise continuous functions. In the present work, we consider dense piece-wise linear functions as this representation is both rich and simple, and provides practical approximations of any unknown function. It greatly enhances the functional space in which we can perform our Bayesian search, and enables the discovery of new learned functions.

For brevity, let us only consider the scalar case, where $\widetilde{\mathcal{L}}[u(\boldsymbol{x}, t; \omega); \omega]$ is the unknown function (Eq. 2.1) of a single scalar state variable. Also, it is often the case that prior information about the range of values taken by the state variable is available, $u(\boldsymbol{x}, t; \omega) \in [u_L, u_R],\ \forall \boldsymbol{x} \in \mathcal{D}$ and $t \in [0, T]$. Now, to define a parameterization using continuous piece-wise linear segments, consider the range $\mathcal{H} = [u_L, u_R]$ to be an indexed collection of intervals with non-zero measure $\{I_i = [u_L^i, u_R^i]\}_{0 \leq i \leq N_I}$ forming a partition of $\mathcal{H}$, i.e.,

$$\mathcal{H} = \bigcup_{i=0}^{N_I} I_i \quad \text{and} \quad \mathring{I}_i \cap \mathring{I}_j = \emptyset \quad \text{for } i \neq j \ , \tag{2.8}$$

and we use $N_I + 1$ points to discretize the range, such that,

$$u_L = u_L^0 < u_R^0 = u_L^1 < ... < u_R^{N_I - 1} = u_L^{N_I} < u_R^{N_I} = u_R \ . \tag{2.9}$$

Let $\{\Psi_0, ..., \Psi_{N_I+1}\}$ be the linear functions defined on each of these element,

$$\Psi_0(u) = \begin{cases} \frac{1}{(u_R^0 - u_L)}(u_R^0 - u) & \text{if } u \in I_0 \ , \\ 0 & \text{otherwise} \end{cases}$$

$$\Psi_k(u) = \begin{cases} \frac{1}{(u_R^{k-1} - u_L^{k-1})}(u - u_L^{k-1}) & \text{if } u \in I_{k-1} \ , \\ \frac{1}{(u_R^k - u_L^k)}(u_R^k - u) & \text{if } u \in I_k \ , \\ 0 & \text{otherwise} \end{cases} \qquad \text{for } k \in \{1, ..., N_I\} \ , \tag{2.10}$$

$$\Psi_{N_I+1}(u) = \begin{cases} \frac{1}{(u_R - u_L^N)}(u - u_L^N) & \text{if } u \in I_{N_I} \ , \\ 0 & \text{otherwise} \end{cases}$$

and $\gamma_k(\omega)'s, \ k \in 0, ..., N_I + 1$ be $N_I + 2$ *stochastic expansion coefficients* that parameterize the unknown function space by taking a linear combination of the functions defined on each elements. Hence, all together we obtain:

$$\widetilde{\mathcal{L}}[u(\boldsymbol{x}, t; \omega); \omega] = \sum_{k=0}^{N_I+1} \gamma_k(\omega) \Psi_k(u(\boldsymbol{x}, t; \omega)) \ . \tag{2.11}$$

Thus, estimating the stochastic parameters γ_k's, in turn leads to learning of the unknown model function. The above formulation ensures C^0 continuity in the functional space. The prior distribution of these parameters define the functional space in which the search is performed. By construction, this parameterized space can be made as dense as desired. Next, we extend this formulation to any general basis, such as higher degree polynomials, etc.

2.2.3 Stochastic Piece-wise Polynomial Function Approximations: Unknown Models

Now, let us say that we want to learn the unknown term $\tilde{\mathcal{L}}[u(\boldsymbol{x}, t; \omega); \omega]$ using a polynomial of order k, P^k. The $N_I + 1$ points introduced in the discretization for the linear case (equation 2.9), will now act as the global nodes. For each of the

interval I_i, we introduce local nodes, $\xi_{i,m} = u^i + \frac{m}{k}(u^{i+1} - u^i)$, and let $\{P^k_{i,0}, ..., P^k_{i,k}\}$ be the Lagrange or similar interpolation polynomials associated with these nodes. For $j \in \{0, ..., k(N_I + 1)\}$ with $j = ki + m$ and $0 \leq m \leq k - 1$, define the function Ψ_j element-wise as follows:

For $1 \leq m \leq k - 1$,

$$\Psi_{ki+m}(u) = \begin{cases} P^k_{i,m}(u) & \text{if } u \in I_i, \\ 0 & \text{otherwise} \end{cases} \tag{2.12}$$

and for $m = 0$,

$$\Psi_{ki}(u) = \begin{cases} P^k_{i-1,k}(u) & \text{if } u \in I_{i-1}, \\ P^k_{i,0}(u) & \text{if } u \in I_i, \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

with obvious modifications if $i = 0$ or $N_I + 1$. Now, let $\gamma_j(\omega),\ j \in 0, ..., k(N_I + 1)$ be $k(N_I + 1) + 1$ stochastic expansion coefficients, hence, the unknown function could be expressed as a linear combination of the polynomial basis (Ψ_j's) defined above,

$$\tilde{\mathcal{L}}[u(\boldsymbol{x}, t; \omega); \omega] = \sum_{j=0}^{k(N_I+1)} \gamma_j(\omega)\Psi_j(u(\boldsymbol{x}, t; \omega)) \tag{2.14}$$

Assigning appropriate priors to stochastic parameters, γ_j's, determines the initial function space in which the search is performed. The above discussion is adopted from chapter 1 of Guermond, 2016 [57].

2.2.4 Bayesian Learning: stochastic DO PDEs, GMM-DO Filter, and Learning Skill

To provide an accurate and informative prior for our new Bayesian learning paradigm with uncertain and unknown nonlinear dynamics and PDEs, we employ Dynamically Orthogonal (DO) equations [11, 12, 13]. The DO equations are optimal reduced

order differential equations that evolve, based on the governing nonlinear dynamics, the dominant probabilistic subspace. Their derivation with uncertain parameters is provided in Appendix A and in Sect. 2.3.3 for biogeochemical specifics.

For the Bayesian learning at each observation time, the GMM-DO filter [16, 17] is used to perform nonlinear, non-Gaussian Bayesian updates of the probability distribution of the state variables, as detailed in Appendix B. This approach is extended to stochastic dynamical models featuring uncertain parameters using the technique of state augmentation [58, 8] (Appendix C), enabling joint Bayesian learning of state variables and parameters. Then, combining the GMM-DO filter with state augmentation, our novel schemes of recasting the learning of compatible and compatible-embedded models into special parameter estimations and of parameterizing the space of unknown model functions using piece-wise linear continuous functions, allow for efficient simultaneous Bayesian estimation of state variable fields, parameters, and model equations themselves, all while using a single modeling system. For the former scheme, the learning occurs within the space of candidate models while for the latter, it occurs outside of that space and into the space of unknown model functions, hence providing the capability for full model discovery. Most importantly, this discovery is interpretable as it is in the form of piece-wise linear continuous functions. In addition, all of our Bayesian estimations provide much more than maximum likelihood estimates: they predict and update the complete joint probability distribution of states, parameters, and models. When the observations are not sufficiently informative to learn and eliminate all but one model, parameter value, or state variable field, our Bayesian learning can provide the correct multi-modal pdfs. Our learning can indeed represent ambiguity, e.g. multiple options are possible, or even equifinality [59], e.g. a set of model estimates have the same likelihood. It can also signal the presence of bias in competing model formulations. Such capabilities will be showcased in (Sect. 2.4).

To evaluate the learning skill, we first compare the mean fields and parameters with the noisy observations, using several error metrics. We also analyze the evolution of the pdfs of fields and parameters, as well as the convergence of these pdfs with

stochastic resolution.

2.3 Biogeochemical-Physical Equations and Simulated Experiments Setup

In this section, we describe the specifics of our simulated Bayesian learning experiments. We start with the biogeochemical differential equations, their coupling with the physics PDEs, and the stochastic DO decomposition with uncertain and unknown terms. This is followed by details of the modeling domain, numerical methods, initialization of the stochastic simulations, true solution generation, simulated observations, and learning metrics.

2.3.1 Biogeochemical Models

The biogeochemical differential equations that we employ are adapted from [60, 61] and references therein, and from Newberger et. al., [3]. They meet the criterion of being compatible with each other, with low complexity models being embedded in higher complexity models (compatible-embedded-models). We will utilize three reaction models: the three-component NPZ model, i.e., nutrients (N), phytoplankton (P), and zooplankton (Z); the four-component NPZD model, i.e., N, P, Z and Detritus (D); and, the five-component NNPZD model, i.e., ammonia (NH_4), nitrate (NO_3), P, Z, and D. The NPZ model is given by,

$$\begin{aligned}
\frac{dN}{dt} &= -G\frac{PN}{N+K_u} + \Xi P + \Gamma Z + R_m\gamma Z(1-\exp^{-\Lambda P}) \ , \\
\frac{dP}{dt} &= G\frac{PN}{N+K_u} - \Xi P - R_m Z(1-\exp^{-\Lambda P}) \ , \\
\frac{dZ}{dt} &= R_m(1-\gamma)Z(1-\exp^{-\Lambda P}) - \Gamma Z \ ,
\end{aligned} \tag{2.15}$$

with G representing the optical model,

$$G = V_m\frac{\alpha I}{(V_m^2+\alpha^2 I^2)^{1/2}} \ , \quad \text{and} \quad I(z) = I_0\exp^{k_w z} \ , \tag{2.16}$$

where z is depth and $I(z)$ models the availability of sunlight for photo-chemical reactions. The parameters in Eqs. 6.26 & 6.27 are: k_w, light attenuation by sea water; α, initial slope of the P-I curve; I_0, surface photosynthetically available radiation; V_m, phytoplankton maximum uptake rate; K_u, half-saturation constant for phytoplankton uptake of nutrients; Ξ, phytoplankton specific mortality rate; R_m, zooplankton maximum grazing rate; Λ, Ivlev grazing constant; γ, fraction of zooplankton grazing egested; and Γ, zooplankton specific excretion/mortality rate. In this NPZ model (Eq. 6.26), the nutrient uptake by phytoplankton is governed by a Michaelis-Menten formulation, which amounts to a linear uptake relationship at low nutrient concentrations that saturates to a constant at high concentrations. The grazing of phytoplankton by zooplankton follows a similar behavior: their growth rate becomes independent of P in case of abundance, but proportional to available P when resources are scarce; hence, zooplankton grazing is modeled by an Ivlev function. The death rates of both P and Z are linear, and a portion of zooplankton grazing in the form of excretion goes directly to nutrients.

For the NPZD model, the only change is in the addition of detritus, which is the intermediate state before dead plankton get converted to nutrients,

$$\begin{aligned} \frac{dN}{dt} &= -G\frac{PN}{N+K_u} + \Phi D + \Gamma Z \ , \\ \frac{dD}{dt} &= R_m \gamma Z(1-\exp^{-\Lambda P}) + \Xi P - \Phi D \ . \end{aligned} \tag{2.17}$$

However, for the NNPZD model, the nutrients are divided into ammonia and nitrates, which are the two most important forms of nitrogen in the ocean [38, 39]. This helps to capture new processes such as, phytoplankton cells preferentially taking up ammonia over nitrate because the presence of ammonia inhibits the activity of the enzyme nitrate reductase essential for the uptake kinetics, the pool of ammonia coming from remineralization of detritus, and part of this ammonia pool getting oxidized to become a source of nitrate referred to as nitrification, etc. [38, 39, 62].

The NNPZD model is given by,

$$
\begin{aligned}
\frac{dNO_3}{dt} &= \Omega NH_4 - G\left[\frac{NO_3}{NO_3+K_u}\exp^{-\Psi_I NH_4}\right]P\,,\\
\frac{dNH_4}{dt} &= -\Omega NH_4 + \Phi D + \Gamma Z - G\left[\frac{NH_4}{NH_4+K_u}\right]P\,,\\
\frac{dP}{dt} &= G\left[\frac{NO_3}{NO_3+K_u}\exp^{-\Psi_I NH_4} + \frac{NH_4}{NH_4+K_u}\right]P - \Xi P - R_m Z(1-\exp^{-\Lambda P})\,,\\
\frac{dZ}{dt} &= R_m(1-\gamma)Z(1-\exp^{-\Lambda P}) - \Gamma Z\,,\\
\frac{dD}{dt} &= R_m\gamma Z(1-\exp^{-\Lambda P}) + \Xi P - \Phi D\,.
\end{aligned}
\tag{2.18}
$$

The above three models aim to capture the lower-trophic-level (LTL) interactions in the ocean ecosystem. They are the Lagrangian or ordinary differential equation (ODE) versions of these models. For realistic ocean field simulations, the above rates of change are material derivatives of dynamic tracers that are coupled with the physics using advection-diffusion-reaction PDEs. Of course, these models are not directly applicable in every ocean region without parameter tuning or modifying the functional form of the reaction terms. Regional diversity is one of the reasons for parameter and functional form (model) uncertainties.

2.3.2 Coupling with the Physics

In biogeochemical-physical models, the physics is provided by solving PDEs for the conservation of mass and momentum (Navier-Stokes), internal energy, and salt, e.g., the ocean primitive equations [63, 64]. These models often contain parameterizations to represent subgridscale processes [65, 66]. In the present work, we employ the incompressible nonhydrostatic Reynolds-averaged Navier-Stokes (RANS) PDEs [67],

$$
\begin{aligned}
\nabla.\boldsymbol{u}(\boldsymbol{x},t) &= 0, \qquad \boldsymbol{x}\in\mathcal{D}\,,\\
\frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial t} + \nabla.(\boldsymbol{u}(\boldsymbol{x},t)\boldsymbol{u}(\boldsymbol{x},t)) &= -\nabla p(\boldsymbol{x},t) + \nu_E\nabla^2\boldsymbol{u}(\boldsymbol{x},t)\,, \qquad \boldsymbol{x}\in\mathcal{D}\,,
\end{aligned}
\tag{2.19}
$$

where $\boldsymbol{u}(\boldsymbol{x},t)$ is the velocity field, $p(\boldsymbol{x},t)$ the pressure field, and ν_E the turbulent eddy viscosity.

The Lagrangian biogeochemical models (Sect. 2.3.1) are coupled with the physics using stochastic advection-diffusion-reaction (ADR) PDEs. For N_ϕ stochastic biogeochemical tracers, $\phi^i(\boldsymbol{x},t;\omega)$'s, we obtain,

$$\begin{aligned}\frac{\partial \phi^i(\boldsymbol{x},t;\omega)}{\partial t} + \underbrace{\nabla.(\boldsymbol{u}(\boldsymbol{x},t)\phi^i(\boldsymbol{x},t;\omega))}_{\text{Advection}} - \underbrace{\mathcal{K}_E\nabla^2\phi^i(\boldsymbol{x},t;\omega)}_{\text{Diffusion}} \\ = \underbrace{S^{\phi^i}(\phi^1,...,\phi^{N_\phi},\theta^1(\omega),...,\theta^{N_\theta}(\omega),\boldsymbol{x},t;\omega)}_{\text{Reaction}}\,, \qquad \forall i=\{1,...,N_\phi\},\end{aligned} \tag{2.20}$$

where $\boldsymbol{u}(\boldsymbol{x},t)$ is the deterministic velocity field governed by (2.19), $\mathcal{K}_E$ is the eddy diffusivity, $S^{\phi^i}(\phi^1,...,\phi^{N_\phi},\theta^1(\omega),...,\theta^{N_\theta}(\omega),\boldsymbol{x},t;\omega)$ are the reaction terms defined by the right-hand-side of the ODEs of Sect. 2.3.1, and the $\theta^l(\omega)$'s, $l=\{1,...,N_\theta\}$, are the uncertain biogeochemical parameters. Biogeochemical reactions are nonlinear in nature, hence, the PDEs (2.20) form a set of strongly nonlinear, stiff, and coupled PDEs.

2.3.3 Biogeochemical-Physical Stochastic Dynamically-Orthogonal PDEs

To solve the system of Eqs. (2.19 & 2.20) efficiently, we now develop the DO equations for the stochastic ADR PDEs (2.20) with model and parameter uncertainty. We first separate the reactions into known, uncertain, and unknown terms, and write (2.20) in vector form,

$$\begin{aligned}\frac{\partial \boldsymbol{\phi}(\boldsymbol{x},t;\omega)}{\partial t} + \nabla.(\boldsymbol{u}(\boldsymbol{x},t)\boldsymbol{\phi}(\boldsymbol{x},t;\omega)) - \mathcal{K}_E\nabla^2\boldsymbol{\phi}(\boldsymbol{x},t;\omega) \\ = \boldsymbol{S}^{\boldsymbol{\phi}}(\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\theta}(\omega),\boldsymbol{\beta}(\omega),\boldsymbol{x},t;\omega) \\ + \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}(\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\theta}(\omega),\boldsymbol{\alpha}(\omega),\boldsymbol{\beta}(\omega),\boldsymbol{x},t;\omega) \\ + \widetilde{\boldsymbol{S}}^{\boldsymbol{\phi}}(\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\gamma}(\omega),\boldsymbol{x},t;\omega)\,,\end{aligned} \tag{2.21}$$

where $\boldsymbol{\phi} = [\phi^i]_{i=1}^{N_\phi}$. The functional form of the first reaction term $\boldsymbol{S}^{\boldsymbol{\phi}}(\bullet) = \left[S^{\phi^i}(\bullet)\right]_{i=1}^{N_\phi}$ is assumed to be known, however it contains N_θ uncertain regular parameters $\boldsymbol{\theta}(\omega) = \left[\theta^k\right]_{k=1}^{N_\theta}$. The second term $\widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}(\bullet) = \left[\widehat{S}^{\phi^i}(\bullet)\right]_{i=1}^{N_\phi}$ is uncertain: it belongs to a family of candidate functions, parameterized using N_α special stochastic parameters $\boldsymbol{\alpha}(\omega) = \left[\alpha^k\right]_{k=1}^{N_\alpha}$, and may contain uncertain regular parameters $\boldsymbol{\theta}(\omega)$. The candidate models of different complexities are combined using N_β special stochastic parameters $\boldsymbol{\beta}(\omega) = \left[\beta^k\right]_{k=1}^{N_\beta}$. The $\beta_k(\omega)$'s multiplied with the original biological tracer fields (as described in Sect. 2.2.1) are absorbed into ϕ_i's and not explicitly shown; however, $\beta_k(\omega)$'s usually appear on the right-hand-side (RHS) of $\boldsymbol{S}^{\boldsymbol{\phi}}(\bullet)$ and $\widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}(\bullet)$. The third term $\widetilde{\boldsymbol{S}}^{\boldsymbol{\phi}}(\bullet) = \left[\widetilde{S}^{\phi^i}(\bullet)\right]_{i=1}^{N_\phi}$ has a functional form completely unknown, and is parameterized using N_γ stochastic expansion coefficients $\boldsymbol{\gamma}(\omega) = \left[\gamma^k\right]_{k=1}^{N_\gamma}$. The DO decomposition for the biogeochemical fields into mean $\bar{\boldsymbol{\phi}}$, modes $\tilde{\boldsymbol{\phi}}_i$, and stochastic coefficients Y_i, is given by,

$$\boldsymbol{\phi}(\boldsymbol{x}, t; \omega) = \bar{\boldsymbol{\phi}}(\boldsymbol{x}, t) + \sum_{i=1}^{N_s} \tilde{\boldsymbol{\phi}}_i(\boldsymbol{x}, t) Y_i(t; \omega)\,. \tag{2.22}$$

The uncertain regular and special stochastic parameters are split into means and deviations, $\boldsymbol{\theta}(\omega) = \bar{\boldsymbol{\theta}} + \boldsymbol{\mathfrak{D}}^{\theta}(\omega)$, $\boldsymbol{\alpha}(\omega) = \bar{\boldsymbol{\alpha}} + \boldsymbol{\mathfrak{D}}^{\alpha}(\omega)$, and $\boldsymbol{\beta}(\omega) = \bar{\boldsymbol{\beta}} + \boldsymbol{\mathfrak{D}}^{\beta}(\omega)$. For the nonlinear reaction terms in $\boldsymbol{S}^{\boldsymbol{\phi}}(\bullet)$ and $\widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}(\bullet)$, as for the nonlinear path planning optimal propulsion term [68, 69], we utilize a local Taylor series expansion around the statistical means, $\bar{\boldsymbol{\phi}}(\boldsymbol{x}, t)$, $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\alpha}}$, and $\bar{\boldsymbol{\beta}}$, to locally represent the nonlinear stochastic effects in the reaction equations as nonlinear mean terms plus stochastic deviations. As we will exemplify, for most uncertainties, such stochastic approximation is efficient for Bayesian learning as it maintains the significant computational advantages of the DO methodology (Appendix A) with respect to the other methods [70]. Handling the $\widetilde{\boldsymbol{S}}[\bullet]$ term is less straightforward because of the need to evaluate the interval in which each state realization value lies at every spatial location and time (see Sect. 2.2.2). Presently, for maximum accuracy, we directly evaluate the $\widetilde{\boldsymbol{S}}[\bullet]$ term for every state realization in a Monte-Carlo fashion. To increase efficiency without much loss

of accuracy, very recent techniques such as dynamic clustering [71, 72, 73] Next, we directly provide the resulting DO equations for the mean, modes, and stochastic coefficients (omitting function arguments and now using i, j, n, and m as summation indices),

$$
\begin{aligned}
\frac{\partial \bar{\boldsymbol{\phi}}}{\partial t} &= -\nabla.(\boldsymbol{u}\bar{\boldsymbol{\phi}}) + \mathcal{K}_E \nabla^2 \bar{\boldsymbol{\phi}} + \boldsymbol{S}^{\boldsymbol{\phi}}|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \mathbb{E}[\widetilde{\boldsymbol{S}}^{\boldsymbol{\phi}}]\,, \\
\frac{\partial \tilde{\boldsymbol{\phi}}_i}{\partial t} &= \boldsymbol{Q}_i - \sum_{j=1}^{N_s} \langle \boldsymbol{Q}_i, \tilde{\boldsymbol{\phi}}_j \rangle \tilde{\boldsymbol{\phi}}_j\,, \\
\frac{dY_i}{dt} &= \sum_{m=1}^{N_s} \langle \boldsymbol{F}_m, \tilde{\boldsymbol{\phi}}_i \rangle Y_m + \sum_{m=1}^{N_\theta} \left\langle \frac{\partial \boldsymbol{S}^{\boldsymbol{\phi}}}{\partial \theta_i}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}, \tilde{\boldsymbol{\phi}}_i \right\rangle \mathfrak{D}_m^{\theta} + \sum_{m=1}^{N_\beta} \left\langle \frac{\partial \boldsymbol{S}^{\boldsymbol{\phi}}}{\partial \beta}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}, \tilde{\boldsymbol{\phi}}_i \right\rangle \mathfrak{D}_m^{\beta} \\
&\quad + \sum_{m=1}^{N_\theta} \left\langle \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \theta_i}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}, \tilde{\boldsymbol{\phi}}_i \right\rangle \mathfrak{D}_m^{\theta} + \sum_{m=1}^{N_\alpha} \left\langle \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \alpha_i}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}, \tilde{\boldsymbol{\phi}}_i \right\rangle \mathfrak{D}_m^{\alpha} \\
&\quad + \sum_{m=1}^{N_\beta} \left\langle \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \beta_i}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}, \tilde{\boldsymbol{\phi}}_i \right\rangle \mathfrak{D}_m^{\beta} + \left\langle \widetilde{\boldsymbol{S}}^{\boldsymbol{\phi}} - \mathbb{E}[\widetilde{\boldsymbol{S}}^{\boldsymbol{\phi}}], \tilde{\boldsymbol{\phi}}_i \right\rangle\,,
\end{aligned}
\tag{2.23}
$$

where,

$$
\begin{aligned}
\boldsymbol{Q}_i = &-\nabla.(\boldsymbol{u}\tilde{\boldsymbol{\phi}}_i) + \mathcal{K}_E\nabla^2\tilde{\boldsymbol{\phi}}_i + \frac{\partial \boldsymbol{S}^{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_i + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\theta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}_n^\theta Y_j} \frac{\partial \boldsymbol{S}^{\boldsymbol{\phi}}}{\partial \theta_n}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \\
&+ \sum_{j=1}^{N_s}\sum_{n=1}^{N_\beta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}_n^\beta Y_j} \frac{\partial \boldsymbol{S}^{\boldsymbol{\phi}}}{\partial \beta_n}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_i + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\theta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}_n^\theta Y_j} \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \theta_n}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \\
&+ \sum_{j=1}^{N_s}\sum_{n=1}^{N_\alpha} C^{-1}_{Y_iY_j} C_{\mathfrak{D}_n^\alpha Y_j} \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \alpha_n}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\beta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}_n^\beta Y_j} \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \beta_n}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \sum_{j=1}^{N_s} C^{-1}_{Y_iY_j}\mathbb{E}[Y_j\widetilde{\boldsymbol{S}}^{\boldsymbol{\phi}}]\,, \\
\boldsymbol{F}_m = &-\nabla.(\boldsymbol{u}\tilde{\boldsymbol{\phi}}_m) + \mathcal{K}_E\nabla^2\tilde{\boldsymbol{\phi}}_m + \frac{\partial \boldsymbol{S}^{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_m + \frac{\partial \widehat{\boldsymbol{S}}^{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}}\bigg|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_m\,,
\end{aligned}
\tag{2.24}
$$

with $C_{\bullet,\bullet}$ representing cross-covariances, $\mathbb{E}[\bullet]$ expectations, and $\langle\bullet,\bullet\rangle$ spatial inner-products operators.

2.3.4 Modeling Domain and Boundary Conditions

Our modeling domain is inspired from the Stellwagen Bank at the edge of Massachusetts Bay, which is a whale feeding ground [74, 75, 62, 76, 77, 78, 79]. The experimental setup consists of a two-dimensional domain with a seamount representing an idealized sill (Fig. 2-1). The mean flow occurs from left to right in the positive x-direction over the seamount. Such flows can create upwelling of nutrients, leading to phytoplankton blooms, zooplankton responses, and nutrient uptake and recycling.

A horizontal length scale of $D \approx 1\ km$ is chosen for the seamount, while the vertical height scale is $H \approx 50\ m$. The overall transverse height of the domain is $H_{in} = 100\ m$. The longitudinal length of the domain is $L = 20\ km$, with center of the seamount at $X_c = 7.5\ km$.

Further, we only consider deterministic boundary conditions (BCs) models. The inlet at the left boundary has Dirichlet BCs for velocity, and zero Neumann for

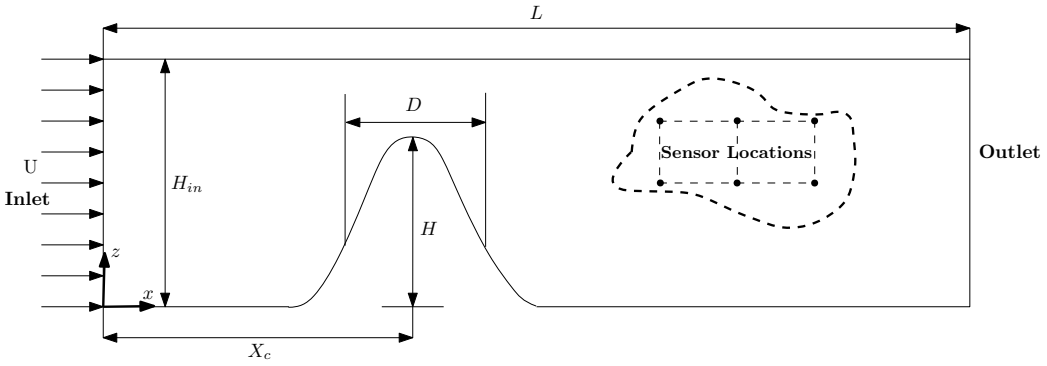


Figure 2-1: Two-dimensional spatial domain of the flow past a seamount. The seamount is defined by $He^{-(x-X_c)^2/D^2}$, where D is the characteristic width, H the height, and X_c the distance between the inlet and the center of the seamount. Observations are collected downstream of the seamount (see example sensor locations inset), with the exact observation locations depending on the particular experiment.

biological tracers,

$$u = U,\ v = 0 \qquad \text{and } \frac{\partial \boldsymbol{\phi}^i}{\partial x} = 0, \qquad \text{at } x = 0\ ,\ \forall i \in \{1, ..., N_\phi\}\ . \tag{2.25}$$

On the top and bottom boundary, free slip for velocity and again zero Neumann for tracers are applied,

$$\frac{\partial u}{\partial z} = 0,\ v = 0 \text{ and } \frac{\partial \boldsymbol{\phi}^i}{\partial z} = 0, \qquad \text{at } z = 0 \ \&\ h\ ,\ \forall i \in \{1, ..., N_\phi\}\ . \tag{2.26}$$

At the outlet on the right boundary, we have open BCs with zero Neumann for all the state variables,

$$\frac{\partial u}{\partial x} = 0,\ \frac{\partial v}{\partial x} = 0 \qquad \text{and} \qquad \frac{\partial \boldsymbol{\phi}^i}{\partial x} = 0, \qquad \text{at } x = L\ ,\ \forall i \in \{1, ..., N_\phi\}\ . \tag{2.27}$$

Finally, on the obstacle surface, no-slip for velocity and zero Neumann for tracers are used,

$$u = 0,\ v = 0 \qquad \text{and} \qquad \frac{\partial \boldsymbol{\phi}^i}{\partial x} = \frac{\partial \boldsymbol{\phi}^i}{\partial z} = 0, \qquad \text{at } z = He^{-(x-X_c)^2/D^2}\ ,\ \forall i \in \{1, ..., N_\phi\}\ . \tag{2.28}$$

2.3.5 Numerical Schemes

The velocity and pressure fields are governed by the incompressible nonhydrostatic RANS PDEs (2.19). The stochastic biogeochemical fields are coupled with this dynamic RANS flow and governed by a dynamic reduced-order representation of the original stochastic ADR PDEs (2.20), the DO ADR PDEs we derived (Eqs. 2.23 & 2.24). For numerical implementation, the physical domain (Sect. 2.3.4) is discretized using a uniform finite-volume staggered C-grid, for both the flow and stochastic biogeochemical fields. The size of finite volumes in each $x-$ and $z-$ direction is equal to $\Delta x = \frac{1}{15}$ and $\Delta z = \frac{1}{15}$ (non-dimensional) respectively, thus, a grid-size of 300×30.

We solve the RANS and biogeochemical DO equations using our modular finite-volume framework [80]. Advection is computed explicitly, using a total variation diminishing (TVD) scheme with a monotonized flux limiter [81]. Diffusion is treated implicitly, with a second-order central difference scheme. All the reaction terms are computed explicitly. To handle the complex boundaries with the structured Cartesian grid, a ghost cell immersed boundary method is adopted for accurate enforcement of the boundary conditions (Sect. 5.2). For time-marching of the PDEs (RANS, DO mean, and DO modes), we use a first-order forward Euler method, while for the stochastic DO coefficient ODEs, we use a four-stage Runge-Kutta scheme. A non-dimensional time-step of $\Delta t = \frac{1}{240}$ is used in all the experiments. It is also ensured that we satisfy the Courant-Friedrichs-Lewy (CFL) condition at all times. We refer to ([82]) and ([14]) for more details on the numerical schemes we employ.

2.3.6 Balanced Initialization: Parameters, State Variable Fields, and Probabilities

The values of the parameters for the physics are chosen such that the flow emulates some coastal ocean dynamics. The dimensional barotropic velocity at the inlet is chosen to be $U \approx 10^{-2}$ to 10^{-1} m/s. The subgridscale eddy-viscosity is $\nu_E \approx 0.01$ to 0.5 m^2/s. Considering the vertical length scale of $H \approx 50$ m for the seamount, we obtain an eddy-viscosity Reynolds number of $Re = \frac{UH}{\nu_E} \approx 1$ to 500. Further,

we do not consider any wind-forcing explicitly. For the initial velocity, we use a divergence free velocity field that satisfies the inlet and outlet boundary conditions, and so mass conservation in the given domain. The pressure field is initialized to be zero throughout the domain.

The biological parameters are either deterministic or stochastic. The values of the deterministic parameters are kept fixed for every realization, while the stochastic parameters are sampled from their respective probability distributions or joint distributions, if available. The stochastic parameters are further divided into two categories, regular and special, where the former were originally present in the biogeochemical models and have biological meanings associated with them, while the later are introduced for unification of candidate models and parameterization of unknown functions. The values of biological parameters used in the main experiments are given in Table 2.1. Probability distributions of all the stochastic parameters are assumed to be uniform and independent of each other, unless otherwise specified. In the experiments presented in this paper, advection-reaction dominates and the eddy-diffusivity for the biological tracers can be taken as negligible, $\mathcal{K}_E \approx 0$, such that the eddy-diffusivity Peclet number $Pe = \frac{UH}{\mathcal{K}_E} \to \infty$. Other experiments (not shown) were also successful however with non-negligible diffusivity, e.g. [8]. In all our simulations, a biological time-scale of the order of 1 *day* is used for all non-dimensionalization purposes.

Following [83, 62, 61, 84], in all the subsequent experiments, biogeochemical fields are initialized in dynamical balance, in accord with their stochastic model PDEs (2.20) and their parameter values. Specifically, the initial concentration fields for every sampled realization is obtained by finding an equilibrium solution corresponding to its sampled parameter values. These equilibrium fields are found by solving the ODE nonlinear biogeochemical models of Sect. 2.3.1 at all depths. Equilibrium is reached when temporal variations become negligible, or the system reaches a limit cycle. Further, we also impose the total biomass, $\sum_{i=1}^{N_\phi} \phi^i(z;\omega) = T_{bio}(z)$, to be conserved, with T_{bio} to be linearly increasing from 10 $mmol\ N\ m^{-3}$ at the surface to 30 $mmol\ N\ m^{-3}$ at the depth of 100 m, for all the biogeochemical models. This

depth-dependent equilibrium solution for each of the biogeochemical state variables is used to initialize the corresponding fields in space, with the seamount masked at every x location. We also ensure that none of the realizations of the stochastic parameters lead to nonphysical equilibrium solutions, such as negative tracer values. The value of 30 $mmol\ N\ m^{-3}$ is used to non-dimensionalize all the biogeochemical fields and parameter values. For the non-dimensionalization of parameters, when needed, we additionally use a length-scale of 50 m (the height H of the seamount) and a time-scale of 1 day.

To initialize the DO decomposition of the biogeochemical fields, after generating the initial fields for each realization, we compute their statistical average and use it to initialize the mean biogeochemical fields. To initialize the DO modes and stochastic coefficients, we take the singular value decomposition (SVD) of the ensemble of mean-removed concatenated fields, keeping the dominant singular values and vectors. We account for the differences in the magnitude of the variability of individual biogeochemical tracers before taking the SVD, by appropriate normalization based on their standard deviations.

2.3.7 True Solution Generation

In the present work, identical-twin experiments [85, 86, 87, 88] are conducted, and the observations are extracted from a simulated truth. To obtain the simulated truth fields for each experiment, a set of parameters and state fields are first sampled from the initial realization space. Second, starting from these parameters and state fields, the Navier-Stokes PDEs (2.19) and the deterministic version of the ADR PDEs (2.20) with the appropriate biogeochemical model are numerically integrated. The result is the simulated truth solution. In each experiment, all the remaining deterministic parameters, modeling domain, and numerical schemes are kept as these of the corresponding stochastic simulation using the DO equations.

2.3.8 Observations and Inference

Sparse observations are taken from the simulated true solution (Sect. 2.3.7). In each experiment, one of the biological tracer fields is observed at 6 to 9 locations (Fig. 2-1). The observation locations are kept in or near the euphotic zone because deeper depths have limited biological variability. The observation schedule is also experiment dependent, however, is it is not more frequent than once every non-dimensional time.

The observation error standard deviation matrix ($\sqrt{\boldsymbol{R}}$ in Eq. 2.3) is assumed diagonal. The linear observation matrix $\boldsymbol{H}$ (Eq. 2.3) is specified such that it predicts the concentration of the observed tracer field at the observation locations by interpolating the concatenated state fields at the observation locations.

Further, the hyper-parameters related to the DO equations and the GMM-DO filter were chosen based on numerical tests and experience [17, 89, 10, 90], for each of the experiments. For the DO equations, for example, the number of modes, of Monte-Carlo coefficient samples, the time-step, etc., were selected so as to be sufficient to capture the dominant uncertainty and evolving probability distribution for each of the state vector fields, parameters, and model equations themselves. For the Bayesian learning with the GMM-DO filter, the BIC and EM algorithm were employed to select the optimal number of GMM components at each data time. Typical BIC-optimized values for N_{GMM} were found to be 10 for the present experiments.

2.3.9 Learning Metrics

We evaluate the performance of our Bayesian learning framework by comparing the learned solution with the true solution from which noisy observations were collected and by examining the posterior joint state-parameter-model probability distributions. For the former solution evaluations, we compare the true fields to the DO mean fields, and the true parameter values to the most probable DO pdf values of the parameters. To quantify performance, we examine the evolution of the Root Mean Square Error (RMSE) of the biogeochemical tracer fields, the uncertain regular ($\boldsymbol{\theta}(\omega)$) and special ($\boldsymbol{\alpha}(\omega)$ & $\boldsymbol{\beta}(\omega)$) parameters, and/or the stochastic expansion

coefficients ($\boldsymbol{\gamma}(\omega)$). The RMSE between a evolved stochastic state field/parameter estimate $\phi(\boldsymbol{x}, t; \omega)$ and its corresponding true field/parameter $\phi^{true}(\boldsymbol{x}, t)$, is given by, $\sqrt{\frac{1}{|\mathcal{D}|}\int_{\mathcal{D}} \mathbb{E}[(\phi(\boldsymbol{x}, t; \omega) - \phi^{true}(\boldsymbol{x}, t))^2] d\boldsymbol{x}}$. The square of RMSE hence consists of two contributions [9], one is the square of the L_2 distance between the mean of the variable in the stochastic run and the simulated truth, while other is the variance of the variable. In every experiment, the RMSE values of each variable are normalized by the corresponding RMSE value just before the first assimilation step. For the latter pdf evaluations, we analyze the evolution of the posterior pdfs of the stochastic DO coefficients, and of the regular and special stochastic parameters. For example, for the DO coefficient realizations, we employ 2-D scatter plots. For the stochastic parameters, we use marginals and kernel-density fits. We also evaluate the convergence of pdf estimates with stochastic resolution, i.e. increasing/decreasing stochastic numerical parameters (N_s, N_r, etc.), see Sect. 2.2.4.

2.4 Application Results and Discussion

In order to demonstrate the capabilities of our Bayesian learning we utilize four sets of identical twin experiments with different coupled biogeochemical-physical dynamics and learning objectives, and perform simultaneous Bayesian estimation of state variables, parameters, and model equations, using observations that are sparse in both space and time. To quantify performance, we evaluate several learning metrics, emphasizing the sharpness of the inference and the accuracy of probability distributions. For each of the four sets of experiments, we conduct multiple studies so as to evaluate the sensitivity to hyper-parameters. However, for each set, we present detailed results for only one experiment and summarize the other results.

2.4.1 Experiments 1: Discriminating among candidate functional forms and smoothing

Biologically, mortality is a linear rate process. The mortality terms of phytoplankton and zooplankton however commonly act as "closure" parameterizations in models because as they allow for recycling of nutrients directly from plankton. As a result, due to the missing intermediate states in the recycling model, the zooplankton mortality and recycling processes are often modeled nonlinearly, with a concentration-dependent loss rate [40]. In this first set of experiments, we use the NPZ model with uncertainty introduced by the ambiguity in the presence or absence of a quadratic zooplankton mortality function, along with the uncertainty in the value of the Ivlev grazing parameter (Λ). The uncertainty in the initial biogeochemical conditions is set in balance with the uncertain parameters and model equations, as explained in Sect. 2.3.6. The learning objective is to simultaneously learn all the biological states, regular parameter Λ, and functional form of zooplankton mortality using a special stochastic parameter, assimilating sparse observations.

The right-hand-side of NPZ model (Eq. 6.26) with the quadratic zooplankton mortality is given by,

$$\begin{aligned} S^N &= -G\frac{PN}{N+K_u} + \Xi P + \Gamma Z + \underbrace{\alpha(\omega)(\tilde{\Gamma} Z^2)}_{\text{Quad. Z Mort.}} + R_m \gamma Z(1-\exp^{-\Lambda(\omega)P}) \\ S^P &= G\frac{PN}{N+K_u} - \Xi P - R_m Z(1-\exp^{-\Lambda(\omega)P}) \\ S^Z &= R_m(1-\gamma)Z(1-\exp^{-\Lambda(\omega)P}) - \Gamma Z - \underbrace{\alpha(\omega)(\tilde{\Gamma} Z^2)}_{\text{Quad. Z Mort.}} . \end{aligned} \tag{2.29}$$

The stochastic parameters are explicitly shown using the realization index (ω), and the ambiguous quadratic mortality term is pointed out. The special stochastic parameter, $\alpha(\omega)$, is restricted to binary values, i.e., either 0 or 1, corresponding to the absence or presence of the quadratic mortality term, respectively. $\Lambda(\omega)$ is sampled from a uniform probability distribution between the non-dimensional values of 3 and 6, and $\alpha(\omega)$ is assumed to have an initial 50%-50% probability of being 0 or 1. The above

stochastic NPZ reactions (Eq. 2.29) are coupled with the RANS flow PDEs and used in the stochastic ADR PDEs that are solved with the DO methodology (Sect. 2.3.3). The other known model parameters related to the physical-biogeochemical model as well as the hyper-parameters for the DO equations are provided in Table 2.1.

True solution generation: The true solution from which observations are extracted, corresponds to the non-dimensional values, 3.6 for Λ, and 1 for α, i.e., the quadratic mortality term present. The true state fields are initialized and evolved as described in Sect. 2.3.7. **Observations and learning parameters:** The observations are sparse in both space and time, and consist of zooplankton measurements at six locations downstream of the seamount, only at every two non-dimensional times, starting at $t = 5$. The data shown in Fig. 2-2 is all that the Bayesian learning framework gets to assimilate over the course of the experiment. Other hyper-parameters related to the GMM-DO filtering are provided in Table 2.1. **Numerical method:** The deterministic equations for the true solution and DO equations for the estimate pdfs are solved using the modular finite-volume framework described in Sect. 2.3.5. **Learning metrics:** As time advances, the sparse data are assimilated using the Bayesian GMM-DO filter in the augmented state space. We compare the true fields and paramaters to their DO estimates (mean and most probable values). To quantify performance, we examine the evolution of the normalized RMSEs (Sect. 2.3.9) for the N, P, and Z fields, and for the $\Lambda(\omega)$ and $\alpha(\omega)$ parameters, as well as the pdfs of the stochastic parameters, DO coefficients, and biological states.

Table 2.1: Values of the various domain-related, biological, physical, and hyper-parameters used in the four sets of experiments. $H = 50\ m$, $max\{T_{bio}(z)\} = 30\ mmol\ N\ m^{-3}$, and time-scale of 1 day, are the characteristic scales used for non-dimensionalization.

Parameters	**Exp. 1**	**Exp. 2**	**Exp. 3**	**Exp. 4**
Biogeochemical model	NPZ	NPZ & NPZD	NPZ	NNPZD
Biological Parameters				

Light attenuation due to sea water, k_w (m^{-1})	0.067	0.067	0.067	0.067
Initial slope of the P-I curve, α $((W\ m^{-2}\ day)^{-1})$	0.025	0.025	0.025	0.025
Surface photosynthetically available radiation, I_o $(W\ m^{-2})$	158.075	158.075	158.075	158.075
Phytoplankton maximum uptake rate, V_m (day^{-1})	1.5	1.5	1.5	1.5
Half-saturation for phytoplankton uptake of nutrients, K_u^* $(mmol\ N\ m^{-3})$	1	1	1	1
NH_4 inhibition parameter, Ψ_I $((mmol\ N\ m^{-3})^{-1})$	–	–	–	1.46
NH_4 oxidation coefficient, Ω (day^{-1})	–	–	–	0.25
Phytoplankton specific mortality rate, Ξ (day^{-1})	0.1	0.1	0.1	unif(0.01, 0.08)
Zooplankton specific excretion and mortality rate, Γ (day^{-1})	0.145	0.145	0.145	unif(0.125, 0.150)
Presence/absence of quadratic zooplankton term, α	unif{0, 1}	unif{0, 1}	–	unif{0, 1}
Quadratic zooplankton specific excretion and mortality rate, $\tilde{\Gamma}$ (day^{-1})	0.2	0.2	0.2	0.2
Zooplankton maximum grazing rate, R_m (day^{-1})	0.52	0.52	0.52	unif(0.52, 0.72)
Ivlev constant, Λ $((mmol\ N\ m^{-3})^{-1})$	unif(0.1, 0.2)	unif(0.1, 0.2)	0.13	unif(0.052, 0.072)

Fraction of zooplankton grazing egested, γ	0.3	0.3	0.2	0.3
Detritus decomposition rate, Φ (day^{-1})	1.03	1.03	1.03	1.03
Diffusion constants – horizontal & vertical, $(\mathcal{K}_E)$	0	0	0	0
Modeling Domain				
Height of the seamount, H (m)	50	50	50	50
Characteristic width of the seamount, D (km)	1	1	1	1
Distance between inlet and center of seamount, X_c (km)	7.5	7.5	7.5	7.5
Domain height, H_{in} (m)	100	100	100	100
Domain length, L (km)	20	20	20	20
Physical Parameters				
Inverse of Eddy-viscosity Reynolds nb., (Λ_{Re})	1	1	1	1/500
DO Parameters				
Number of Modes, N_s	20	40	20	15
Number of Monte-Carlo samples, N_r	10,000	10,000	1,000	10,000
GMM-DO Parameters				
State being observed	Z	Z	N	P
Observation error standard deviation, $(\sqrt{\boldsymbol{R}})$	0.05	0.05	0.035	0.04
Size of Observation vector, N_y	6	6	8	9
Observation start time (non-dim.)	5	5	1	2

Time interval between assimilations (non-dim.)	2	2	2	1
Observation end time (non-dim.)	25	25	25	25

Learning results

Figure 2-3 shows the initial state and parameters of the system (at $t = 0$), while Fig. 2-4 shows the evolved prior state and parameters of the system at $t = 5$ (i.e. just before the 1st observational episode). There are significant differences between the true and prior DO mean fields of the biogeochemical tracers. During these first five non-dimensional time units, a phytoplankton bloom develops just downstream (top-right) of the seamount: upwelling of nutrients above the seamount within the euphotic zone feeds the growth in phytoplankton biomass in the wake.

In Fig. 2-5, we illustrate the evolving statistics of the stochastic dynamical system from $t = 0$ to $t = 5$ just before data assimilation. We show fields of the phytoplankton standard deviation and dominant three DO modes (Panels 2-5a & 2-5b). The standard deviation fields clearly highlight the significant uncertainty around the phytoplankton subsurface maxima and bloom, reaching 30 percent of the mean field maxima. The uncertain subsurface maxima and bloom also clearly affect the DO modes. In Panels 2-5c & 2-5d, we show the joint distribution of the top four stochastic coefficients, along with the prior GMM fits using 10 components (Panel 2-5d). We use the Bayesian information criterion (BIC; [91]) to find the optimal number of components required [16]. The joint distributions demonstrate the highly non-Gaussian nature of the stochastic DO coefficients, which the DO equations are able to evolve, and the GMM-DO filter is able account for. The strong parametric uncertainties is reflected by the thin 2D joint coefficient distributions. In addition, the realizations of the stochastic coefficients are clearly divided into two groups, each corresponding to the presence and absence of the quadratic mortality term, .

At $t = 5$, the first sparse data is assimilated. Fig. 2-6 shows the posterior mean fields, prior and posterior parametric distributions, and the normalized RMSE values

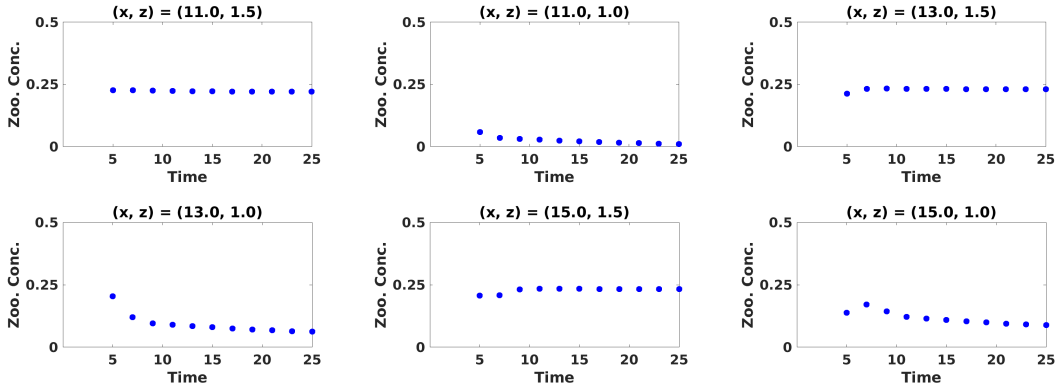


Figure 2-2: Experiments-1. Time-series of zooplankton data collected at six observation locations (with coordinates given in the respective titles).

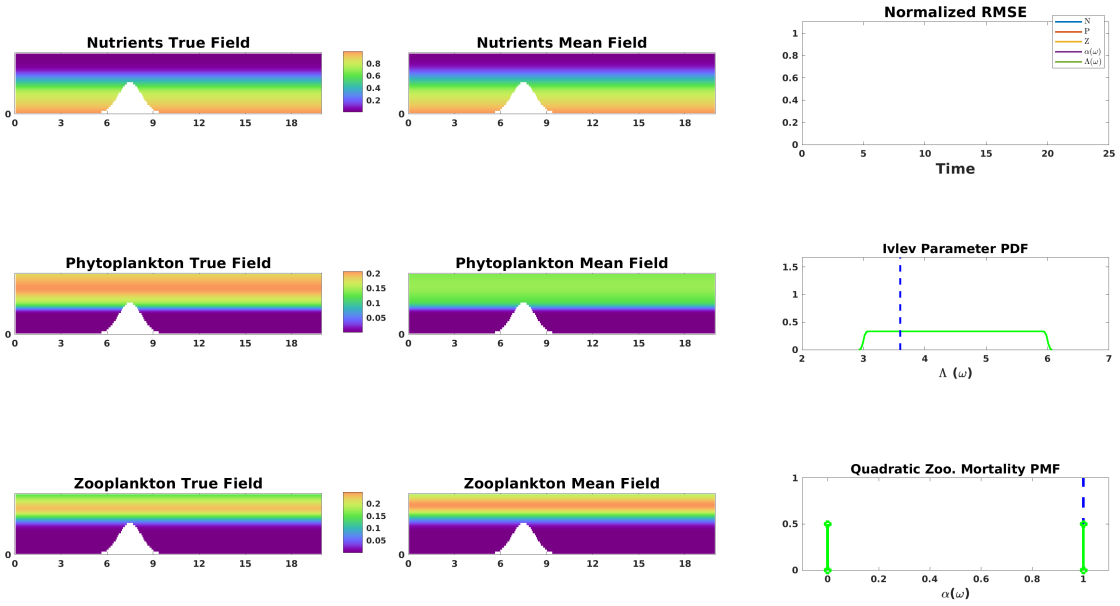


Figure 2-3: Experiments-1: State of the true and estimate NPZ fields and parameters at $t = 0$ (i.e. initial conditions). The first two columns consist of the non-dimensionalized true (left) and estimate mean (right) tracer fields of N, P and Z. In the third column, the top panel shows the variation of normalized root-mean-square-error (RMSE) with time for the stochastic state variables and parameters. The next two panels contain the pdf of the non-dimensional $\Lambda(\omega)$ and $a(\omega)$ (to learn presence or absence of quadratic zooplankton mortality), with their true unknown values marked with blue dotted lines. The velocity field is deterministic with $Re = 1$.

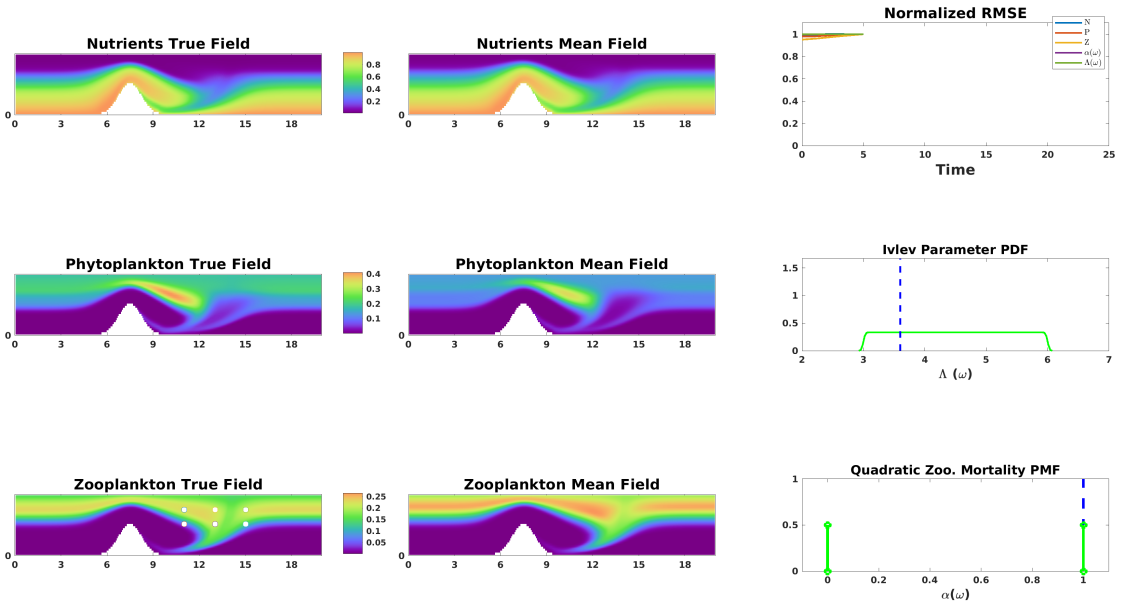
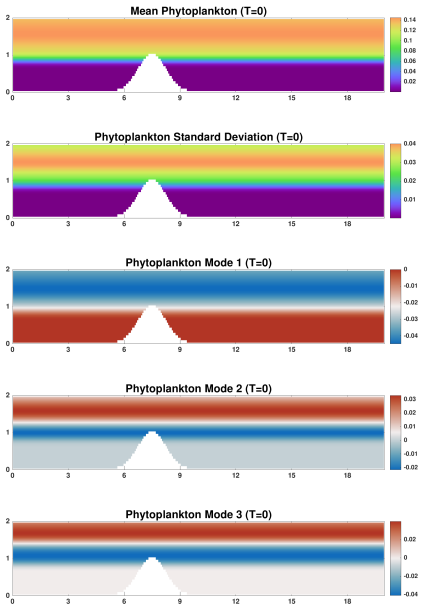


Figure 2-4: Experiments-1: As Fig. 2-3, but for the prior fields and parameters at $t = 5$ (i.e. just before the 1st assimilation). Additionally, the white circles on the zooplankton true field mark the six observation locations.

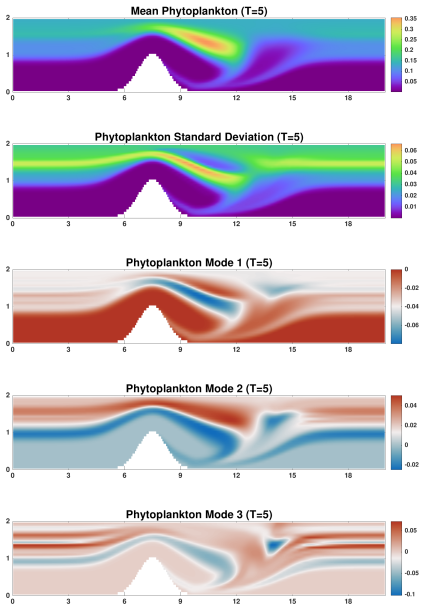
for the mean fields and two stochastic parameters. By just observing zooplankton at six locations, the GMM-DO filter simultaneously update all the biological fields and parameters. This is evident from the mean fields getting aligned with the true fields and quantified by the RMSE reductions of about 20 to 30 percent. Also visible is the slight change in the pdf for $\Lambda(\omega)$ and a higher probability value for $\alpha(\omega)$ being one. The six data are so far much more informative about the mortality term than about the Ivlev parameter.

Next, in Fig. 2-7, we illustrate the same posterior mean fields, prior and posterior parameters, and normalized RMSE values, but at $t = 15$, i.e., at the sixth data assimilation. The flow is fully developed with the biogeochemical fields well learned, as quantified by the normalized RMSEs at about 40 percent. The GMM-DO filter unambiguously detects the presence of quadratic mortality of Z. The pdf of $\Lambda(\omega)$ is also accumulated around its true value, but is multi-modal, indicating nonlinearities and remaining ambiguity.

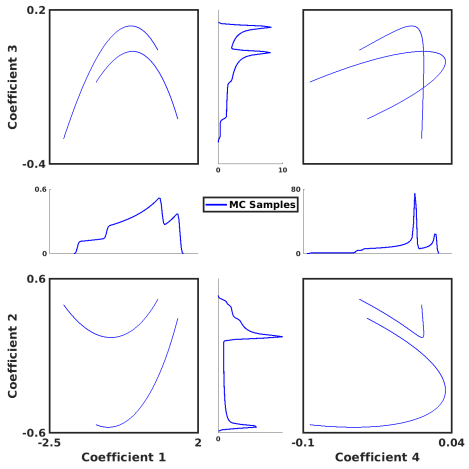
Finally, at $t = 25$, after 11 assimilation events, the same quantities are shown in



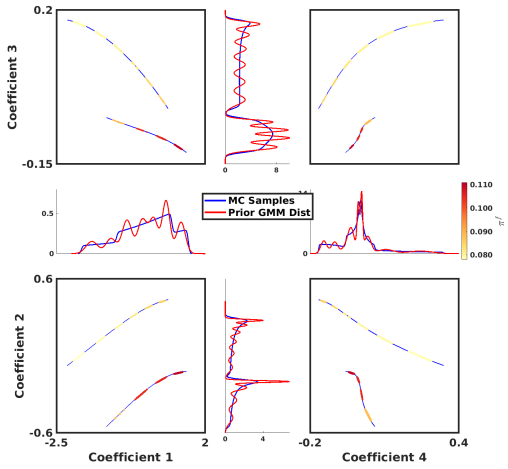
(a) Phytoplankton mean, standard deviation and top three DO modes, at $t = 0$



(b) Phytoplankton mean, standard deviation and top three DO modes, at $t = 5$ (prior)



(c) Joint distributions and respective marginals of the top four stochastic DO coefficients, at $t = 0$



(d) Joint distributions and respective marginals of the top four stochastic DO coefficients, along with the GMM fit, at $t = 5$ (prior)

Figure 2-5: Experiments-1: Statistics for the initial ($t = 0$) and prior ($t = 5$, just before the 1st assimilation) states of the stochastic NPZ ADR dynamical system.

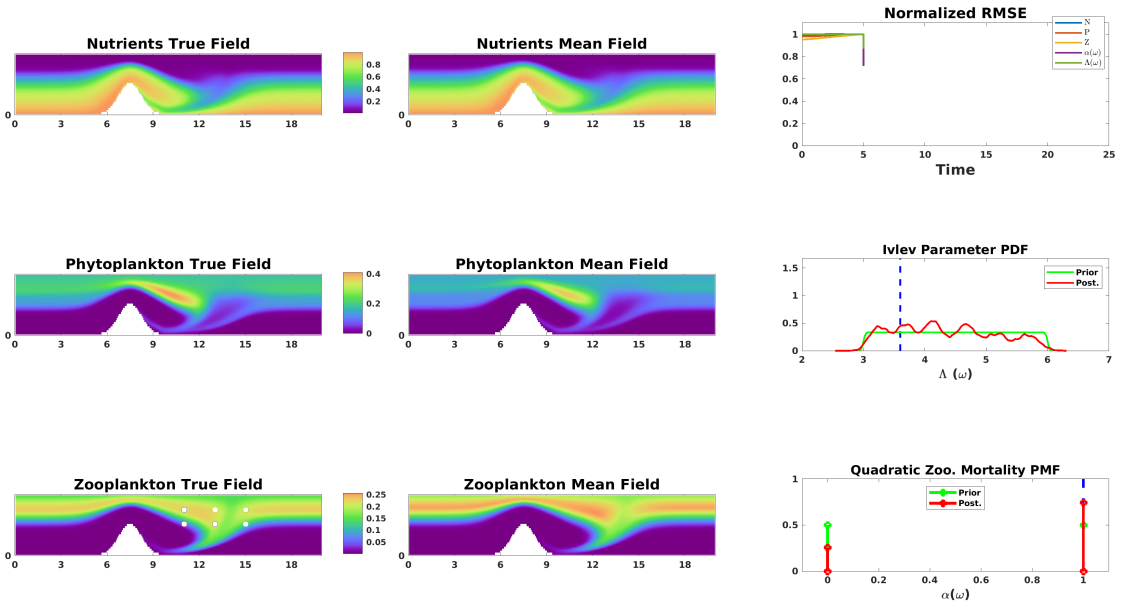


Figure 2-6: Experiments-1: As Figs. 2-3 & 2-4, but for posterior fields and parameters at $t = 5$ (i.e. just after the 1st assimilation).

Fig. 2-8. All the biogeochemical mean and true fields match with each other with RMSEs around 20 percent or less. The probability of the presence of the quadratic mortality term is now almost one, while the $\Lambda(\omega)$ pdf has a clear peak near 3.6 with a couple other much lower biased peaks around it. In general, the presence of lower peaks in pdfs of parameters indicate alternative combinations that could explain the data, and also the ability of the GMM-DO filter to capture non-Gaussian pdfs. The learning is also evident from the sustained decrease in the normalized RMSEs at every assimilation step for all the biogeochemical fields and parameters.

Many similar experiments were completed, changing various hyperparameters related to the GMM-DO filter, such as the biological variable being observed, observation locations, frequency, start-time, etc. Observations from simulated truths with different combinations of $\Lambda(\omega)$ and $\alpha(\omega)$ were also used. We found that the biological variable being observed has an impact on the sharpness of the inference or learnability of the given learning objectives. For example, observing N led to the learning of two distinct combinations of $\Lambda(\omega)$ & $\alpha(\omega)$, 3.1 & 0, and 3.6 & 1, respectively with

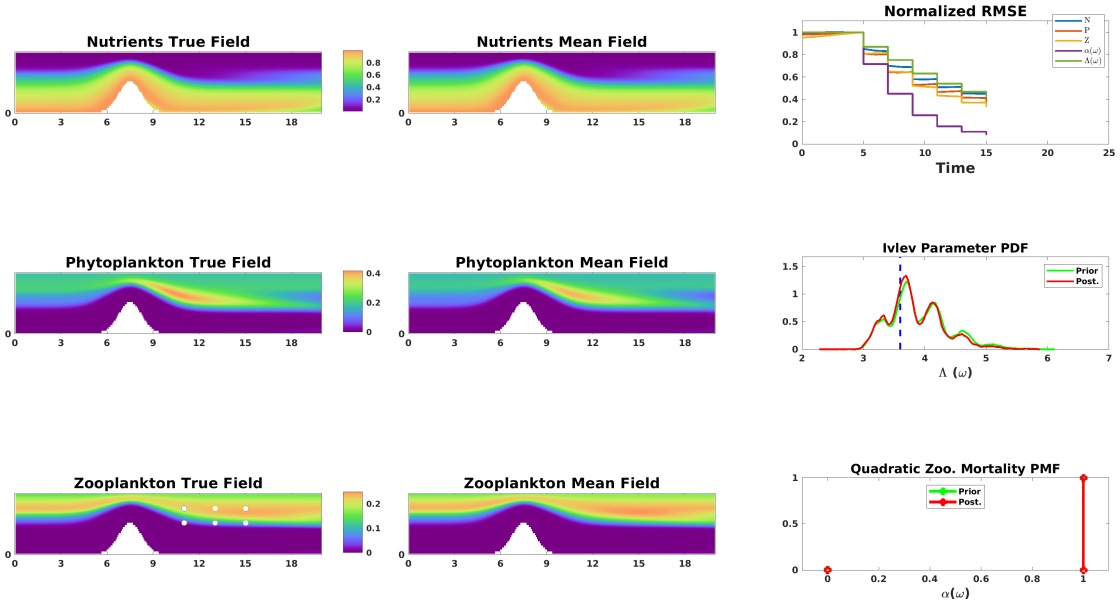


Figure 2-7: Experiments-1: As Figs. 2-3 & 2-4 but for posterior fields and parameters at $t = 15$ (i.e. just after the 6th assimilation).

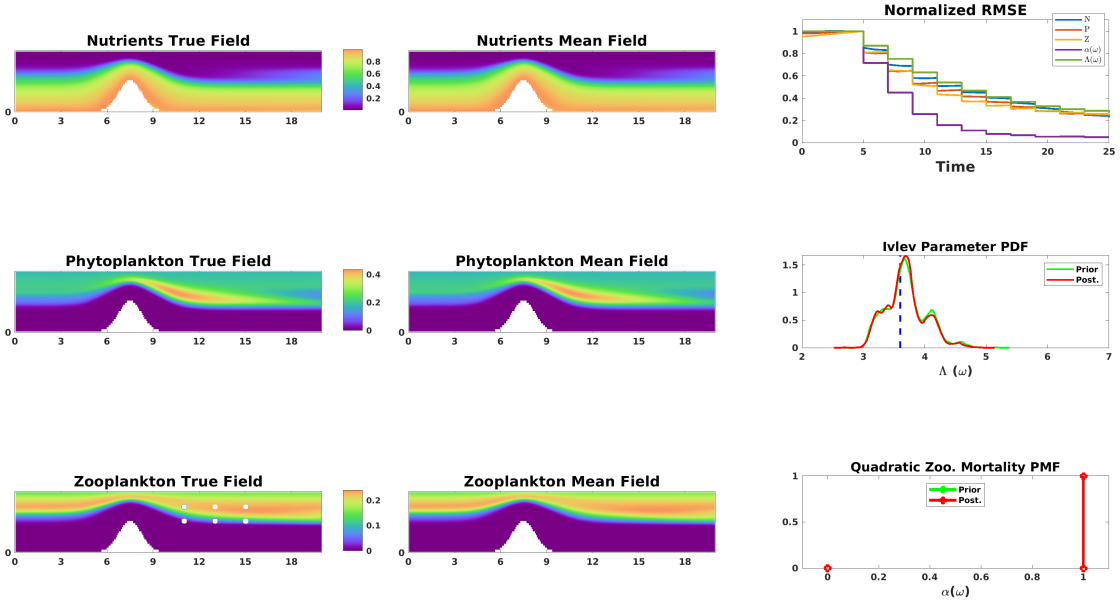


Figure 2-8: Experiments-1: As Figs. 2-3 & 2-4 but for posterior fields and parameters at $t = 25$ (i.e. just after the 11th assimilation).

nearly equal amount of confidence [92]. Decreasing the amount of observation data, or increasing the value of the observation error standard deviation led to larger uncertainty in the learned states and parameters. We also confirmed the convergence of our GMM-DO Bayesian posteriors by repeating learning experiments with an increasing number of DO modes and coefficients (not shown), until the results converged to those shown. This convergence of the pdfs of the parameters and DO coefficients, and of the DO modes and mean, indicates that our Bayesian GMM-DO filter provides accurate pdf estimates, and thus shows what has been learned without or with some ambiguity remaining. For the latter case, the multi-model posterior pdfs show that additional observations are needed to sharpen the inference further.

Once all the observations are collected, we also perform retrospective Bayesian smoothing [93] in order to maximize the information content of the posterior pdf of the state variables and parameters. We use the counterpart of the GMM-DO filter, called the GMM-DO smoother [94, 95], which enables non-Gaussian smoothing in an computationally efficient fashion by utilizing the DO subspace. In Fig. 2-9a, we provide the prior, posterior, and the smoothed stochastic coefficients for the DO subspace at time $t = 5$. It should be noted that the prior coefficients have seen no observational data, the posterior coefficients only contain information of the 1^{st} observational episode, while, the smoothed coefficients contains information of all the 11 observational episodes. The stochastic coefficient realizations are translated using the projection of their respective means onto the subspace to demonstrate the fact, that the pdfs of the posterior and the smoothed coefficients are contained within the support of pdf of the prior. Similar to the prior coefficients, the posterior coefficients are still divided into two groups because a single observational episode is insufficient to eliminate the ambiguity about the presence or absence of the quadratic mortality term. However, because the smoothed coefficients contain information from the future observational episodes, the ambiguity is resolved and the realizations are mostly part of one of the groups. Normalized RMSE of the smoothed states and parameters are provided in Fig. 2-9b, and one can notice that it is always lower than that of the corresponding filtered variables due to access to future observations in the smoothed

case.

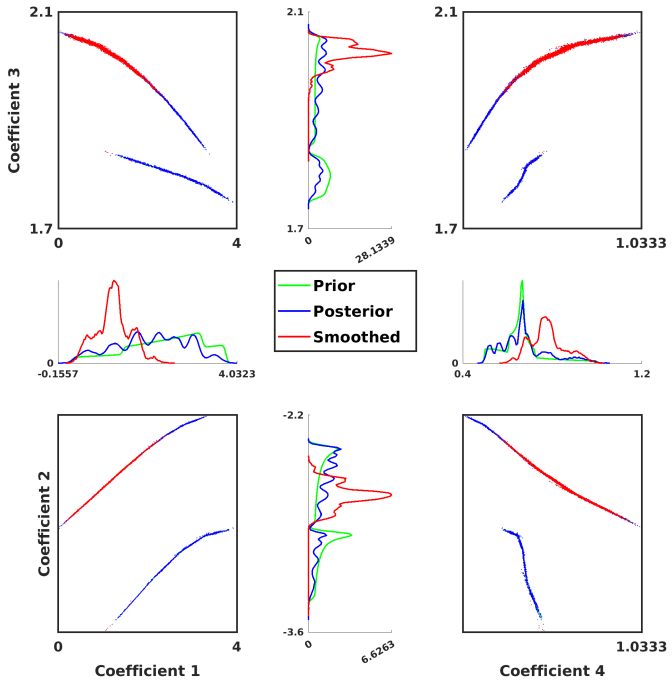
2.4.2 Experiments 2: Discriminating among models of different complexities

In the second set of experiments, the primary goal is to learn the complexity of the biogeochemical model, e.g., its state variables, along with the biogeochemical fields and Ivlev grazing parameter. Two candidates hierarchical model classes, NPZ and NPZD, are considered possible. To represent them with a single modeling system, we embed the former into the latter using our special stochastic parameter, $\beta(\omega)$. We multiply the detritus state variable (D) and other appropriate terms with $\beta(\omega)$, such that, the value of 1 derives the NPZD model, while the value of 0 derives the NPZ model (see Eq. 2.7). Thus, the RHS of the general stochastic model which encompasses both NPZ and NPZD models is given by,

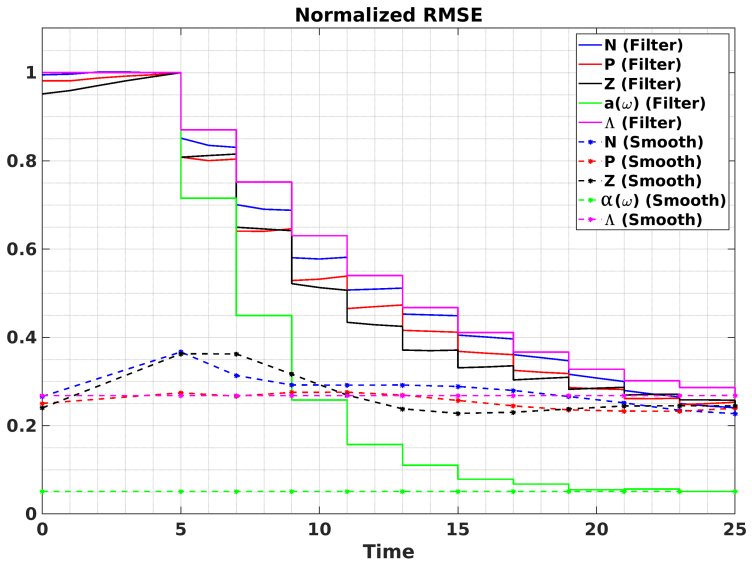
$$\begin{aligned}
S^N &= -G\frac{PN}{N+K_u} + \Phi D' + \Gamma Z + (1-\beta(\omega))\Xi P \\
S^P &= G\frac{PN}{N+K_u} - \Xi P - R_m Z(1-\exp^{-\Lambda(\omega)P}) \\
S^Z &= R_m(1-\beta(\omega)\gamma)Z(1-\exp^{-\Lambda(\omega)P}) - \Gamma Z \\
S^{D'} &= \beta(\omega)R_m\gamma Z(1-\exp^{-\Lambda(\omega)P}) + \beta(\omega)\Xi P - \Phi D' \\
D' &= \beta(\omega)D \ ,
\end{aligned} \tag{2.30}$$

where D' is the modified detritus state. Once again, $\Lambda(\omega)$ is sampled from a uniform probability distribution between the non-dimensional values of 3 and 6, and $\beta(\omega)$ is assumed to have 50%-50% probability of being 0 or 1. The stochastic NPZD reactions (Eq. 2.30) are coupled with the RANS flow PDEs and used in the stochastic ADR PDEs that are solved with the DO methodology (Sect. 2.3.3). The other known physical-biogeochemical parameters as well as the hyper-parameters for the DO equations are given in Table 2.1.

True solution generation: The true solution corresponds to the NPZ model with a non-dimensional value of 3.6 for the Λ parameter. The state fields are initialized



(a)



(b)

Figure 2-9: Results corresponding to the smoothing part of Experiments - 1. *(a):* Joint distributions (scatter-plots with realizations) of the top four DO stochastic coefficients for the prior, posterior, and the smoothed states at $t = 5$. Individual probability density plots (line-plots) are also provided in the middle of scatter-plots. Projection of the mean onto the subspace is added to each realization of the stochastic coefficients. The realization of the prior are completely covered by the posterior realizations, thus, not visible. *(b):* Variation of RMSE with time for all the stochastic state variables and parameters. The full-lines ('—') corresponds to the forward filtering pass, and the dashed-lines ('- - -') to the smoothing pass.

and evolved as described in Sect. 2.3.7. **Observations and learning parameters:** The observations are sparse in both space and time, and again consist of zooplankton measurements at six locations downstream of the seamount, only at every two non-dimensional times, starting at $t = 5$. Other hyper-parameters related to the GMM-DO filtering are provided in Table 2.1. **Numerical method:** Similar to the last set of experiments, the DO equations and the deterministic governing equations for the true solution are solved using the modular finite-volume framework described in Sect. 2.3.5. **Learning metrics:** As time advances and sparse data are assimilated, we compare the true fields and parameters their DO estimates. To quantify performance, we examine the evolution of the: normalized RMSEs of state fields and parameters, pdfs of stochastic parameters, and variances of DO coefficients.

Learning results

Figure 2-10 shows the state and parameters of the system at $t = 5$, just before the first observational episode. The most distinctive difference is between the true and mean detritus fields. Since the true model is NPZ, the true detritus field is equal to zero, while the mean detritus field is non-zero because half of the realizations correspond to the NPZD model. The RMSEs of all the variables exactly equal 1, because their respective values just before the first assimilation were used for normalization. The pdf of $\Lambda(\omega)$ is uniform in the main range, and $\beta(\omega)$ has 0.5 probability of being 0 or 1. The variances of the top five modes show a rapid decay with mode number, with the top two variances orders of magnitude larger. The variances of modes 3 and 4 differ initially but become similar over time, indicating a potential cross-over at $t = 5$.

In Fig. 2-11, we directly show the state of the system at time $t = 25$, after eleven GMM-DO data assimilation (six zooplankton values every two non-dimensional times). We find that our Bayesian learning framework is able to learn the true model to be NPZ, along with the posterior pdf of $\Lambda(\omega)$ concentrated around the true value of 3.6. The mean fields also match the true fields, especially the detritus mean field becoming very close to 0 at all the spatial locations. The RMSEs for all the variables decrease over time, up to about $t = 15$. At that time, the RMSE for the phytoplankton

field increases due to a mismatch in the strength of the bloom, thus showing that the zooplankton data are not sufficiently informative for the same. The pdf of $\Lambda(\omega)$ features multiple peaks and thus still indicates that competing hypotheses remain for different pairs of parameter values; this was already the case in the intermediate assimilation steps (not shown). The evolution of the variances of the top five modes shows that these variances can increase and cross-over, for example, lower modes become more important as learning progresses. As the bloom develops, more complex nonlinear dynamics is activated, leading to the growth of some uncertainty modes. Results show that our Bayesian filter captures this as well as biases and non-Gaussian behaviors in the pdfs.

We performed other experiments with parameter sensitivity studies similar to those of Experiments-1; similar trends were found.

2.4.3 Experiments 3: Learning unknown functional form

In our third set of experiments, the primary goal is to learn the functional form of the zooplankton mortality without any prior knowledge of candidate forms, along with the uncertain biological tracer fields. We utilize stochastic piece-wise linear functions to parameterize a large set of possible functional forms within a specified range, as explained in Sect. 2.2.2. Such a parameterization encompasses many different classes of functions, for example, polynomial, exponential, logarithmic, sinusoidal, etc. The right-hand-side of the stochastic NPZ model with the unknown function is given by,

$$\begin{aligned}
S^N &= -G\frac{PN}{N+K_u} + \Xi P + \Gamma Z + \underbrace{F(Z;\omega)}_{\text{Unknown Function}} + R_m\gamma Z(1-\exp^{-\Lambda P}) \\
S^P &= G\frac{PN}{N+K_u} - \Xi P - R_m Z(1-\exp^{-\Lambda P}) \\
S^Z &= R_m(1-\gamma)Z(1-\exp^{-\Lambda P}) - \Gamma Z - \underbrace{F(Z;\omega)}_{\text{Unknown Function}}
\end{aligned} \tag{2.31}$$

From prior knowledge [3], the non-dimensional value of zooplankton is assumed non negative and its maximum value to be 0.3. Thus, $F(Z;\omega)$ is set to be composed

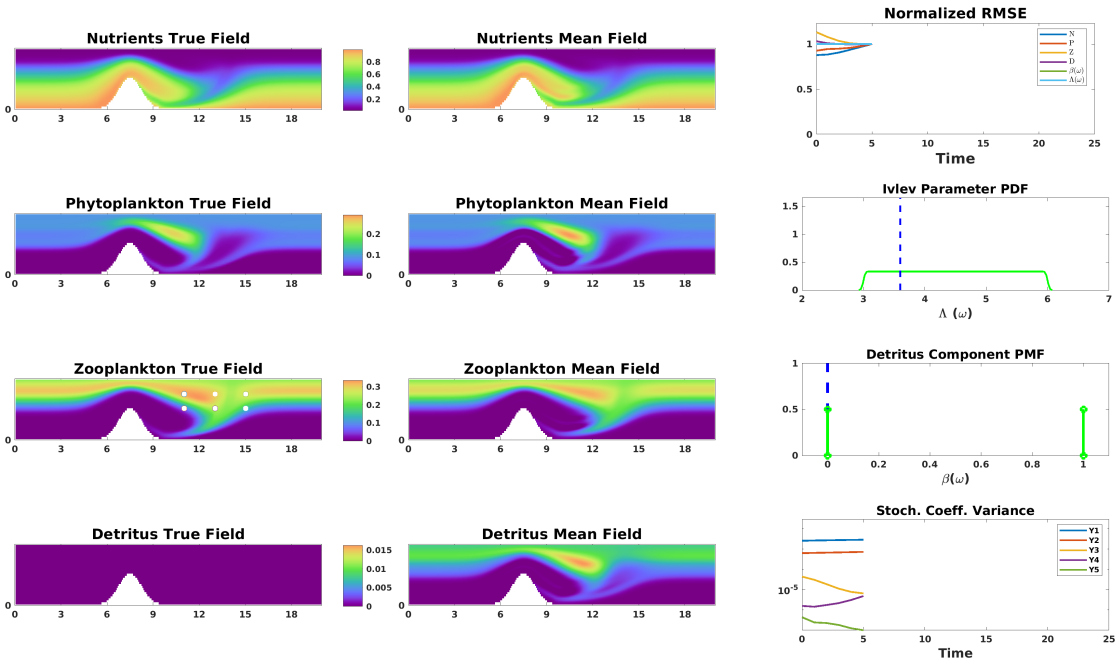


Figure 2-10: Experiments-2: State of the true and prior estimate NPZD fields and parameters at $t = 5$ (i.e. just before the 1st assimilation). The first two columns consist of the non-dimensionalized true (left) and estimate mean (right) tracer fields of N, P, Z, and D. In the third column, the first panel shows the variation of normalized RMSE with time for all the stochastic state variables and parameters. The next two panels contain the pdf of the non-dimensional $\Lambda(\omega)$ and $\beta(\omega)$ (to learn the complexity, NPZ vs. NPZD), with their true unknown values marked with blue dotted lines. The last panel shows the evolution with time of the variance (log scale) of the top five modes. The velocity field is deterministic with $Re = 1$. Additionally, the white circles on the zooplankton true field mark the six observation locations.

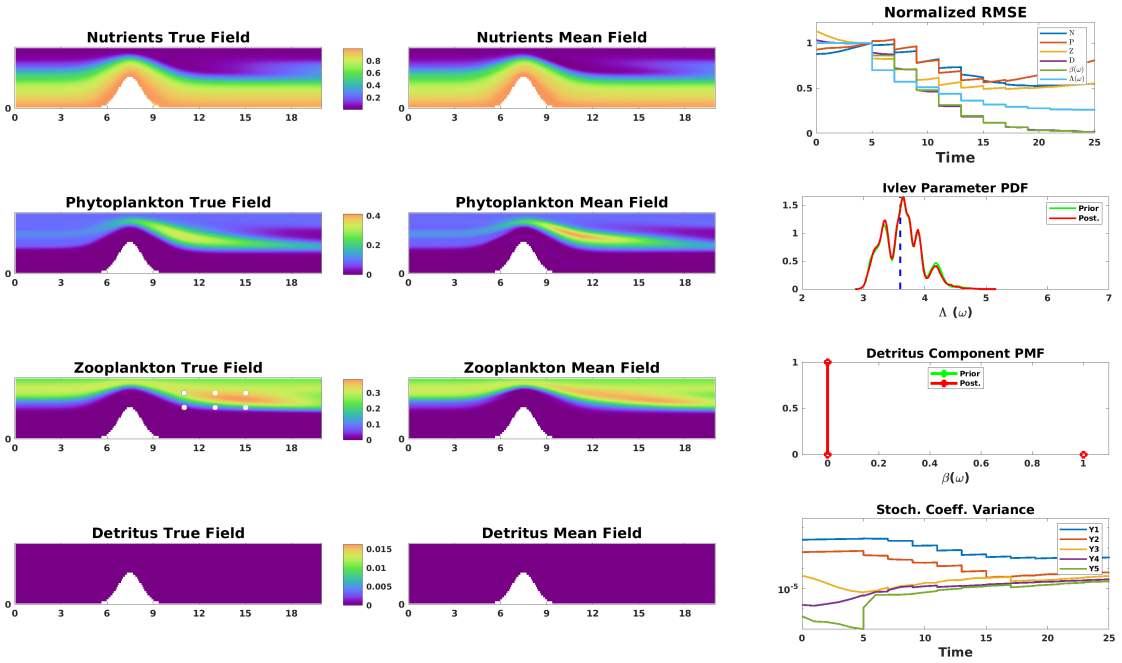


Figure 2-11: Experiments-2: As Fig. 2-10 but for posterior fields and parameters at $t = 25$ (i.e. just after the 11th assimilation).

of any continuous piece-wise linear segments in the interval $Z \in [0, 0.3]$. Dividing this interval $[0, 0.3]$ into 10 equal non-overlapping sections, such that, $0 = Z_L^0 < Z_R^0 = 0.03 = Z_L^1 < ... < Z_R^9 = 0.27 = Z_L^{10} < Z_R^{10} = 0.3$, $F(Z;\omega)$ is thus represented as,

$$F(Z;\omega) = \sum_{k=0}^{11} \gamma_k(\omega)\Psi_k(Z) \tag{2.32}$$

where,

$$\Psi_0(Z) = \begin{cases} \frac{1}{0.03}(0.03 - Z) & \text{if } 0 \leq Z \leq 0.03 \ , \\ 0 & \text{otherwise} \end{cases}$$

$$\Psi_k(Z) = \begin{cases} \frac{1}{(Z_R^{k-1} - Z_L^{k-1})}(Z - Z_L^{k-1}) & \text{if } Z_L^{k-1} \leq Z \leq Z_R^{k-1} \ , \\ \frac{1}{(Z_R^k - Z_L^k)}(Z_R^k - Z) & \text{if } Z_L^k \leq Z \leq Z_R^k \ , \qquad \text{for } k \in \{1, ..., 10\} \ , \\ 0 & \text{otherwise} \end{cases}$$

$$\Psi_{11}(Z) = \begin{cases} \frac{1}{0.03}(Z - 0.27) & \text{if } 0.27 \leq Z \leq 0.3 \ , \\ 0 & \text{otherwise} \end{cases} \tag{2.33}$$

Each set of realizations of $\gamma_k' s,\ k \in \{0, ..., 11\}$ are sampled in such a way that they do not lead to a prior with unnatural highly fluctuating functions. The function range is set within 0 and 0.08; it is non-negative as mortality is negative in the zooplankton equation (2.31). To initialize the tracer fields, we find equilibrium solutions corresponding to each realization of the zooplankton mortality function. The stochastic NPZ reactions (Eq. 2.31) are coupled with the RANS flow PDEs and used in the stochastic ADR PDEs that are solved with the DO methodology (Sect. 2.3.3). Table 2.1 provides the values of other known model and hyper- parameters. The learning objective of these experiments is to learn $F(Z; \omega)$ by estimating $\gamma_k' s$ along with the biological tracer fields.

True solution generation: The true solution contains quadratic zooplankton mortality, with values of the other parameters provided in Table 2.1. **Observations and learning parameters:** Observations remain sparse in time and space, but here consists of the nutrient field at 8 spatial locations, starting at $t = 1$ and occurring every two non-dimensional times. In these experiments, we start the assimilation at the earlier $t = 1$ time in order to limit the exploding growth of uncertainty in the system, because each function realization leads to very different biological dynamics, several of which would lead to nonphysical biological states. Other hyper-parameters

related to the GMM-DO filtering are provided in Table 2.1. **Numerical method:** As for the last two sets of experiments, the DO equations and the deterministic governing equations for the true solution are solved using the modular finite-volume framework described in Sect. 2.3.5. **Learning metrics:** We compare the true fields and parameters their DO estimates. To quantify performance, we also examine the evolution of the normalized RMSEs and pdf and realizations of the stochastic piece-wise linear functions.

Learning results

Figure 2-12 illustrates the prior at $t = 1$. Every realization in the space of the unknown function is assumed to be equilikely. In general, mortality being 0 for $Z = 0$ is common knowledge. Otherwise, it could act as a sink for zooplankton and lead to negative tracer values. However, we let this be discovered by the learning algorithm. The DO biogeochemical mean fields are quite far from the unknown true fields, and the prior function realizations are not similar to the true quadratic mortality.

As the eight N observations are assimilated every two non-dimensional times, nearly all the piece-wise linear function realizations converge to the true quadratic mortality. Results after 13 GMM-DO assimilation in Fig. 2-13 show this. We find however that the N data are not as informative about mortality function for Z beyond 0.25. This is in part because the maximum value reached in the true Z field is ~ 0.2, which limits the uncertainty reduction in that larger Z regime. The mean fields also converge to the true fields. The normalized RMSEs of all the biogeochemical fields indeed decrease at each assimilation. The learned phytoplankton mean field however remain a bit higher than true fields, in part because they were much higher initially. It is also because the observed data (here eight N data) are not equally informative about all the learning objectives. As in [96, 1, 9], this is confirmed by mutual information fields (not shown).

Other experiments included studying the effect of incorporating or excluding prior knowledge such as the function value being 0 for $Z = 0$ and using smoothly varying function realizations. For the former, sampling γ_k's independent of each other led to

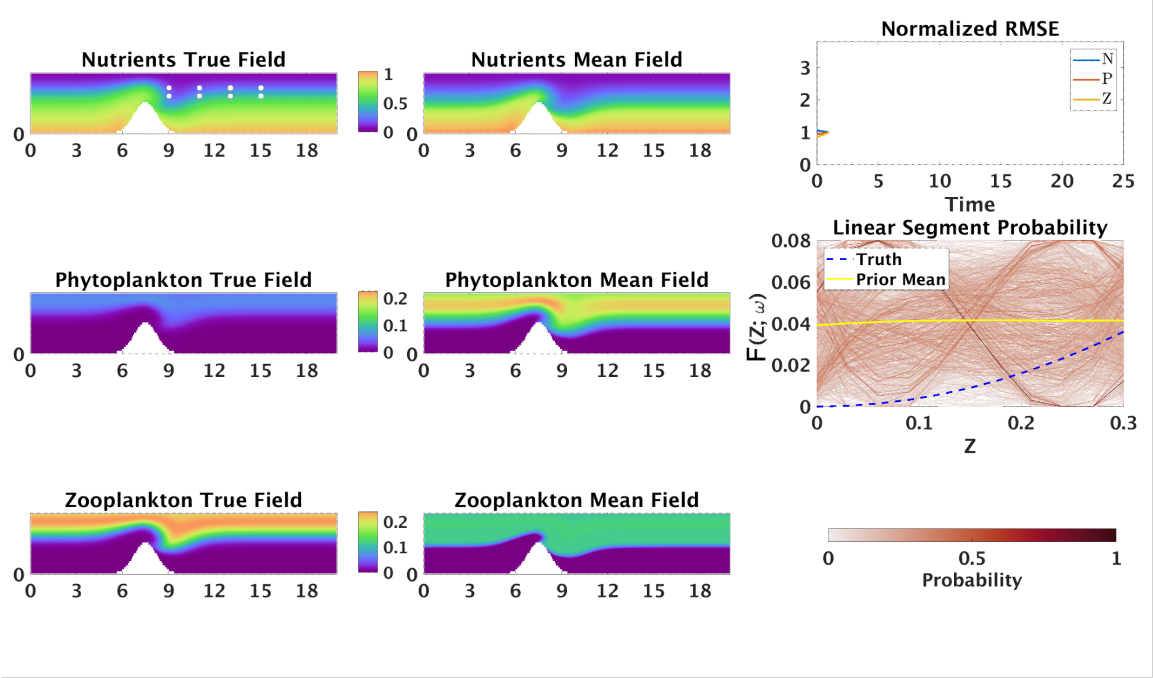


Figure 2-12: Experiments-3: State of the true and prior estimate NPZ fields and parameters at $t = 1$ (i.e. just before the 1st assimilation). The first two columns consist of the non-dimensionalized true (left) and estimate mean (right) tracer fields of N, P and Z. In the third column, the first panel shows the evolution of normalized RMSE for all the stochastic state variables. The second panel contains all the realizations of the unknown functional form approximated by piece-wise linear segments. The function realizations are colored according to their respective normalized probability density values. The velocity field is deterministic with $Re = 1$. Additionally, the white circles on the nutrient true field mark the 8 observation locations.

highly fluctuating function realizations which completely impaired the learnability of the unknown function. For the latter, enforcing $\gamma_0 = 0$ sets $F(0;\omega) = 0$ for all realizations, which improved the convergence between the learned function realizations and the true function. Finally, increasing the number of independent observations (more N data, data for Z or P as well, etc.) also improved the sharpness of our GMM-DO inference: in all examples we show, we highlight cases with sparse observations as in real ocean applications.

We also repeat this experiment using quadratic piece-wise polynomials instead of linear ones. In Fig. 2-14, we directly provide the posterior at $T = 25$. Once again, the mean and nearly all the ensemble members of the functional approximated polynomial approximations converges to a quadratic curve, which is the truth, and is successful

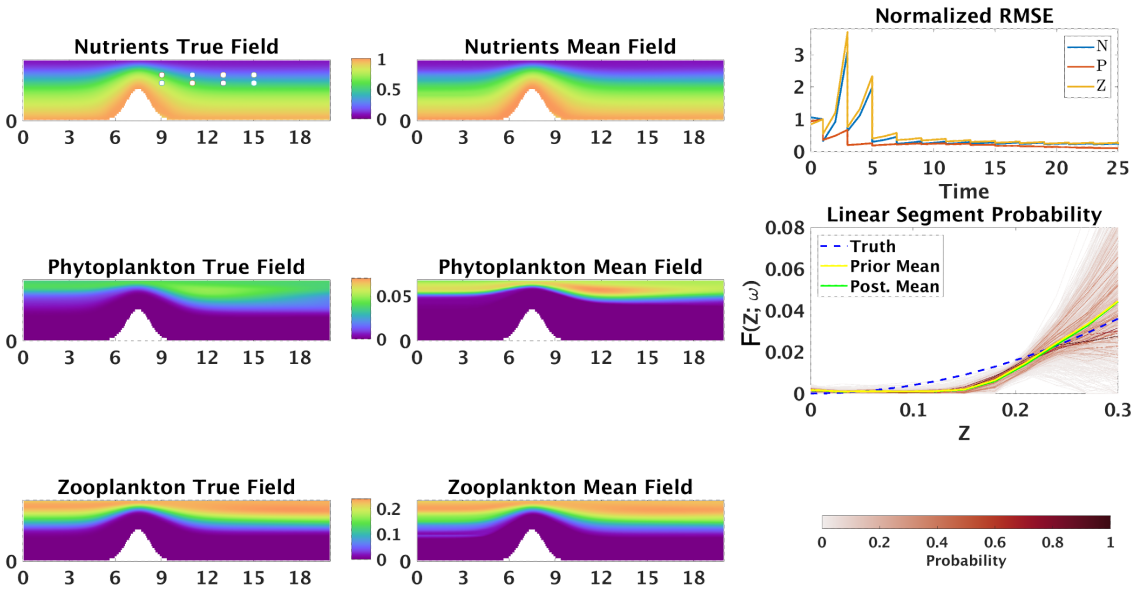


Figure 2-13: Experiments-3: As Fig. 2-12 but for posterior fields and function at $t = 25$ (i.e. just after the 13th assimilation).

in learning the common-sense logic of the mortality function being 0 for $Z = 0$. At the same time, the mean and true fields also look similar in the posterior at $T = 25$. This learning is also evident from the decrease in RMSE in Fig. 2-14 at each learning step. The use of quadratic (Fig. 2-14) over linear (Fig. 2-13) segments also improves the quality of the learned ensemble of unknown function and it is better matches the truth. This is possibly the case because the true function is itself quadratic.

2.4.4 Experiments 4: Learning in chaotic dynamics

In the last set of experiments, in order to robustly test our algorithms, the aim is to learn a five-component NNPZD model with a flow of Reynolds number $Re = 500$. At such high Re, vortices start to shed in the wake of the seamount and the flow chaotic. The learning objectives include all 5 biogeochemical fields, the Ivlev grazing parameter (Λ), the phytoplankton-specific mortality rate (Ξ), the zooplankton maximum grazing rate (R_m), the zooplankton specific mortality (Γ), and the presence or absence of the quadratic zooplankton mortality term. The stochastic NNPZD reactions, with all the

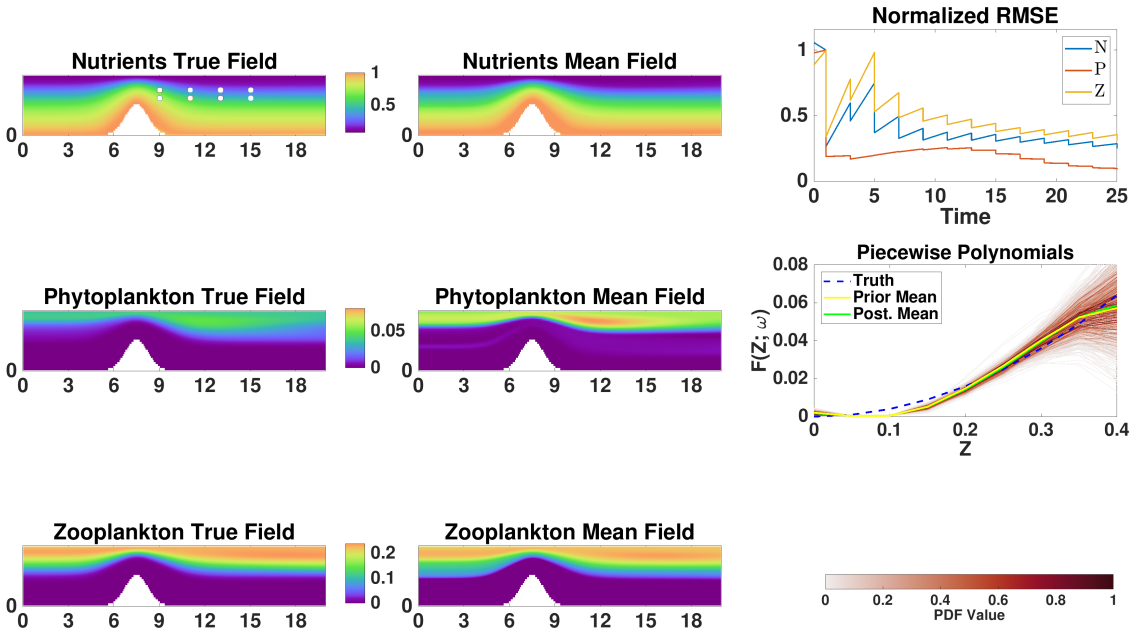


Figure 2-14: The posterior state of the NPZ model based stochastic dynamical system used in Experiment 3, at $T = 25$ (i.e. just after the 13th observational episode). The unknown functional form is approximated using piece-wise quadratic segments, while, the rest of the description is same as Figure 2-13.

uncertain parameters explicitly containing ω as an argument, are given by,

$$
\begin{aligned}
S^{NO_3} &= \Omega NH_4 - G\left[\frac{NO_3}{NO_3+K_u}\exp^{-\Psi_I NH_4}\right]P\,,\\
S^{NH_4} &= -\Omega NH_4 + \Phi D + \Gamma(\omega)Z + \alpha(\omega)\underbrace{(\Gamma_2 Z^2)}_{\text{Quad. Z Mort.}} - G\left[\frac{NH_4}{NH_4+K_u}\right]P\,,\\
S^{P} &= G\left[\frac{NO_3}{NO_3+K_u}\exp^{-\Psi_I NH_4} + \frac{NH_4}{NH_4+K_u}\right]P - \Xi(\omega)P\,,\\
&\quad - R_m(\omega)Z(1-\exp^{-\Lambda(\omega)P})\,,\\
S^{Z} &= R_m(\omega)(1-\gamma)Z(1-\exp^{-\Lambda(\omega)P}) - \Gamma(\omega)Z + \alpha(\omega)\underbrace{(\Gamma_2 Z^2)}_{\text{Quad. Z Mort.}}\,,\\
S^{D} &= R_m(\omega)\gamma Z(1-\exp^{-\Lambda(\omega)P}) + \Xi(\omega)P - \Phi D\,.
\end{aligned}
\tag{2.34}
$$

Initially, we assume uniform and independent pdfs for the 4 uncertain regular parameters and equiprobability for the quadratic zooplankton mortality term to be present or absent. The NNPZD reactions (2.34) are coupled with the deterministic RANS flow PDEs and used in the stochastic ADR PDEs that are solved with the DO methodology (Sect. 2.3.3). The other known physical-biogeochemical model parameters as well as the hyper-parameters for the DO equations are provided in Table 2.1.

True solution generation: The true solution from which observations are extracted, corresponds to the non-dimensional values, 1.5 for Λ, 0.04 for Ξ, 0.6 for R_m, 0.14 for Γ, and 0 for α, i.e. the quadratic mortality term absent. The state fields are initialized and evolved as described in Sect. 2.3.7. **Observations and learning parameters:** Observations remain sparse and univariate, but due to the unstable and fast dynamics of the flow, there is a need for a bit more frequent data than in other experiments. The phytoplankton field is observed at nine locations starting at $t = 2$ and subsequently every one non-dimensional time. In total, we assimilate 24 times, i.e. until $t = 25$. Other hyper-parameters related to the GMM-DO filtering are provided in Table 2.1. **Numerical method:** The DO equations and the deterministic governing equations for the true solution remain solved using the finite-volume framework (Sect. 2.3.5). **Learning metrics:** We compare the true fields and parameters their DO estimates. To quantify performance, we compute the evolution of the

normalized RMSEs for all the 5 biological fields and 5 stochastic parameters. We also analyze the evolution of pdfs of the regular and special stochastic parameters, and the variances of DO coefficients.

Learning results

Figure 2-15 shows the prior estimates at $t = 2$. The flow has just started to develop. There are significant differences between the true and mean biogeochemical fields. The normalized RMSEs are equal to 1 by construction. The pdfs of all parameters remained as they were initially since no data has been assimilated.

Figure 2-16 illustrates the posterior estimates at $t = 2$, just after the first assimilation. Large corrections were made to the mean tracer fields (also visible in their RMSEs that decay by about 15 to 25%), and the GMM-DO learning already predicts the absence of quadratic zooplankton term. These first 9 P observations are not as informative however about the other parameters (their RMSEs only decay by about 4% to 8%).

Figure 2-17 shows the estimates at $t = 25$, after 24 GMM-DO assimilation. In addition to the mean fields, our augmented filter has been learning the 4 regular parameters as well. Their posterior pdfs have become Gaussian which occurs in intermediate assimilation steps (not shown). We also show the evolution of variance of the top 3 modes. We find that the total variance on average either decreases or remains similar, while that of individual modes in general decreases but may also increase in accord with the stochastic dynamics. The velocity field being chaotic renders the learning more challenging in this experiment but our framework can still meet all the learning objectives, even with sparse and univariate data.

Other experiments were performed. As expected, they demonstrated sensitivity to the schedule, type, and quantity of observations. With only nine sparse and univariate data, starting them after the chaos sets in, or sampling even less frequently than every one non-dimensional time, led to posterior pdf of some stochastic parameters that were not concentrated around their respective true values. Similar results were found if even less than nine data were collected. Adding other observation types improved

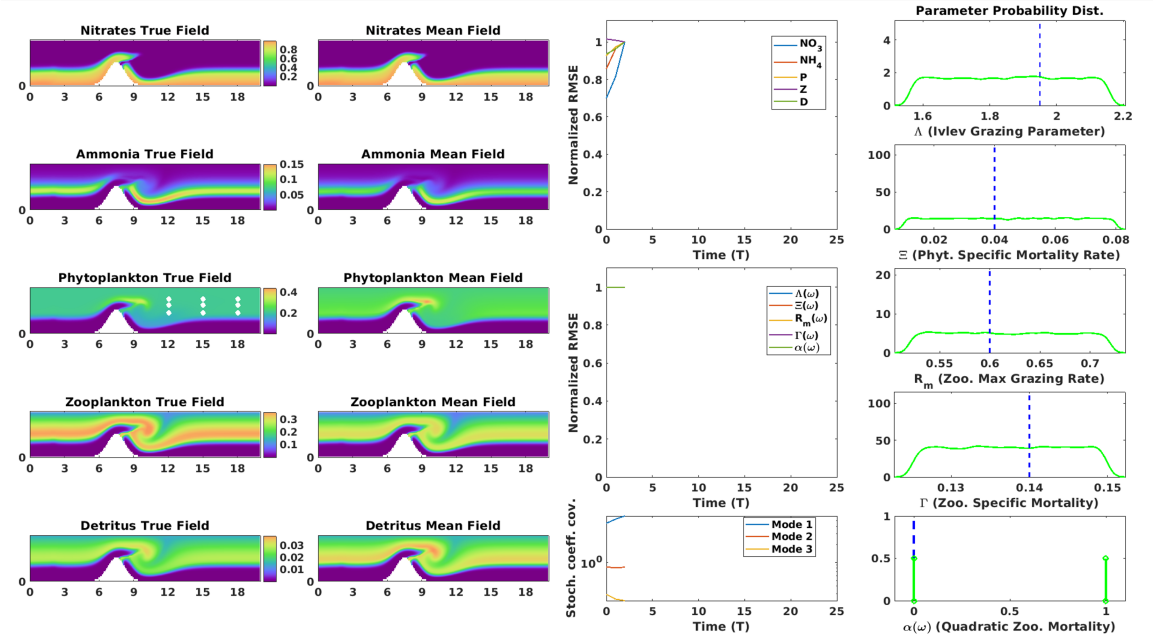


Figure 2-15: Experiments-4: State of the true and prior estimate NNPZD fields and parameters at $t = 2$ (i.e. just before the 1st assimilation). The first two columns consist of the non-dimensionalized true (left) and estimate mean (right) fields of NO_3, NH_4, P, Z, and D. In the third column, the first two panels show the evolution of the normalized RMSEs for the 5 state variables and 5 parameters. The third panel shows the evolution of variance of the top 3 DO modes. In the fourth column, the panels contain the pdf of the non-dimensional $\Lambda(\omega)$, $\Xi(\omega)$, $R_m(\omega)$, $\Gamma(\omega)$, and $\alpha(\omega)$ (learns the presence or absence of quadratic zooplankton mortality), with their true unknown values marked with blue dotted lines. The velocity field is deterministic with $Re = 500$. Additionally, the white circles on the phytoplankton true field mark the 9 observation locations.

the learning. For other sensitivity studies, trends similar to other experiments were found.

2.5 Summary

Biogeochemical-physical models for the ocean are inherently uncertain due to the inability of capturing all the complex interactions and processes in the ocean ecosystem with a single mathematical model. Uncertainty could manifest in many different forms, such as the initial conditions, boundary conditions, parameters, and the model equations themselves. In general, Bayesian approaches are advantageous for meld-

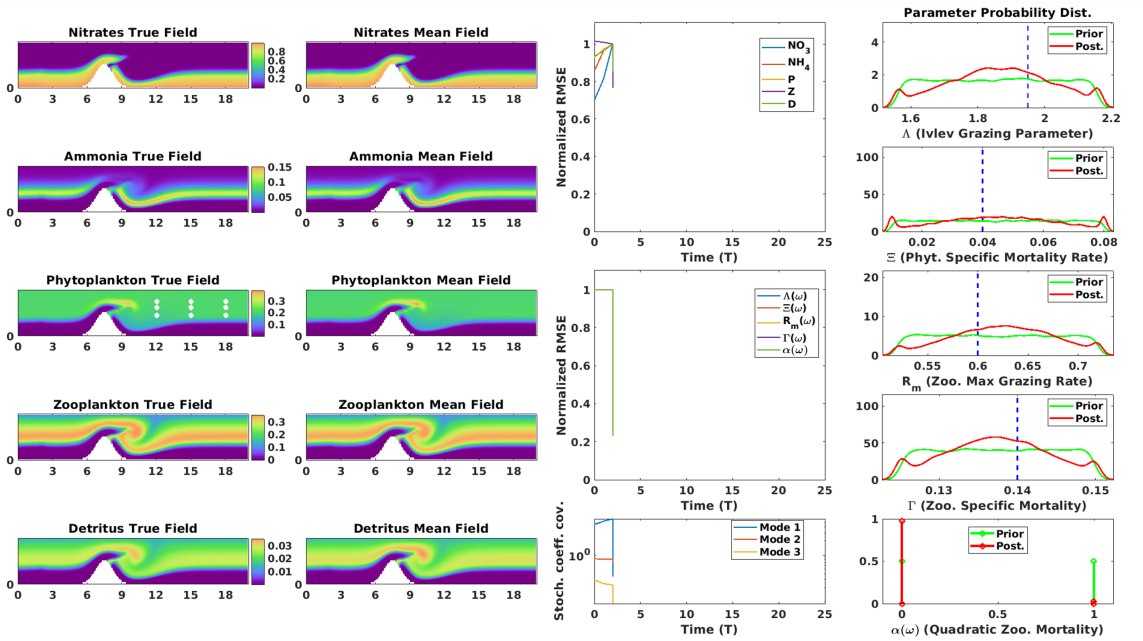


Figure 2-16: Experiments-4: As Fig. 2-15, but for posterior fields and parameters at $t = 2$ (i.e. just after the 1st assimilation).

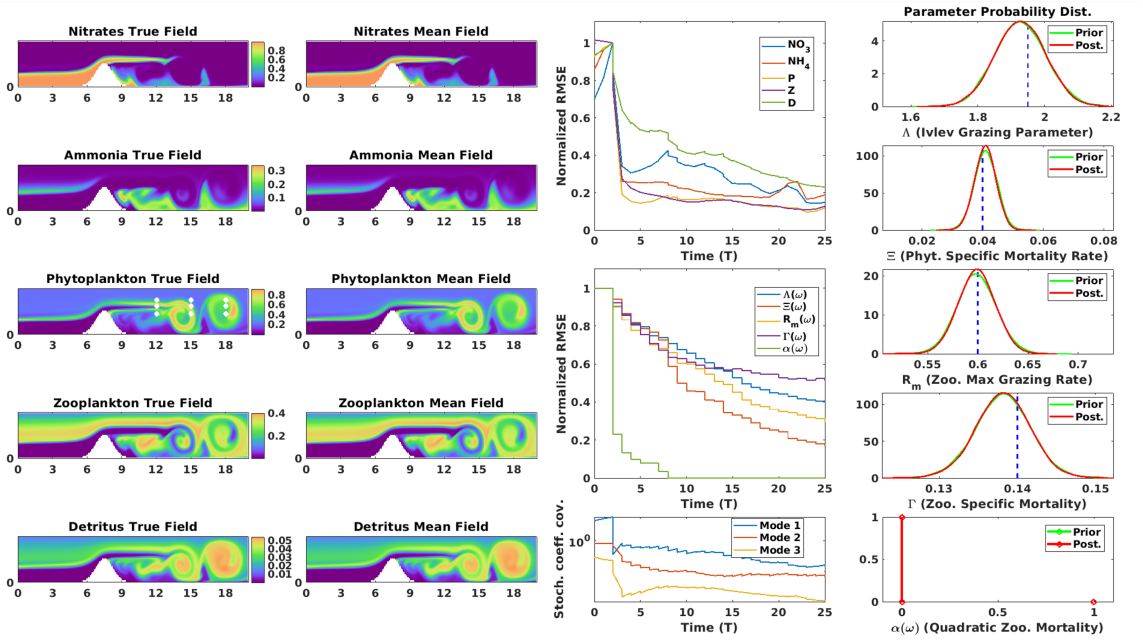


Figure 2-17: Experiments-4: As Fig. 2-15, but for posterior fields and parameters at $t = 25$ (i.e. just after the 24th assimilation).

ing observations with the model, as they provide the ability to take into account all the existing prior knowledge into the learning process, accompanied by the associated uncertainty estimates. Thus, we build upon the approach developed in Lu and Lermusiaux, 2014 & 2021 [10, 8] for the simultaneous estimation of states and parameters along with discrimination among candidate models in high-dimensional stochastic dynamical systems using sparse observations. However, often none of the candidate models is exactly equal to the true model, or the functional form is yet completely elusive to scientists. Mostly, also the candidate models are compatible with each other, for example, only certain functional terms in a model are unknown, or low complexity models are embedded in higher complexity models. These situations were addressed in two novel ways: first, using special stochastic parameters to unify all the candidates into a single general model; second, parameterizing unknown functions using stochastic piece-wise polynomial functions, allowing us to search in an infinite candidate space. Our new methodology not only seamlessly and rigorously discriminated between existing models, but also extrapolated out of the space of models to discover newer ones. In all cases, the results were generalizable and interpretable, and our Bayesian estimations provided much more than maximum likelihood estimates: they predicted and updated the complete joint probability distribution of states, parameters, and models. All of this is achieved just at the cost of single stochastic model simulation with parameter estimation, enabling both discrimination and discovery of models. Our rigorous PDE-based Bayesian learning framework combines the Dynamically Orthogonal (DO) equations for efficient reduced-dimension uncertainty evolution; and the Gaussian mixture model (GMM) DO filtering algorithm for the nonlinear, non-Gaussian inference of the states, parameters, and model equations simultaneously.

The performance of our Bayesian learning framework is evaluated using a series of identical twin experiments with compatible and embedded models of the three-component NPZ model (nutrients (N), phytoplankton (P), and zooplankton (Z)), the four-component NPZD model (N, P, Z and Detritus (D)), and the five-component NNPZD model (ammonia (NH_4), nitrate (NO_3), P, Z, and D). In the first set of

experiments, we use the NPZ model, with uncertainty introduced due to the unknown Ivlev grazing parameter value and ambiguity in the presence or absence of the quadratic zooplankton mortality term. We were able to meet our learning objectives of simultaneously estimating the biological state variables, Ivlev parameter, and the unknown functional form just using sparse zooplankton observations in space and time. We further noticed lower peaks in the posterior probability distributions of the parameters, which often indicate alternative combinations of parameter values which could potentially explain the observed data, thus showcasing the ability of our framework to capture non-Gaussian statistics. Once all the data becomes available, we also demonstrate smoothing backward in time. In the second set of experiments, we demonstrated the ability to learn the complexity of the model by identifying the true model among NPZ and NPZD along with the uncertain Ivlev grazing parameter. In the third set of experiments, we assumed no prior knowledge about the functional form of zooplankton mortality and generated a function space using stochastic piecewise linear segment approximations. Such a formulation helps us to perform a search in a rich functional space, and discover new functions. We also repeat the experiment using quadratic segments which improves the quality of the learned function. The last set of experiments involved learning the complicated NNPZD model in an unsteady deterministic flow with vortex shedding. The NNPZD model had uncertainty in all the tracer fields, four parameters, and in the presence or absence of the zooplankton mortality term. All of the learning objectives were achieved simultaneously.

These four sets of experiments acts as complementary benchmarks, allowing us to showcase all the features of our learning framework. Next steps include application of this Bayesian learning framework to more complex oceanic applications. Even though we demonstrate our learning framework using biogeochemical models, it is applicable to any problem with model uncertainty, for example, in medical applications, economics, energy, etc. Our framework could give scientists in different disciplines not only the ability to choose between competing existing hypotheses, but to also discover new ones in an efficient manner.

Chapter 3

Bayesian Learning for Fish Models[1]

Fisheries are a major industry in the coastal states of India, employing millions of people and contributing to 1.1% of GDP and 5.3% of agricultural GDP. Globally, the Indian fishing industry is the third largest in the world. The total marine fish production is around 3 billion metric tons. Indian waters contain about 2,500 species of finfishes and shellfishes. Among these, there are about 65 commercially important species or groups. In 2004, 52% of these commercially important groups were pelagic and midwater species. In 2006, over 600,000 metric tons of fish were exported, to some 90 countries, earning over $1.8 billion [97]. Increased demand for fish, coupled with unsustainable fishing practices lead to over-exploitation and fast depletion of fish stocks. Coastal fisheries and aquaculture stocks often thrive on very specific water conditions—building capabilities for coastal physical-ecosystem forecasting and monitoring will help ensuring and managing the survival and reproduction of healthy stock. Without sustainable fisheries management and conservation practices in place, there could be dire consequences for the many communities that rely on the ocean for their economic well-being.

The ocean ecosystem is commonly divided into two main levels, Lower Trophic Levels (LTL) and Higher Trophic Levels (HTL) including fish. A number of models have been proposed in the literature for either of the levels (see section 3.1). How-

[1]Parts of this chapter appeared in: Gupta, A., Haley, P. J., Subramani, D. N., & Lermusiaux, P. F. (2019, October). Fish modeling and Bayesian learning for the Lakshadweep Islands. *In OCEANS 2019 MTS/IEEE SEATTLE (pp. 1-10). IEEE.*

ever, due to the semi-empirical methodology of developing these models, there is uncertainty associated with the parameters, functional forms, and the level of complexity of fish models. Thus, in this chapter, we use our novel PDE-based Bayesian learning framework to showcase a series of learning experiments that simultaneously infer the augmented state variables, parameters, and parameterizations of the fish model SEAPODYM (described in the next two sections) coupled with a LTL dynamical model and a nonhydrostastic varaible-density Boussinesq flow past a seamount.

3.1 Fish Modeling

First, we provide a brief literature review of different models used for the two main parts of the marine ecosystem.

3.1.1 Lower Trophic Level

There exist many well-studied models of varying complexities for LTL. A basic model is a simple, 3-component nutrient-phytoplankton-zooplankton (NPZ) model [42, 43, 62]; Franks, 2002 ([40]) provides a thorough review on development of such models. In a workshop on the status of upper layer coupled biological-physical modeling [44], researchers proposed couplings of various mixed layer physical models with NPZ and NPZD (NPZ-Detritus) biological models. Fasham et. al., 1990 ([45]) presented a 7-component model of the annual cycles of plankton dynamics and nitrogen in the ocean. One of the most complex lower-trophic level marine ecosystem model is the European Regional Seas Ecosystem Model (ERSEM, [46, 47, 48]), initially developed for the North Sea.

3.1.2 Higher Trophic Level

HTL models vary greatly in how they model fish; some model individual fishes in Lagrangian sense, some are empirical data-based models, and some treat them as aggregate (continuous) biomass and capture more realistic biological interactions and

processes. The effective coupling of LTL models and HTL full-life cycle fish models is notoriously challenging, mainly due to the difficult practical and theoretical problems associated with resolving relevant temporal and spatial scales at all biologically meaningful trophic levels. Nonetheless, they can be coupled using different mathematical functions that model source and mortality terms, and close the ecosystem. One of the oldest models developed for fisheries management is the MultiSpecies Virtual Population Analysis (MSVPA) [98]. It solves a system of coupled nonlinear equations in terms of biomass of species and number of fishes belonging to each cohort averaged over large spatial and temporal scales. Parameterization requires stomach-content data of fish and estimates of the number of fish in a particular cohort; this requires lots of hard-to-obtain data, data only sparsely collected for a small number of fish species.

Larval Individual Based Models (IBMs) [99] attempt to model the larval stage, which is in-between LTL and HTL. They start with an ensemble of eggs seeded in the domain, and let them advect and interact with the underlying physical and biogeochemical fields, while mortality is also modeled as a stochastic event which determines whether individual eggs develop to the juvenile stage. A drawback to this approach is the fact that larvae cannot really represent fish population.

A prominent box model is NUMERO.FISH [100, 101], which simulates the daily predator-prey interactions and biogeochemical cycling of phytoplankton, zooplankton, nutrients, and detritus. The FISH model simulates the daily growth and mortality of herring in each of multiple age-classes and is coupled to NEMURO via zooplankton-dependent herring consumption, excretion, and egestion. The FISH model is based on an energy balance equation that equates energy consumed with energy expended and gained.

Interacting Particle Model for Migration of Pelagic Fish [102] models individuals rather than keeping track of the density of a population. Particles look to their neighbors to determine their directional heading at each time step, averaging the neighbors' directional headings to determine their own. This allows the particles to move together as a group. Size spectra models [103] are based on the biomass spectrum

theory, which assumes that size governs biological rates and predatory interactions. In size-spectra studies, the whole ecosystem or community is represented by a continuum of biomass and organisms are represented only in terms of their body size. The bio-ecological processes taken into account to model consumers are predation, mortality, assimilation and use of energy for maintenance, growth and reproduction.

Ordinary differential equations based models include the Ecopath with Ecosim (EwE) Ecosystem Modeling Suite [104, 105]. EwE facilitates the construction of a static ecosystem model (Ecopath) that can then be used to run time-based dynamic (Ecosim) and spatial (Ecospace) simulations. Modelling in EwE begins with creating a mass-balance model using Ecopath to obtain a static snapshot of the ecosystem under study. The underlying principle behind the mass balance approach is to balance the energy flow among different trophically linked functional groups by solving a set of simultaneous linear equations (one equation for each functional group).

SEAPODYM (Spatial Ecosystem And Populations Dynamics Model) [106, 107] is an Advection-Diffusion-Reaction (ADR) equation-based model that couples a physical-biological interaction model at basin scales, combining a forage (prey) production model with an age-structured population model of targeted (tuna predator) species. An adult habitat index combines the spatial distribution of tuna forage biomass with a temperature function defined for each species. Young and adult tuna movements are constrained by this adult habitat index while a spawning habitat index is used to constrain the recruitment to environmental conditions. Related ADR models were used for regional fisheries management [108].

Lastly, recent research involves machine learning approaches such as training Artificial-Neural-Nets (ANNs). When using ANNs, typically the output is in the form of catch-per-unit-effort (CPUE) and input includes Sea-Surface-Temperature (SST), Sea-Surface-Height (SSH), gradient of SST, chlorophyll-a, latitude, longitude, time, and other relevant quantities [109, 110, 111, 112].

3.2 Learning and Modeling Methodology

3.2.1 Physical Model

The physical model is described by the stochastic nonhydrostatic Navier-Stokes equations with a variable-density Boussinesq approximation,

$$\begin{aligned} \nabla.\boldsymbol{u} &= 0 \ , \\ \frac{\partial \boldsymbol{u}}{\partial t} &= -\nabla.(\boldsymbol{u}\boldsymbol{u}) - \nabla P + \Lambda_{Re}(\omega)\nabla^2\boldsymbol{u} + \left[\frac{\rho'}{\rho_o}\right]\mathbf{g} \ , \\ \frac{\partial \rho'}{\partial t} &= -\nabla.(\rho'\boldsymbol{u}) + \kappa\nabla^2\rho' \ , \end{aligned} \tag{3.1}$$

where $\boldsymbol{u} \equiv \boldsymbol{u}(\boldsymbol{x}, t; \omega)$ is the two-dimensional stochastic velocity field; $P \equiv P(\boldsymbol{x}, t; \omega)$, the stochastic pressure field that contains contributions from the hydrostatic pressure due to the variable density as well as the nonhydrostatic pressure; ρ_o, the mean density; $\rho' \equiv \rho'(\boldsymbol{x}, t; \omega) = \rho(\boldsymbol{x}, t; \omega) - \rho_o$, the density perturbation from the mean; $\boldsymbol{g} = -g\boldsymbol{e_z}$; κ, the constant of kinematic diffusivity; and $\Lambda_{Re}(\omega)$ is here an uncertain parameter equivalent to the inverse of eddy-viscosity (ν_E) Reynolds number ($Re = \frac{UL}{\nu_E}$). This stochastic system belongs to a domain $\boldsymbol{x} : \{x, z\} \in \mathcal{D}$, and ω is a realization index belonging to a measurable sample space Ω. We also consider the density perturbation to be solely a function of temperature $T(\boldsymbol{x}, t; \omega)$, given by the relation, $\rho' = \alpha(T - T_o)$, where α is the coefficient of expansion and T_o is a reference temperature. We specify uncertain initial velocity $\boldsymbol{u}(\boldsymbol{x}, t_{init}; \omega) = \boldsymbol{u}_{init}(\boldsymbol{x}; \omega)$ and temperature $T(\boldsymbol{x}, t_{init}; \omega) = T_{init}(\boldsymbol{x}; \omega)$ fields. The velocity uncertainty is initialized by adding sinusoidal perturbations to a divergence-free domain-confirming potential flow, while for density, different stable stratified profiles are considered for each realization.

3.2.2 LTL-Biological Model

The lower-trophic-level biogeochemical model used in the present study is adapted from Newberger et. al., 2003 ([3]). We employ the three-component NPZ model (nu-

trients ($N(\boldsymbol{x},t;\omega)$), phytoplankton ($P(\boldsymbol{x},t;\omega)$), and zooplankton ($Z(\boldsymbol{x},t;\omega)$)). The NPZ model is given by,

$$\begin{aligned} S^N &= -G\frac{PN}{N+K_u} + \Xi P + \Gamma_1 Z + a(\omega)\Gamma_2 Z^2 + R_m\gamma Z(1-\exp^{-\Lambda(\omega)P}) \ , \\ S^P &= G\frac{PN}{N+K_u} - \Xi P - R_m Z(1-\exp^{-\Lambda P}) \ , \\ S^Z &= R_m(1-\gamma)Z(1-\exp^{-\Lambda P}) - \Gamma_1 Z - a(\omega)\Gamma_2 Z^2 \ , \end{aligned} \tag{3.2}$$

where: G represents the optical model given by, $G = V_m\frac{\alpha I_l}{(V_m^2+\alpha^2 I_l^2)^{1/2}}$ and $I_l(\boldsymbol{x}) = I_l^o \exp^{k_w z}$. Along with the uncertain state variables, we assume a uncertain Ivlev grazing parameter ($\Lambda(\omega)$) and a special binary stochastic parameter ($a(\omega) \in \{0,1\}$).

The biogeochemical models are coupled with the physical model using stochastic Advection-Diffusion-Reaction (ADR) equations. Let $\phi^i(\boldsymbol{x},t;\omega), i = \{1,2,3\}$ represent the three stochastic biological tracers, the ADR equations are then,

$$\frac{\partial \phi^i}{\partial t} + \nabla.(\boldsymbol{u}\phi^i) - \kappa\nabla^2\phi^i = S^{\phi^i}(\phi^1,\phi^2,\phi^3,\boldsymbol{x},t;\omega), \quad i = \{1,2,3\} \ , \tag{3.3}$$

where $\boldsymbol{u}(\boldsymbol{x},t;\omega)$ is the stochastic velocity field which is derived from the physical model (equation 3.1), Pe is the Peclet number, and $S^{\phi^i}(\phi^1,\phi^2,\phi^3,\boldsymbol{x},t)$ are the reaction equations for various biological variables which are given by the NPZ biogeochemical model (equation 3.2). The initial conditions for this model are here generated by solving for stable equilibrium solution ($S^{\phi^i} = 0, i = \{1,2,3\}$) for each of the parameter realizations.

3.2.3 Fish Model

We use the spatial ecosystem and population dynamics model (SEAPODYM) based on an ADR formulation that focuses on spatial tuna population dynamics [113]. It couples low-trophic-level (LTL) and high-trophic-level (HTL) biological models. The physical and LTL biological models, given in sections 3.2.1 and 3.2.2, respectively, provide estimates of stochastic physical state variables such as velocity ($\boldsymbol{u}(\boldsymbol{x},t;\omega)$), temperature ($T(\boldsymbol{x},t;\omega)$), and primary production ($P(\boldsymbol{x},t;\omega)$). The primary produc-

tion acts as a source for the forage ($F(\boldsymbol{x},t;\omega)$), after taking into account the recruitment time and mortality, given by source $S(\boldsymbol{x},t;\omega) = \frac{1}{\lambda}P(\boldsymbol{x},t;\omega)\exp^{-m_r T_r(\omega)}$, where λ is the mortality, m_r is a loss coefficient, and $T_r(\omega)$ is the uncertain recruitment time. Thus, the forage field is governed by,

$$\frac{\partial F}{\partial t} + \nabla.(\boldsymbol{u}F) - \kappa\nabla^2 F = -\lambda F + S\ . \tag{3.4}$$

Tuna tend to favor certain temperature ranges and high food concentrations; the habitat index, given by $I(\boldsymbol{x},t;\omega) = g(F(\boldsymbol{x},t;\omega))\phi(T(\boldsymbol{x},t;\omega) - T_o)$, acts as a spatial field which defines the favorability of location for the fish, here tuna. We take $g(F(\boldsymbol{x},t;\omega)) = F(\boldsymbol{x},t;\omega)$ and,

$$\phi(T(\boldsymbol{x},t;\omega) - T_o) = 1/(1+\exp^{-(T(\boldsymbol{x},t;\omega)-T_o)})\ .$$

Gradients of the habitat index then affect the movement of the fishes. This is captured by defining effective advection velocities, $A_x(\boldsymbol{x},t;\omega) = u(\boldsymbol{x},t;\omega) + \chi\frac{\partial I(\boldsymbol{x},t;\omega)}{\partial x}$ and $A_y(\boldsymbol{x},t;\omega) = v(\boldsymbol{x},t;\omega) + \chi\frac{\partial I(\boldsymbol{x},t;\omega)}{\partial y}$. Hence, the population density ($P_{den}(\boldsymbol{x},t;\omega)$) of tuna is then governed by an ADR equation with the effective advection,

$$\frac{\partial P_{den}}{\partial t} + \nabla.(\mathbf{A}P_{den}) = \nabla.(D\nabla P_{den})F - Z(I)P_{den} + R\ , \tag{3.5}$$

where D is the diffusion coefficient; $Z(I)$ is a habitat index dependent mortality coefficient given by $\lambda_z \exp^{-\lambda_I I}$; and R is growth rate. Again the initial uncertainty estimates for $F(\boldsymbol{x},t;\omega)$ and $P_{den}(\boldsymbol{x},t;\omega)$ are here found by solving for the stable equilibrium solutions of equations 3.4 and 3.5 for each realizations.

3.2.4 GMM-DO Bayesian Learning

As mentioned in the last chapter, a Bayesian learning setting involves choosing a prior probability distribution for the state variables ($\mathbf{X} \in \mathbb{R}^{N_X}$) of interest, taking into account all sources of uncertainties, $p_{\mathbf{X}}(\mathbf{x})$. Observations ($\mathbf{Y} \in \mathbb{R}^{N_Y}$), with likelihood

($p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$) are used to estimate the posterior probability of the states ($p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y})$) [56, 93]. In the present problem, the state variable consists of physical, lower-trophic-level and higher-trophic level biological variables, governed by a coupled physical-biological-fish model, along with initial conditions, parameters, and parameterization uncertainties. For the observation likelihood, we assume a linear model given by, $\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{V}$; where $\mathbf{H} \in \mathbb{R}^{N_Y \times N_X}$ is the sparse observation matrix; and $\mathbf{V} \in \mathbb{R}^{N_Y}$ is a zero-mean, uncorrelated Gaussian noise with covariance matrix $\mathbf{R} \in \mathbb{R}^{N_Y \times N_Y}$. Uncertainty propagation is performed using an efficient reduced-dimension uncertainty quantification method, the Dynamically Orthogonal (DO) equations [11, 12, 15]. In the current application, even though the physics, LTL-biological, and the fish model are coupled, however, they do not share stochastic coefficients. Thus, we use the following truncated KL-decompositions,

$$\begin{aligned}
\begin{bmatrix} u(\boldsymbol{x},t;\omega) \\ v(\boldsymbol{x},t;\omega) \\ \rho'(\boldsymbol{x},t;\omega) \end{bmatrix} &= \begin{bmatrix} \bar{u}(\boldsymbol{x},t) \\ \bar{v}(\boldsymbol{x},t) \\ \bar{\rho'}(\boldsymbol{x},t) \end{bmatrix} + \sum_{i=1}^{S_{vel}} \begin{bmatrix} \tilde{u}_i(\boldsymbol{x},t) \\ \tilde{v}_i(\boldsymbol{x},t) \\ \tilde{\rho'}_i(\boldsymbol{x},t) \end{bmatrix} Y_i^{vel}(t;\omega) \\
\begin{bmatrix} N(\boldsymbol{x},t;\omega) \\ P(\boldsymbol{x},t;\omega) \\ Z(\boldsymbol{x},t;\omega) \end{bmatrix} &= \begin{bmatrix} \bar{N}(\boldsymbol{x},t) \\ \bar{P}(\boldsymbol{x},t) \\ \bar{Z}(\boldsymbol{x},t) \end{bmatrix} + \sum_{i=1}^{S_{bio}} \begin{bmatrix} \tilde{N}_i(\boldsymbol{x},t) \\ \tilde{P}_i(\boldsymbol{x},t) \\ \tilde{Z}_i(\boldsymbol{x},t) \end{bmatrix} Y_i^{bio}(t;\omega) \\
\begin{bmatrix} F(\boldsymbol{x},t;\omega) \\ P_{den}(\boldsymbol{x},t;\omega) \end{bmatrix} &= \begin{bmatrix} \bar{F}(\boldsymbol{x},t) \\ \bar{P}_{den}(\boldsymbol{x},t) \end{bmatrix} + \sum_{i=1}^{S_{fish}} \begin{bmatrix} \tilde{F}_i(\boldsymbol{x},t) \\ \tilde{P}_{den_i}(\boldsymbol{x},t) \end{bmatrix} Y_i^{fish}(t;\omega)
\end{aligned} \tag{3.6}$$

where $\begin{bmatrix} \bar{u} \\ \bar{v} \\ \bar{\rho'} \end{bmatrix}, \begin{bmatrix} \bar{N} \\ \bar{P} \\ \bar{Z} \end{bmatrix}, \begin{bmatrix} \bar{F} \\ \bar{P}_{den} \end{bmatrix}$ are the means; $\begin{bmatrix} \tilde{u}_i \\ \tilde{v}_i \\ \tilde{\rho'}_i \end{bmatrix}, \begin{bmatrix} \tilde{N}_i \\ \tilde{P}_i \\ \tilde{Z}_i \end{bmatrix}, \begin{bmatrix} \tilde{F}_i \\ \tilde{P}_{den_i} \end{bmatrix}$ are the modes; and $Y_i^{vel}(t;\omega), Y_i^{bio}(t;\omega), Y_i^{fish}(t;\omega)$ are the respective stochastic coefficients. The three sets of modes follow dynamic orthogonality w.r.t. themselves, thus allowing us to derive the corresponding DO equations. To handle uncertain functional terms, we make use of the methodology of special stochastic parameters and stochastic piece-wise polynomials developed in sections 2.2.1, 2.2.2 and 2.2.3. For detailed DO derivation

of a general stochastic dynamical system with uncertain parameters, and initial and boundary condition, see appendix A.

To perform inference of the augmented state variables, parameters and model equations, we make use of a PDE-based machine learning framework developed by combining the DO method with a Gaussian mixture model (GMM) filtering algorithm [114, 10, 115], and implemented in a finite volume framework [80]. For details, see appendices B & C.

3.3 Experimental Setup

In this section we describe details of the modeling domain, numerical methods, initialization of the stochastic simulations, true solution generation, observations, etc., which for the most part are similar to those in chapter 2.

3.3.1 Simulated Experiments and Dynamics

The experimental setup for the Bayesian learning consists of a 2-dimensional domain with a seamount representing an idealized sill or strait that can create an upwelling of the nutrients and thus phytoplankton blooms. The domain is exactly the same as that used in chapter 2, figure 2-1. The seamount also forces the advection of cold water upward, that leads to a competing effect on the habitat index, thus limiting the tuna to very specific depths. With the nonhydrostatic dynamics, internal waves, recirculations, and other instabilities can also be created downstream of the seamount, leading additional biogeochemical-fish responses. This domain is inspired by the Stellwagen Bank off of Massachusetts. Here, flow occurs from left to right in the positive x-direction over the seamount. For velocity, we apply a Dirichlet boundary condition for the inlet, no-slip for the bump, free-slip at top and bottom, and open boundary at the outlet. For the tracer fields, we use zero-Neumann on all the boundaries. The parameter values associated with the domain are provided in Table 3.1.

3.3.2 Numerical Method

We solve the derived DO equations using a modular finite-volume framework [80]. The geometry is discretized using a uniform, staggered C-grid. The advection operator is discretized using a total variation diminishing (TVD) scheme with a monotonized central limiter [81]. Diffusion is treated implicitly, with a second-order central difference scheme. All the reaction terms are computed explicitly. To handle the complex boundaries with the structured Cartesian grid, a ghost cell immersed boundary method is adopted for accurate enforcement of the boundary conditions (Sect. 5.2). A first-order forward Euler method is used to evolve the mean and the modes, and a four-stage Runge-Kutta scheme is employed for the stochastic coefficients. It is ensured that we satisfy the Courant-Friedrichs-Lewy (CFL) condition at all times. Refer to Ueckermann and Lermusiaux, 2013 [82], and Feppon and Lermusiaux, 2018 [14] for more details on the numerical schemes we use.

3.3.3 Initialization

The parameters contained in the physical and biological models can be divided into two categories, deterministic parameters and stochastic parameters. While the values of deterministic parameters are fixed for every realization, but the stochastic ones can vary based on their probability distributions. To initialize, parameter values are sampled for the stochastic parameters (both regular and special) from their initial joint probability distributions, and the corresponding equilibrium solution is found for each realization. Definitions and values of all the parameters are provided in Table 3.1.

Physical model

The flow is slow moving with the dimensional velocity at the inlet to be approximately $U \approx 10^{-2}\ m/s$, while the eddy viscosity is considered to be $\nu_E \approx 0.1 - 10\ m^2/s$. Now with a horizontal length scale of $D \approx 1\ km$ for the seamount, the typical value of eddy viscosity based Reynolds number (Re) turns out to be in the range, $Re = 1 - 100$.

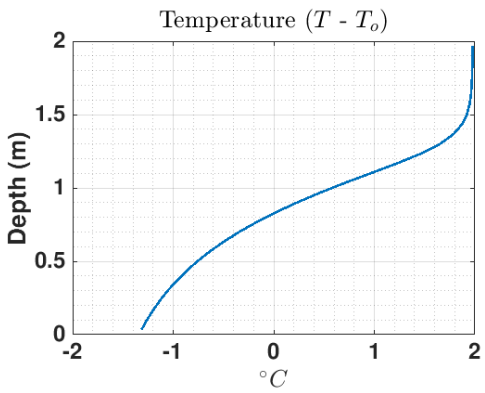
Using this eddy viscosity based Re, helps us to implement Large Eddy Simulations (LES) in the simplest form, thus capturing large scale dynamics. For the mean velocity, we use a divergence free velocity field, conforming to the given domain. For cases with uncertainty in the velocity field, the modes are initialized using a streamfunction, while the stochastic coefficients using a variance function, as given in Ueckermann and Lermusiaux, 2013 [82]. And the Reynolds number ($Re(\omega)$) is sampled from its prescribed distribution. To initialize the temperature field, we use a generalized logistic function given by,

$$T(z) = A + \frac{K - A}{(C + Q\exp^{-Bz})^{1/\nu}} \tag{3.7}$$

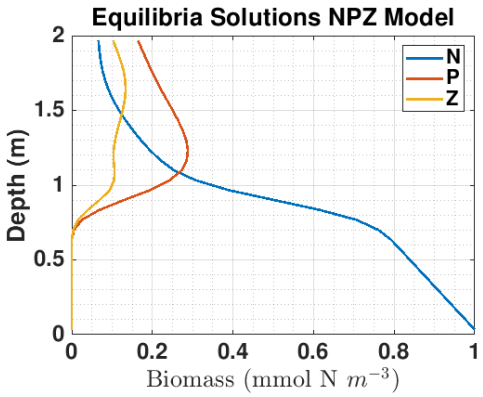
where z is the depth, and K, A, C, Q, B, ν are various parameters which control the shape of the function. These parameters are sampled from normal distribution to create ensemble of temperature profiles, which are close to realistic scenarios. A sample temperature profile is presented in figure 3-1. Finally, we recreate all the velocity realizations and take a joint SVD with temperature realizations, to initialize the mean, modes and stochastic coefficients for the physics part.

LTL-biological model

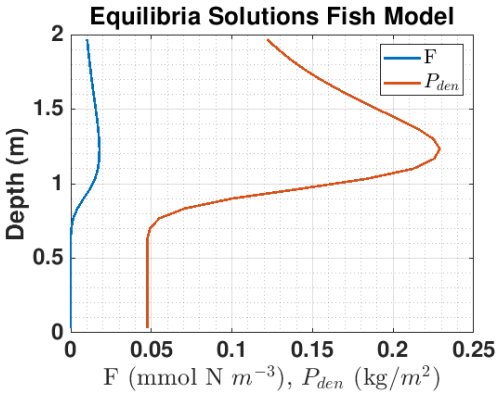
To find the equilibrium solutions, consider reference frame without the physics, i.e. the advection and diffusion terms, hence the biological models are now only depth ($-hH < z < 0$) dependent. Once our system is in equilibrium, the concentration of biological tracers does not change with depth and time. As all the models respect biomass conservation $\sum_{i=1}^{N_\phi} \frac{d\phi^i}{dt} = 0$, hence all the tracers sums to constant total biomass $\sum_{i=1}^{N_\phi} \phi^i = N_T$, which could vary with depth. We consider N_T to be linearly increasing from 10 $mmol\ N\ m^{-3}$ at the surface to 30 $mmol\ N\ m^{-3}$ at the depth of 100 m, for all the models. A non-linear solver is used to find the steady equilibrium solutions of the equations in the NPZ model (equation 3.2; or the one modified with the piece-wise function approximation). Figure 3-1 shows the one-dimensional (z) solutions for each of the models, for a particular set of parameter values. Hence for



(a) Sample temperature profile



(b) NPZ equilibrium solution



(c) Fish equilibrium solution

Figure 3-1: Sample temperature profile and equilibrium solution for the LTL-biological and fish model.

every realization, we use the corresponding equilibrium solution for each of its tracer field, to initialize the masked domain at every $x-$ location. It should be ensured that none of the realization of the stochastic parameters leads to unphysical equilibrium solutions. The value of 30 $mmol\ N\ m^{-3}$ is used to non-dimensionalize all the tracer fields and appropriate parameter values. Next, we take Singular Value Decomposition (SVD) of the ensemble to initialize mean, modes and stochastic coefficients.

Fish model

The forage and the fish population density is also initialized by first computing the equilibrium solutions. The reaction part in equations 3.4 and 3.5 is equated to zero, which is also dependent on primary production (phytoplankton concentration) from

the LTL-biological model, and the temperature field from the physics model. Creation of the ensemble is followed by joint SVD to initialized the mean, modes and stochastic coefficients.

3.3.4 True Solution Generation

Truth is generated using a deterministic run with a particular set of parameter values which lie within the realization space. The governing equations are the deterministic version of the physical, LTL-biological and the fish model. Boundary conditions and velocity field are either deterministic, or corresponding to a particular realization from the stochastic run. The velocity and tracer fields are also initialized by reconstructing the realization from the stochastic fields at $t = 0$ (t is used for non-dimensional times), corresponding to the parameter values chosen for the true run. The numerical methods to solve the deterministic governing equations, domain ($\mathcal{D}$) layout, along with the non-dimensional domain and time discretizations ($\Delta x = \Delta y = \frac{1}{15}$, and $\Delta t = \frac{1}{240}$ are kept exactly the same as those used for evolving the stochastic DO equations.

3.3.5 Observations and Inference

Observations sparse in both space and time are taken from the generated true solution (section 3.3.4). Depending on the experiment, one of the biological tracer fields is observed at 8-15 locations. The observation locations are kept in the euphotic zone because below the euphotic zone there is not much dynamics happening. The observations happen at every 2 non-dimensional times. Observation error standard deviation matrix ($\sqrt{\boldsymbol{R}}$ in section 3.2.4) represents the confidence in the sensors, and in all the experiments, the sensors are considered independent of each other. The linear observation matrix $\boldsymbol{H}$ (section 3.2.4) is specified such that it identifies the concentration of the tracer field corresponding to the observation locations.

Also, the parameters related to the DO equations and the GMM-DO filter, for example, number of modes, monte-carlo samples, maximum number of GMM mixtures

to be trying to fit at every assimilation step, etc. were chosen based on experience and numerical tests. It was made sure that they are sufficient to capture the uncertainty associated with each of the experiments.

3.4 Application Results and Discussions

To demonstrate the capabilities of the Bayesian learning framework, we perform simultaneous estimations of state variables, parameters, and parameterizations in the coupled physical-biological-fish model using only very sparse observations. We employ so-called "identical twin experiments" in which observations are made from a simulated truth generated using a deterministic run with a particular set of parameter values which lie within the uncertain realization space.

3.4.1 Experiments 1: Uncertain hydrostatic physics

In the first set of experiments, for the physical model, uncertainty is in the initial conditions for the state variables and the eddy viscosity parameter (Λ_{Re}). We consider the Boussinesq coupling between momentum and density equation to be absent, with the density acting as a passive tracer. In the lower-trophic-level biological model, uncertainty is introduced by the ambiguity in the presence or absence of quadratic zooplankton mortality functional ($a \in \{0, 1\}$), along with the Ivlev grazing parameter (Λ). In the fish model, the uncertainty comes from the presence of uncertain primary production and the recruitment time (T_r) in the source term of the forage equation (equation 3.4) and from uncertain physical variables in the effective advection velocities (equation 3.5). The goal is to learn all the state variables fields, along with the uncertain parameters and parameterizations, through a few observations of the zooplankton field. These data are sparse in both space and time, with the observations only available at six locations every two non-dimensional times, starting at time $T = 3$ and ending at $T = 11$. The parameter values used in this experiment, adapted from the literature, are given in Table 3.1.

Table 3.1: Values of the parameters used in the coupled nonhydrostatic physical-biological-fish model. For the non-dimensionalization, the scalings used are: $N_T = 30\ mmol\, N\, m^{-3}$, $H = 50\ m$, $D = 1\ km$, and time-scale of 12.5 day.

Parameters	**Values**
Domain	
Horizontal length scale, D (km)	1
Vertical length scale, H (m)	50
Domain length, L (non-dim.)	20
Domain height, h (non-dim.)	2
Seamount center, X_c (non-dim.)	7.5
Physical Model	
Inlet velocity, U (cm/s)	1
Eddy viscosity, ν_E (m^2/s)	10
Inverse of Eddy viscosity based Reynolds number, Λ_{Re} (non-dim.)	unif(0.01, 1)
Diffusion constants in horizontal and vertical, $\mathcal{K}_x$ & $\mathcal{K}_z$ $(m^2/s$; same for all tracers, except fish density)	0.01 & 0.001
Reference temperature, T_o $(^\circ C)$	15
Coefficient of expansion, α $(kg/m^3/^\circ C)$	1.5×10^{-7}
LTL-Biological Model	
Light attenuation coefficient, k_w (m^{-1})	0.067
Slope of the P-I curve, α $((W\ m^{-2}\ day)^{-1})$	0.025
Surface available radiation, I_l^o $(W\ m^{-2})$	158.075
Phytoplankton maximum uptake rate, V_m (day^{-1})	1.5
Half-saturation for phytoplankton uptake of nutrients, K_u $(mmol\ N\ m^{-3})$	1
Phytoplankton specific mortality rate, Ξ (day^{-1})	0.1
Linear zooplankton mortality rate, Γ_1 (day^{-1})	0.145

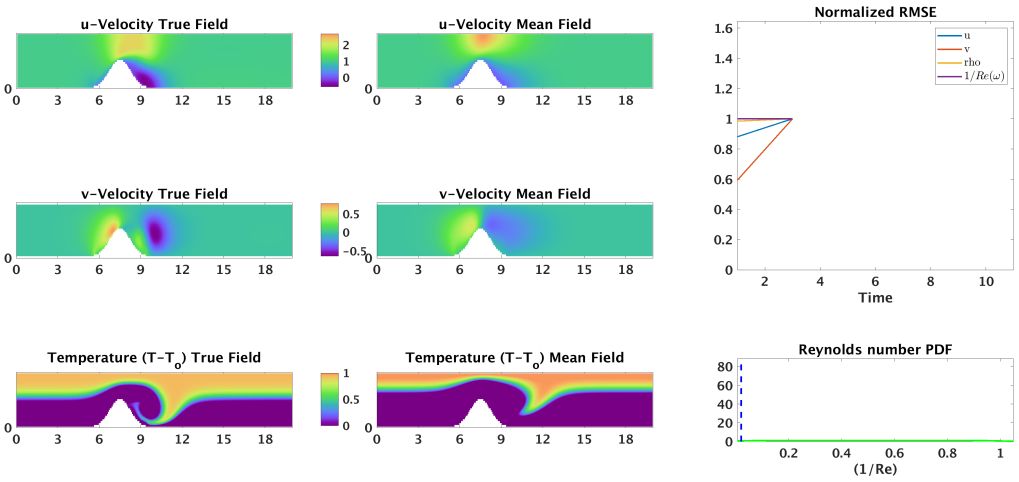
Presence or absence of quadratic zooplankton mortality term, a (non-dim)	unif{0, 1}
Quadratic zooplankton mortality rate, Γ_2 (day^{-1})	0.2
Zooplankton max grazing rate, R_m (day^{-1})	0.52
Ivlev constant, Λ $((mmol\ N\ m^{-3})^{-1})$	unif(0.1, 0.2)
Fraction of zooplankton grazing egested, γ	0.3
Fish Model	
Forage mortality, λ (yr^{-1})	4.6
Forage loss coefficient, m_r (day^{-1})	0.025
Recruitment time, T_r (day)	unif(75, 100)
Fish mortality coefficient parameter, λ_z (day^{-1})	0.8
Fish mortality coefficient parameter, λ_I (non-dim)	100
Gradient of habitat index proportionality coefficient, χ (m/day)	400
Fish recruitment rate, R $(\text{kg/m}^2\text{/day})$	4
Fish diffusion coefficient, D_x & D_y (m^2/s)	0.1 & 0.01
Others	
Number of realizations, N_{MC}	10,000
State being observed	Z
Observation error standard deviation, $(\sqrt{\boldsymbol{R}})$	0.05
Number of observation locations, N_Y	6
Observation start time (non-dim)	3
Time interval between observations (non-dim)	2
Observation end time (non-dim)	11

Figure 3-2 shows the prior of the system at $T = 3$, i.e., just before the first set of observations are available. There are many differences between the mean and true fields of all the state variables. The blue dotted line in the probability plots of the

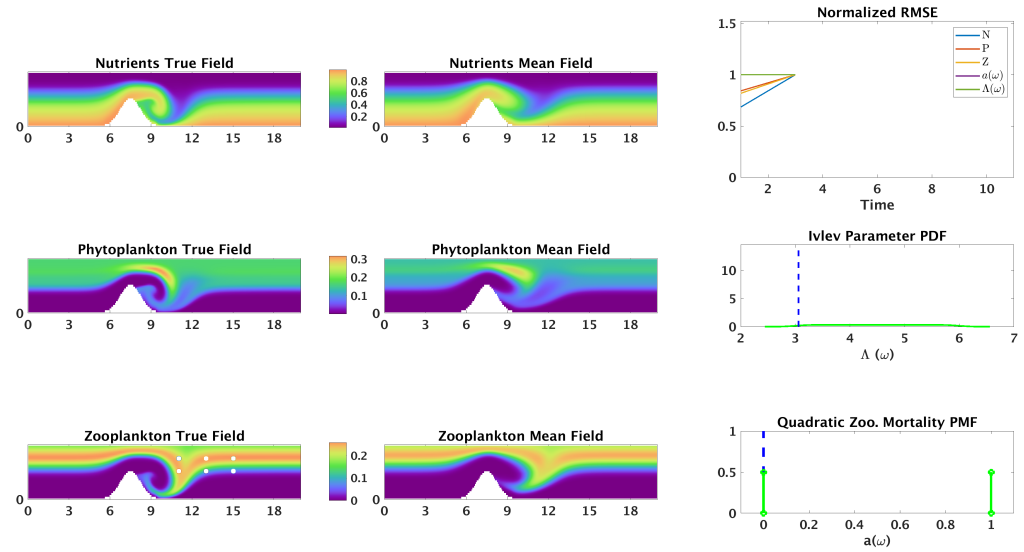
parameters marks the true non-dimensional values. The prior probabilities of these parameters are considered to be uniform within a certain range. A phytoplankton bloom develops in top-right of the seamount due to the upwelling of nutrients from the bottom, which causes an increase in the forage concentration. In-turn, the fish population increases. A vortex also starts to develop in the wake of the seamount. We provide the corresponding standard deviation fields in Figure 3-3. There exists a large amount of uncertainty in the exact location and size of both the bloom and the vortex.

In figure 3-4, we provide the posterior of the system after two observational episodes, i.e. at $T = 5$. By observing the zooplankton field, we are not only able to correct the biological model tracers and its parameters, but also the dynamics of the flow, as seen by the clustering of the Λ_{Re} distribution around its true value. Though we do not see a large correction in the fish model state variables, the probability distribution for the recruitment time (T_r) begins to approach the true value. We use the variation of Root Mean Square Error (RMSE) over time to judge performance. RMSE is the L_2 distance between the mean of the random variables in the stochastic run and the simulated truth. The RMSE value for each of the variables at every time is normalized by the corresponding RMSE value just before the first assimilation step. Hence, our findings are corroborated by the decrease in RMSE for the parameters and state variables (except the temperature field), and the fact that assimilating the first observation at $T = 3$ was not effective.

Finally, in figure 3-5, we present the posterior after 5 observational episodes at $T = 11$. We unambiguously learn all the parameter values from the data, even detecting the absence of quadratic mortality term from our NPZ model. We observe agreement between the mean and true fields for the velocities, NPZ tracers, and the forage. It is interesting to note that we make no correction to the temperature field. This is perfectly as expected, because temerature does not affect the biological tracers. Since in the present simulation, temperature is a passive tracer (because of no Boussinesq coupling), the zooplankton data contains no information about the temperature field, thus, it is not identifiable from the given data (also called the problem

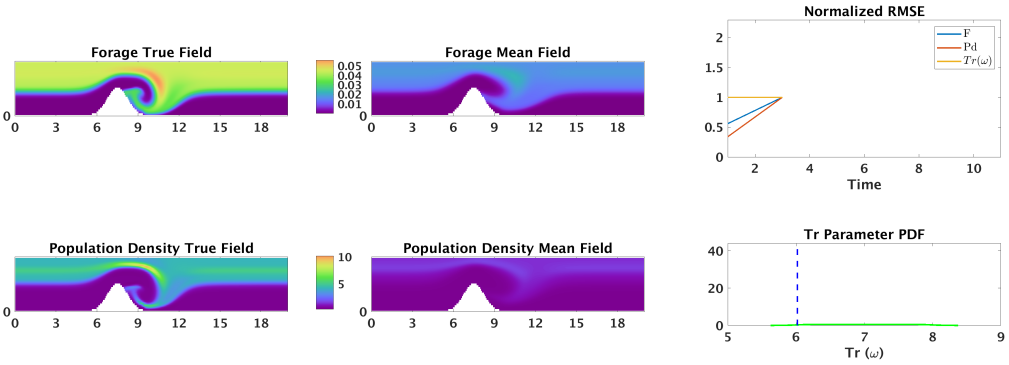


(a) Physics model



(b) LTL-Biological model

Figure 3-2: The prior state of the stochastic dynamical system used in the experiments-1, at $T = 3$ (i.e. just before the 1st observational episode). (a), (b), (c): The first two columns consist of the true (left) and mean (right) field of the state variables of the corresponding models. In the third column, the first plot shows the variation of normalized RMSE with time for various stochastic state variables and parameters. The remaining plot(s) contain the probability distribution of the respective uncertain parameters of $\Lambda_{Re}(\omega)$, $\Lambda(\omega)$, $a(\omega)$ (to learn the presence or absence of quadratic zooplankton mortality), and recruitment time $T_r(\omega)$. The white circles on the zooplankton true field mark the observation locations. *(Cont.)*



(c) Fish model

Figure 3-2: The prior state of the stochastic dynamical system used in the experiments-1, at $T = 3$ (i.e. just before the 1st observational episode). (a), (b), (c): The first two columns consist of the true (left) and mean (right) field of the state variables of the corresponding models. In the third column, the first plot shows the variation of normalized RMSE with time for various stochastic state variables and parameters. The remaining plot(s) contain the probability distribution of the respective uncertain parameters of $\Lambda_{Re}(\omega)$, $\Lambda(\omega)$, $a(\omega)$ (to learn the presence or absence of quadratic zooplankton mortality), and recruitment time $T_r(\omega)$. The white circles on the zooplankton true field mark the observation locations.

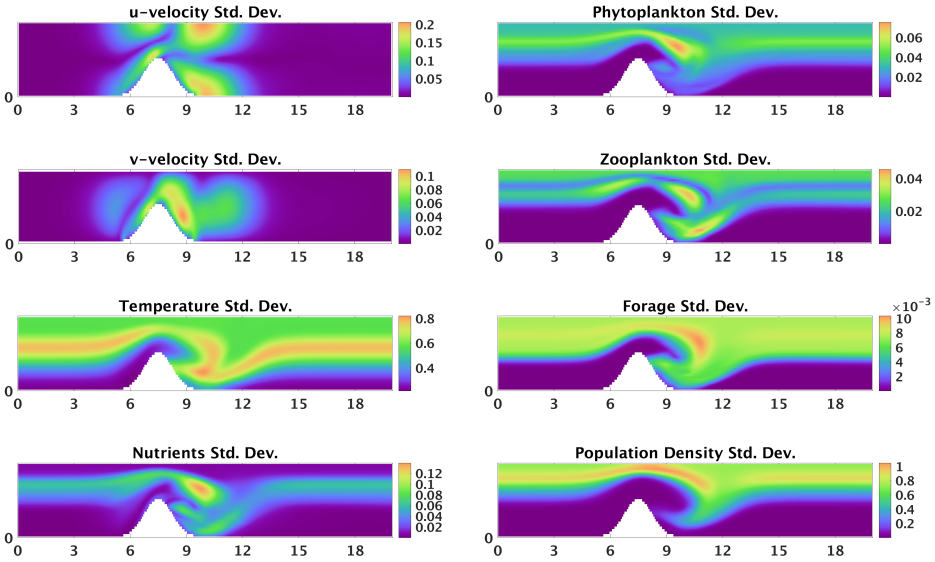
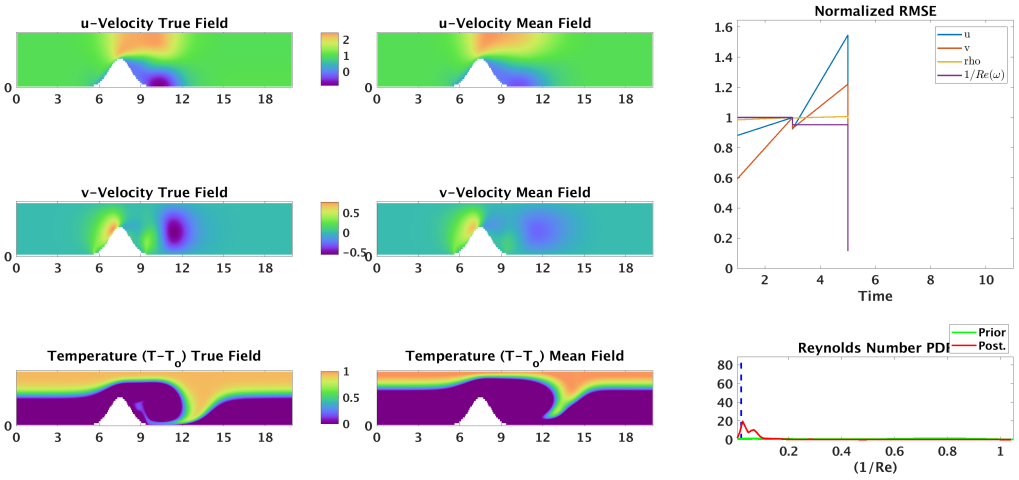
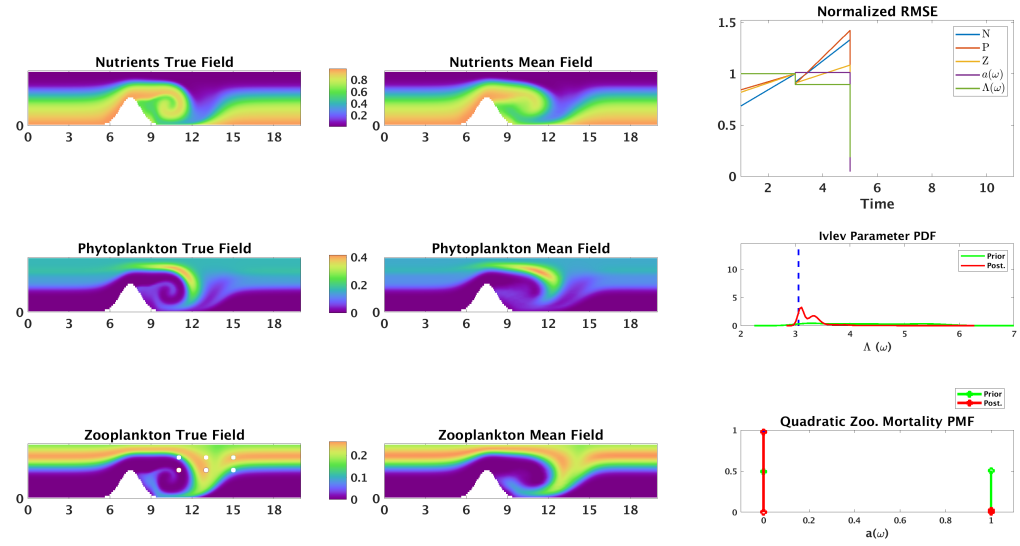


Figure 3-3: The prior standard deviation of the stochastic dynamical system used in the experiments-1, at $T = 3$ (i.e. just before the 1st observational episode).

of identifiability, see section 5.4.3). The fish population density is directly affected by the temperature field, about which we have no information, but is indirectly related



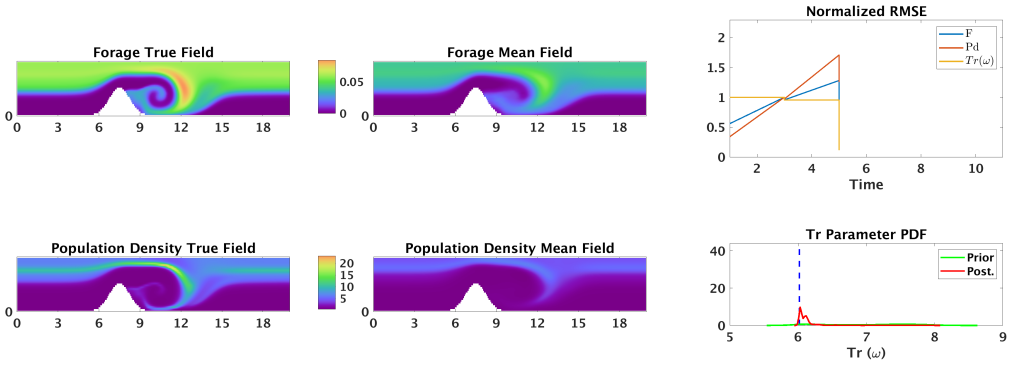
(a) Physics model



(b) LTL-Biological model

Figure 3-4: Posterior state of the stochastic dynamical system used in the experiment-1, at $T = 5$ (i.e. just after the 2nd observational episode). Description same as that of figure 3-2. *(Cont.)*

to zooplankton through the primary production and forage; hence, we are able to learn the fish population from zooplankton data through the somewhat weak link of primary production and forage.



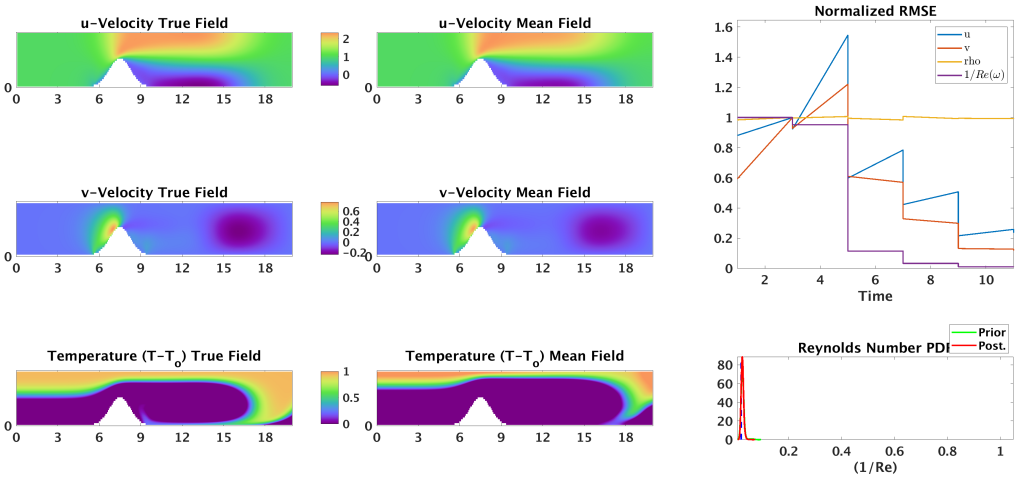
(c) Fish model

Figure 3-4: Posterior state of the stochastic dynamical system used in the experiment-1, at $T = 5$ (i.e. just after the 2nd observational episode). Description same as that of figure 3-2.

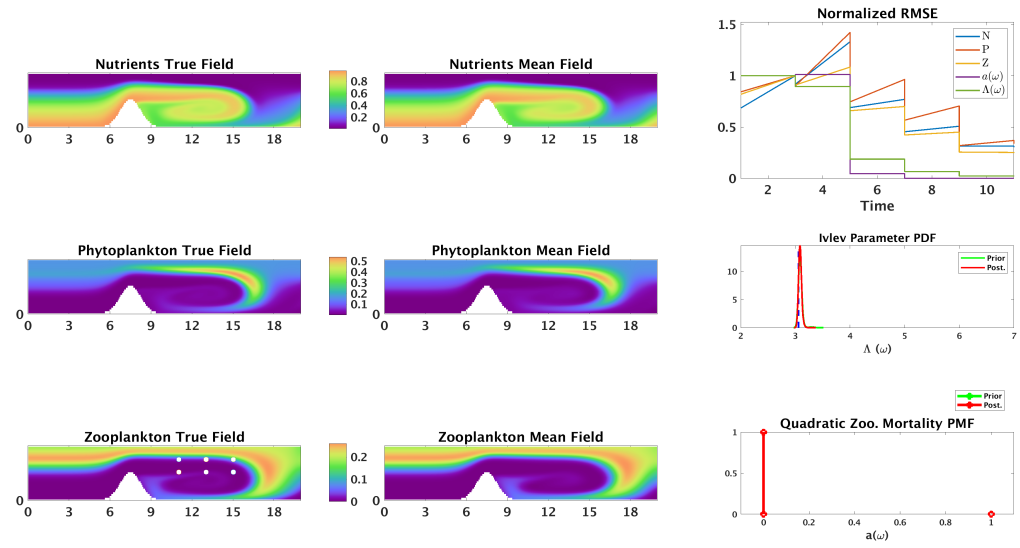
3.4.2 Experiments 2: Deterministic nonhydrostatic physics

In these experiments, to show how the overall learnability of the fish model from zooplankton observations can be improved, we turn on the full temperature-momentum Boussinesq coupling. Thus, leading to more complex nonhydrostatic dynamics including internal waves, however, we consider the physical model to be known, i.e. deterministic.

In figure 3-6, we provide the posterior state of the system directly after 10 observational episodes at $T = 21$, and as expected, there is a better match between the GMM-DO mean fields and the true fields for the fish model tracers. The probability distribution for the T_r parameter has also become concentrated around its true value. The effects of the known internal lee waves are clearly visible on all coupled physics-LTL-fish fields. As a result, the forage field is more challenging to learn than before. Even though the physics is known, due to the complicated nature of this flow dynamics, a larger number of observational episodes were indeed needed to achieve the learning objectives.



(a) Physics model

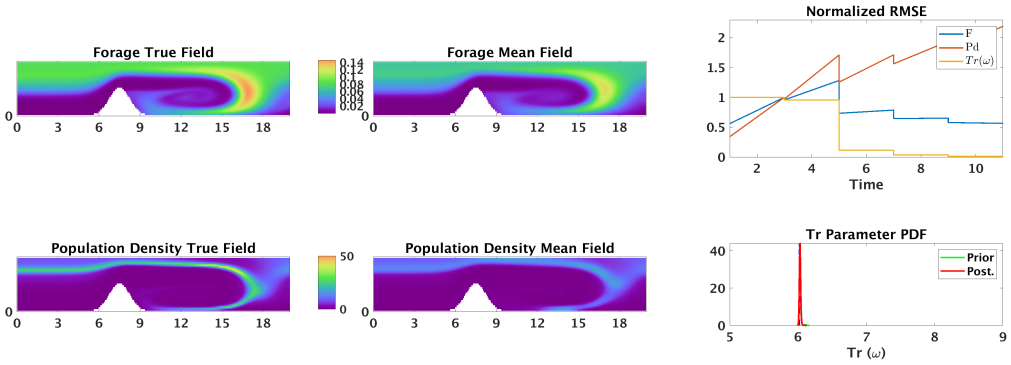


(b) LTL-Biological model

Figure 3-5: The posterior state of the coupled physical-biological-fish model based stochastic dynamical system used in the experiments-1, at $T = 11$ (i.e. just after the 5th observational episode). Description same as that of figure 3-2. *(Cont.)*

3.4.3 Experiments 3: Uncertain nonhydrostatic physics with model discovery

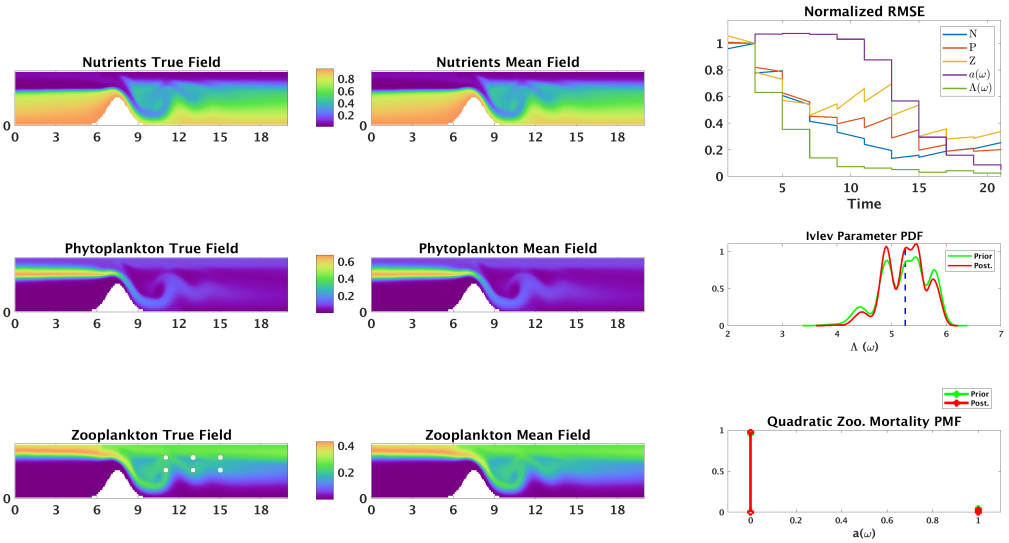
In our last set of experiments, for the physical model, we consider uncertainty in the initial velocity fields, vertical stratification of temperature, and the eddy viscos-



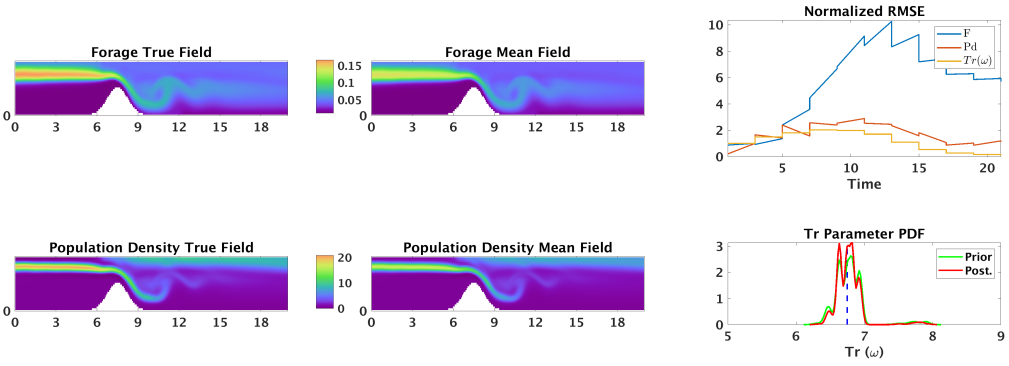
(c) Fish model

Figure 3-5: The posterior state of the coupled physical-biological-fish model based stochastic dynamical system used in the experiments-1, at $T = 11$ (i.e. just after the 5th observational episode). Description same as that of figure 3-2.

ity. Also, the full temperature-momentum Boussinesq coupling is turned-on leading to more complex dynamics including internal waves. For the LTL NPZ model, we consider the zooplankton mortality to be completely unknown, thus, left to be discovered by the Bayesian learning machine. While for the HTL fish model, we only consider the recruitment time parameter to be unknown within a certain range. Similar to the experiments - 3 in chapter 2, once again, we assume the domain and range of the zooplankton mortality function to be $[0, 0.3]$ and $[0, 0.08]$, respectively. The unknown function is assumed to be composed of 10 continuous and stochastic piece-wise quadratic segments. In the current experiment, we will observe nutrients at 15 observation locations from a simulated truth which exists in the initial prior. The observations start at $T = 1$, and come in every two non-dimensional times till $T = 11$. For the prior at $T = 1$, we can notice that the mean of the state variables of our stochastic prediction are quite different than the true state variables. For the unknown function, we do not assume anything other than a bounded domain and range, and we allow our function to take any form. From the RMSE, we can notice, that as more and more observations come in, we are able to meet all our learning objectives. The mean and true fields of the state variables look very similar at $T = 11$ posterior, along with the parameter pdfs getting concentrated around the true value. However,



(a) LTL-Biological model



(b) Fish model

Figure 3-6: The posterior state of the coupled physical-biological-fish model based stochastic dynamical system used in the experiments-2, at $T = 21$ (i.e. just after the 10th observational episode). Description same as that of figure 3-2.

the peak of the Reynolds number pdf misses the true value, because the algorithm actually learns the value of inverse of Reynolds number, thus making it very sensitive to small errors in learning. All the zooplankton mortality function realizations gets concentrated around the truth which is a sigmoid. For Z in the range 0.25 to 0.30, the machine is not able to learn the functional form because none of the observations lie in that range. Such problems can be mitigated by incorporating more prior knowledge about the system while selecting the function space to search in.

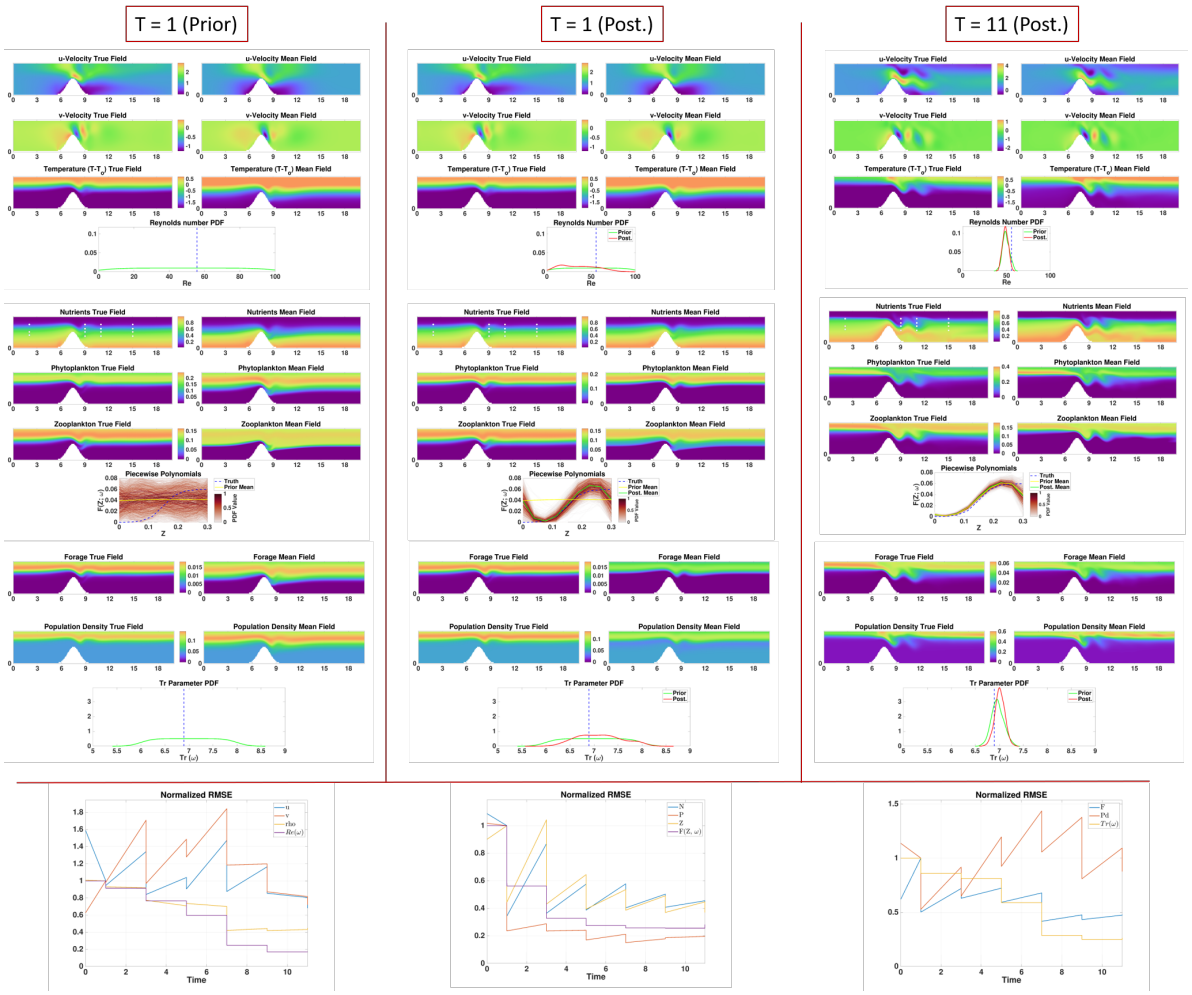


Figure 3-7: The prior state of the stochastic dynamical system used in the experiment-2, at $T = 1$ (i.e. just before the 1^{st} observational episode), followed by the posterior at $T = 1$, and posterior at the final time of $T = 11$ (after the 6 observational episodes). Every column consists of (left) and mean (right) field of the state variables of the corresponding models. At the bottom of the state variable plot of each of the physics, LTL biology, and fish model; pdf of Reynolds number, ensemble of function realizations (colored according to their respective normalized probability density values), and pdf of recruitment time parameter are respectively plotted. The bottom-most row consists of normalized RMSE variation with time for each model. The white circles on the nutrients true field mark the observation locations, and the dotted lines the true zooplankton mortality function, and other parameter values.

3.5 Summary

We provided a comprehensive overview of the current state-of-the-art in fish modeling. Taking into account different sources of uncertainty including initial conditions, parameters, and parameterizations, we numerically integrated a stochastic coupled nonhydrostatic ocean physical-biological-fish dynamical model in an idealized domain, using the Dynamically Orthogonal (DO) methodology, an adaptive reduced-dimension stochastic modeling technique for efficient uncertainty evolution. The uncertain functional forms (parameterizations) are handled using the novel methodologies of stochastic special parameters and stochastic piece-wise polynomial functions developed in the previous chapter. As a part of our PDE-based Bayesian learning framework, we then use the GMM-DO filter to perform a nonlinear inference of high-dimensional states containing multidisciplinary unknown states, parameters, and parameterizations in a series of experiments of varying complexities. The experimental setup consists of a uncertain nonhydrostastic variable-density Boussinesq flow past a seamount, and we also demonstrate interdisciplinary learning. The learning results are promising for use with realistic fish modeling simulations.

Chapter 4

Bayesian Discovery of Ocean Acidification Models Using Real-World Data

Monitoring, quantifying, and predicting the three-dimensional and time-dependent ocean acidification processes, from the atmospheric exchanges and river discharges to the ocean interior, over days to decades, remains a fascinating observational, theoretical, and modeling challenge. This challenge is the long-term driver of our "Bayesian Intelligent Ocean Modeling and Acidification Prediction Systems" (BIOMAPS) research (`http://mseas.mit.edu/Research/BIOMAPS/`). Ocean acidification (OA), or the progressive decrease in pH of seawater, is caused primarily by excess atmospheric CO_2 and is linked to climate change [116, 117, 118]. Its chemical perturbations are expected to be larger in coastal regions than on global average [119, 120]. In the Gulf of Maine and Massachusetts Bay regions, the shellfish growth and reproduction are affected by coastal acidification, with negative impacts on crustaceans (lobsters, crabs) and both wild and farmed mollusks (scallops, oysters, clams, mussels), hence also on major industries and employment sources [121]. Improving the monitoring, modeling, and forecasting of regional OA is urgent.

The overarching goal of our research is to develop and demonstrate principled Bayesian intelligent ocean modeling and acidification prediction systems that discrim-

inate among and infer new OA models, rigorously learning from data-model misfits and accounting for uncertainties, so as to better monitor, predict, and characterize OA over time scales of days to months in the Massachusetts Bay and Stellwagen Bank regions.

4.0.1 Problem Statement

One-dimensional (1-D) models for the ocean biogeochemistry provide an important tool for the study, understanding, and modeling of interplay of various (often elusive) physical, biological, and carbonate processes [122]. However, they are marred by a variety of issues, such as, missing processes and interactions, seasonal and regional variability in parameter values, multiple candidate functional representations of the same process by different scientists, etc. Thus, the goal of this work is to use and showcase our novel Bayesian learning paradigm to simultaneously estimate states, parameters, and functional form of missing processes with quantifiable uncertainty, using model-data misfits between an existing coupled physical-biological-carbonate model and observed real world *in-situ* OA data in the Gulf of Maine (GoM) during a research cruise in the middle of August, 2012.

4.1 Modeling Methodology

In what comes next, we first describe the 1-D coupled physical-biological-carbonate model used in the present study, followed with the different data sources available in the GoM region during the time of interest. We then describe the procedures for estimating initial state uncertainty, parameter values, relevant physical conditions, etc. which are important to ensure that the model simulations provide an accurate and representative prior estimate for state uncertainty.

4.1.1 Coupled physical-biological-carbonate model

We will use a model similar to the Hadley Centre Ocean Carbon Cycle (HadOCC) model [123], where the biological part will consist of a modified version of a four-component system (nutrients (N), phytoplankton (P), zooplankton (Z), and detritus (D)) developed by Tian *et al.* [77] for the Gulf of Maine, along with dissolved inorganic carbon (DIC), and total alkalinity (TA) for the carbonate part. The NPZD model is given by,

$$\begin{aligned}
\frac{dN}{dt} &= -U_P + \lambda G_Z + \varepsilon f_T(T(z))D \ , \\
\frac{dP}{dt} &= U_P - G_Z - m_P f_T(T(z))P^2 + s_P \frac{\partial P}{\partial z} \ , \\
\frac{dZ}{dt} &= \gamma G_z - m_Z f_T(T(z))Z^2 \ , \\
\frac{dD}{dt} &= (1-\gamma-\lambda)G_Z + m_P P^2 + m_Z Z^2 - \varepsilon f_T(T(z))D + s_D \frac{\partial D}{\partial z} \ ,
\end{aligned} \tag{4.1}$$

with U_P representing the phytoplankton growth, regulated by nitrogen limitation based on Michaelis-Menten kinetics ($f_N(N)$), photosynthetically active radiation ($f_I(I(z))$), and temperature limitation ($f_T(T(z))$); G_Z representing the zooplankton grazing; and $M_Z(Z)$ representing the zooplankton mortality; all given by,

$$\begin{aligned}
U_P &= \mu_{max} f_N(N) f_I(I(z)) f_T(T(z)) P \ , \qquad f_N(N) = \frac{N}{N+K_N} \ , \\
f_I(I(z)) &= (1 - \exp(\alpha I(z)/\mu_{max}))\exp(-\beta I(z)/\mu_{max}) \ , \\
I(z) &= I_0 \exp\left(-k_W z - k_P \int_{-z}^{0} P dz - k_D \int_{-z}^{0} D dz\right) \ , \\
G_Z &= \frac{g_{max} Z P^2}{P^2 + K_P^2} \ , \qquad f_T(T(z)) = \exp(-a|T(z) - T_{opt}|) \ .
\end{aligned} \tag{4.2}$$

In the above equations, the concentration of biological variables is in $mmol\ N\ m^{-3}$ (measured in nitrogen), z is depth, and the other parameters are: μ_{max} is the maximum growth rate of phytoplankton; K_N is the half-saturation constant; α is the light-growth slope; β is the inhibition coefficient; I_0 is photosynthetically active radiation (PAR) at the sea surface; k_W is the attenuation coefficient of water; T_{opt}

is optimal temperature for phytoplankton growth; a is the temperature coefficient; g_{max} is the zooplankton maximum grazing rate; K_P the half-saturation constant for zooplankton grazing; γ is the assimilation coefficient; m_z the zooplankton mortality coefficient; m_p the phytoplankton mortality coefficient; λ is the active respiration zooplankton expressed as a fraction of grazing; s_P and s_D are the phytoplankton and detritus sinking speeds, respectively; and ε is the remineralization rate of detritus. The carbon in the system is coupled with the nitrogen by fixed carbon-nitrogen ratios, C_P, C_Z, and C_D,

$$\begin{aligned} \frac{d(DIC)}{dt} &= -C_P\frac{dP}{dt} - C_Z\frac{dZ}{dt} - C_D\frac{dD}{dt} - \gamma_c C_P U_P \ , \\ \frac{d(TA)}{dt} &= -\frac{1}{\rho_w}\frac{dN}{dt} - \frac{2\gamma_c C_P U_P}{\rho_w} \ . \end{aligned} \tag{4.3}$$

Neither DIC or TA has any effect on the biology because phytoplankton growth is not carbon limited. The last term in the DIC equation represents the precipitation of calcium carbonate to form shells and other hard body parts, which subsequently sink below the euphotic zone, also known as "hard flux". This flux is modeled to be proportional (and additional) to the uptake of carbon for primary production. Also, the chemistry dictates the decrease in total alkalinity by two molar equivalents for each mole of carbonate precipitated. In general, TA is measured in $\mu mol\ kg^{-1}$ while the biological variable (e.g. N) are measured in $mmol\ m^{-3}$, thus, we divide the right-hand-side (RHS) of the TA equation with density of the sea-water (ρ_w).The units of DIC concentration are $mmol\ m^{-3}$.

The above biological and carbonate models are often coupled with physical models to introduce both spatial and temporal components. For our experiments, we use a 1-D diffusion-reaction PDE with vertical eddy mixing parameterized by the operator, $\partial/\partial z\,(K_z(z,M)\partial/\partial z(\bullet))$, where K_z is a dynamic eddy diffusion coefficient. A mixed layer of varying depth ($M = M(t)$) is used as a physical input to the OA models. Thus, each biological and carbonate state variable $B(z,t)$ is governed by the following

non-autonomous PDE,

$$\frac{\partial B}{\partial t} = S^B + \frac{\partial}{\partial z}\left(K_z(z, M(t))\frac{\partial B}{\partial z}\right) , \tag{4.4}$$

$$K_z(z, M(t)) = K_{z_b} + \frac{(K_{z_0} - K_{z_b})(\arctan(-\gamma_t(M(t) - z)) - \arctan(-\gamma_t(M(t) - D_z)))}{\arctan(-\gamma_t M(t)) - \arctan(-\gamma_t(M(t) - D_z))} , \tag{4.5}$$

where K_{z_b} and K_{z_0} are the diffusion at the bottom and surface respectively, γ_t is the thermocline sharpness, and D_z is the total depth. The 1-D model and parameterizations are adapted from Eknes and Evensen, 2002 [4]. They simulate the seasonal variability in upwelling, sunlight, and biomass vertical profiles. We simulate the photosynthetically-available radiation (PAR) $I_0(t)$ using the instantaneous incoming radiation model proposed by Peixoto and Oort, 1992 [124]. First, the incident solar radiation at the top of the atmosphere (Q_0) is given by,

$$\begin{aligned} Q_0 &= \max\left[S_c\left(\frac{d_m}{d}\right)^2(\sin\phi\sin\delta + \cos\phi\cos\delta\cos h)\right] \\ \left(\frac{d_m}{d}\right)^2 &= 1 + 0.035\cos\left[\frac{2\pi}{365}(6 - yd)\right] \\ \delta &= -\frac{23.45\pi}{180}\cos\left[\frac{2\pi}{365}(355 - yd)\right] \\ h &= 2\pi\left(\frac{t_{GMT}}{24} + \frac{\psi}{360} - \frac{1}{2}\right) \end{aligned} \tag{4.6}$$

where $S_c = 1360W\ m^{-2}$ is the solar constant; d_m is the earth's mean distance from the sun; d is the earth's current distance from the sun; ϕ is the latitude (radians); δ is the solar declination; h is the hour angle, having a value of zero at local solar noon; yd is the year-day (including fractional part); t_{GMT} is the (decimal) time of day, Greenwich Mean Time (e.g. at 1415 GMT, $t_{GMT} = 14.25$); and ψ is the longitude (degrees). Further, it is assumed that only only 76% of the radiation penetrates the atmosphere on a cloudless day and only 45% of it is available as PAR [62]. Thus,

PAR (in $W\ m^{-2}$) is given by,

$$I_0(t) = 0.45 \times 0.76 Q_0(t)\ . \tag{4.7}$$

It should be noted that t will need to be converted to yd and t_{GMT} in equation 4.6.

Apart from the effects of processes such as nutrient uptake / mineralisation, nitrification or denitrification on TA as modeled by equation 4.3, TA is also strongly controlled by factors such as precipitation, evaporation, water mass mixing, carbonate dissolution, and precipitation [125]. Thus, TA is often decomposed into two components, diagnostic and prognostic [126]. The prognostic component is fully advected and diffused by the circulation model and simulates the variability of TA due to all the biological processes, riverine input, etc. While the diagnostic component is calculated using a linear regression between TA and salinity (S). For the diagnostic part, we use an empirical linear model optimized for the Gulf of Maine and valid at depths (Dr. Patrick J. Haley Jr., *pers. comm.*),

$$TA = \begin{cases} (198.10 + 61.75S)/1000, & S < 32.34 \\ (744.41 + 44.86S)/1000, & S \geq 32.34 \end{cases} \tag{4.8}$$

where TA is in $mmol\ kg^{-1}$ and S in PSU (practical salinity unit).

4.1.2 Data

The main region of interest for this study is the Gulf of Maine (GoM). Thus, we utilize a number of different data sources in the GoM region for the purposes of model initialization, parameterization, oceanographic analysis, and data assimilation.

GOMECC-2: The primary source of OA data is the second Gulf of Mexico and East Coast Carbon (GOMECC-2; [127]) Cruise on board the R/V Ronald H. Brown which happened in July-August, 2012. It started from Miami (July 21, 2012), into the Gulf of Mexico and then along the East US coast to its end at Boston (August 13, 2012). The effort was in support of the coastal monitoring and research objectives

of the NOAA Ocean Acidification Program (OAP). In total, 7 observations were made in the GoM region on August 12 & 13, 2012, and the measured variables of interest included temperature (T), salinity (S), nitrate (NO_3), chlorophyll-a (Chl-a), and total alkalinity (TA). We are only interested in modeling the off-shore ocean acidification, thus, we will not use the two profiles closest to the coast because they spend too much time in the Maine coastal current (MCC). Also, the one profile farthest to the right experiences very different ocean conditions, thus, is not relevant for our analysis. See figure 4-1 for reference.

GTSPP buoy: A buoy part of the global temperature and salinity profile program (GTSPP; [128]) and located just top-left ($70.43°W$, $43.18°N$) to the relevant GOMECC-2 data profiles (figure 4-2). It was active during the months of July and August in 2012, and made hourly observations of T, S, and velocity at depths of $1m$, $20m$, and $50m$. We also have access to wind velocities and wind-stress at the buoy location [129].

Sea surface temperature (SST): SST images collected by the Advanced High Resolution Radiometer (AVHRR) on the NOAA polar-orbiting satellites, and processed by the Ocean Remote Sensing Group at Applied Physics Laboratory, Johns Hopkins University [130]. We will utilize images for the Northern Gulf Stream region observed between July 21 - August 13, 2012.

World ocean database (WOD): We will utilize *in-situ* synoptic WOD data of all years (until 2018; [131]), for the variables of S, NO_3, and Chl-a, and the months of July and August. Data is selected in the region surrounding the relevant GOMECC-2 profiles and not very near to the coast (excluding MCC region), as shown in figure 4-2. The profiles present in the WOD which meet our location criterion, are further cleaned manually to remove extreme outliers and which are non-physical / biological. It should be noted, that the WOD data used did not contain GOMECC-2 profiles.

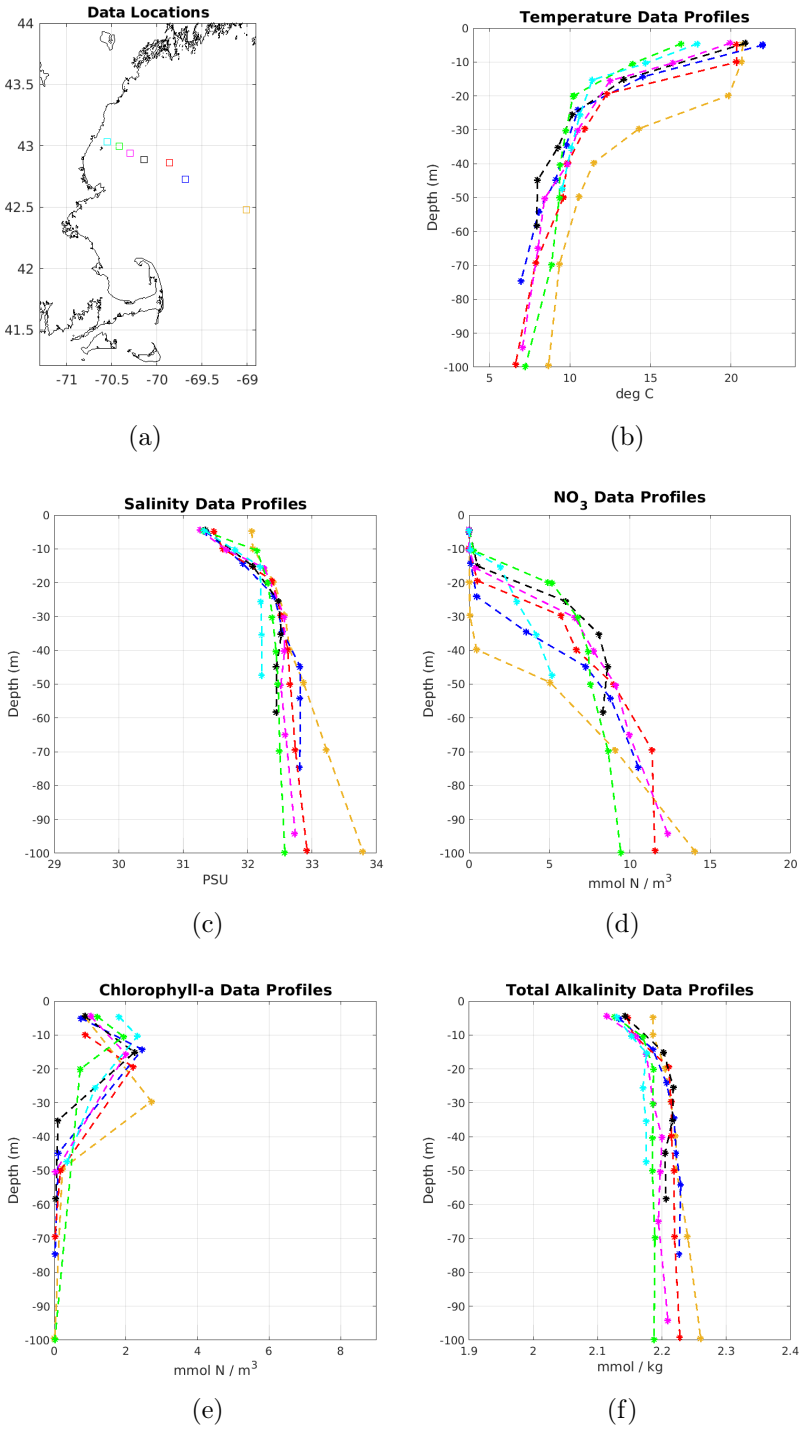


Figure 4-1: Data collected at 7 observation locations in the Gulf of Maine during the second Gulf of Mexico and East Coast Carbon (GOMECC-2) cruise. *(a):* Data locations; *(b):* Temperature; *(c):* Salinity; *(d):* Nitrate; *(e):* Chlorophyll-a; and *(d):* Total alkalinity data profiles. Color correspondence exists between data locations and profiles.

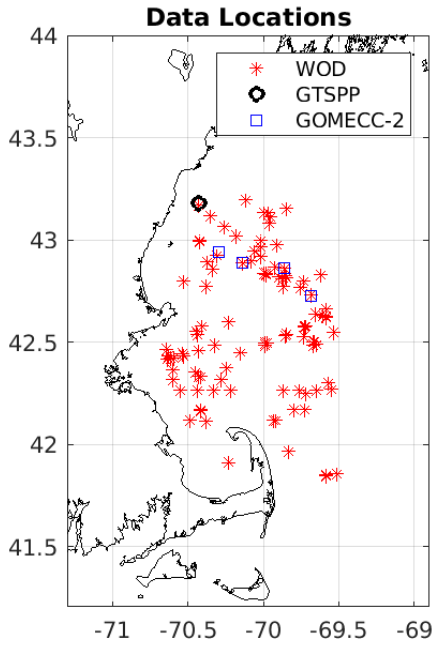


Figure 4-2: Locations of data profiles belonging to different data sources that were actually used for model initialization and assimilation. *WOD* stands for world ocean database, *GTSPP* for global temperature and salinity profile program, and *GOMECC-2* for Gulf of Mexico and East Coast Carbon #2 cruise.

4.1.3 Initialization

Historical state uncertainty: To compute the historical uncertainty for the months of July and August of all the biological and carbonate states, we start with the WOD data profiles of the observed variables of S, NO_3, and Chl-a as described in section 4.1.2 and shown in figure 4-3. Due to lack of WOD data profiles where all the three variables were observed simultaneously, we create joint vertical empirical orthogonal functions (EOFs; [132]) for the pairs of $\{S, Chl\text{-a}\}$, and $\{S, NO_3\}$ as shown in figure 4-4. Next, our goal is to generate joint random realizations for the three observed variables. We first create random joint realizations of S and Chl-a using the corresponding joint modes and eigenvalues of $\{S, Chl\text{-a}\}$ data pairs. Followed by which, we solve a system of linear equations to find coefficients corresponding to the joint modes of $\{S, NO_3\}$ which will create the same salinity realization as obtained earlier in the least-square sense. Using the set of coefficient and joint modes of $\{S, NO_3\}$ data pairs, we can compute the matching NO_3 realization. This procedure could also be done in the other order, using $\{S, NO_3\}$ first, followed by $\{S, Chl\text{-a}\}$. We create 1000 Monte-Carlo (MC) realizations, shown in figure 4-5. Realizations for the non-observed variables are computed from those of the observed ones using approximate relationships provided in table 4.1 and adopted from Beşiktepe *et al.*, 2003 [62]. We also apply a Savitzky-Golay filter [133] to each realization for smoothing them out.

Parameter values: The parameter values for the biological model are adopted from either Tian *et al.*, 2015 [77] or Beşiktepe *et al.*, 2003 [62], as both the studies provide values optimized for the GoM region. Parameter values for the carbonate model are adopted from Palmer and Totterdell, 2001 [123], while that for the physical model by Dr. Patrick J. Haley Jr. (*pers. comm.*). A comprehensive list of all the parameter values used in the experiment are listed in table 4.2.

Physics: For the time period of interest, the mixed layer depth is estimated to be $4m$ based on a root-mean-square wind stress of 0.4 $dynes/cm^2$ and an Ekman

Unobserved Variables	**Relation to Observed Variables**
$N(z;\omega)$	$NO_3(z;\omega)$
$P(z;\omega)$	$(C_{Chl}/(12 \times C_N)) \times Chl\text{-a}(z;\omega)$
$Z(z;\omega)$	$0.5 \times P(z;\omega)$
$D(z;\omega)$	$0.05 \times P(z;\omega)$
$DIC(z;\omega)$	$C_N \times NO_3(z;\omega)$
$TA(z;\omega)$	$\begin{cases} (198.10 + 61.75S(z;\omega))/1000, & S < 32.34 \\ (744.41 + 44.86S(z;\omega))/1000, & S \geq 32.34 \end{cases}$

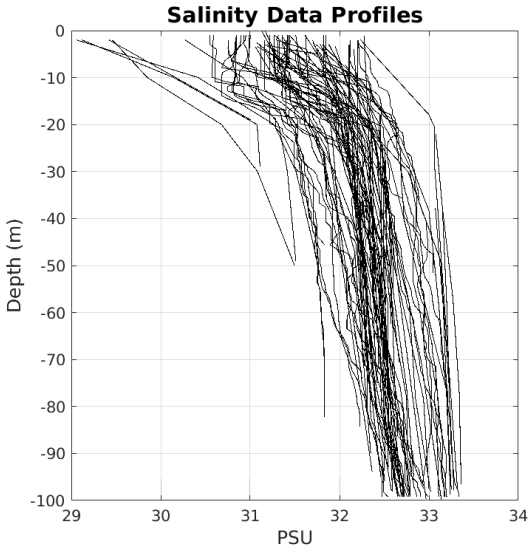
Table 4.1: Relationships between realizations of different observed and unobserved variables for initialization. ω is the realization index and z is the depth. For parameter definitions and values, see table 4.2.

factor of 0.06, and the transition from the mixed layer diffusion to the much smaller background value by $8m$ (Dr. Patrick J. Haley Jr., *pers. comm.*). The variation of vertical diffusion coefficient ($K_z(z, M(t))$) is assumed constant in time and provided in figure 4-6(a). The photosynthetically active radiation (PAR) is computed using equations 4.6 & 4.7 and at the centroid of the GOMECC-2 data locations, ($70°W$, $42.85°N$). Further, the mean temperature and salinity profiles computed from the GOMECC-2 data are used in model parameterizations when ever needed and treated to be deterministic. See figure 4-6.

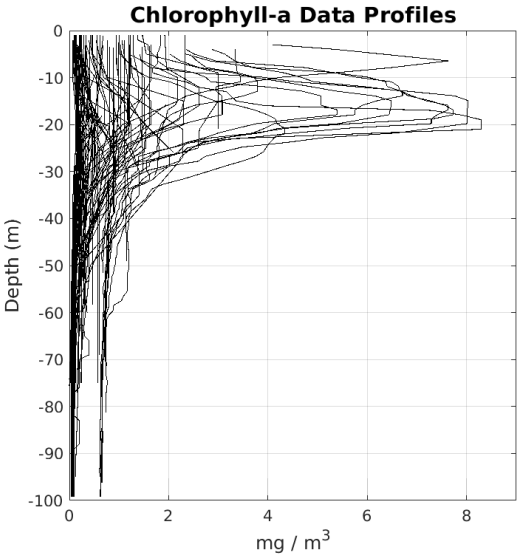
Boundary conditions: A Neumann zero boundary condition is specified at the bottom to let the sinking biomass / carbonate exit the domain without any hinderence. While on the top, a Robin condition is specified to help remember the initial concentrations in order to compensate for the lack of surface forcing, and at the same time allowing for some adjustment. This also removes singularity from the PDE system. The boundary conditions are given by,

$$\begin{aligned} \frac{1}{2}B(z,t) + \frac{\partial B(z,t)}{\partial z} &= 0, \quad \text{at } z = 0\ , \\ \frac{\partial B(z,t)}{\partial z} &= 0, \quad \text{at } z = D_z\ , \end{aligned} \tag{4.9}$$

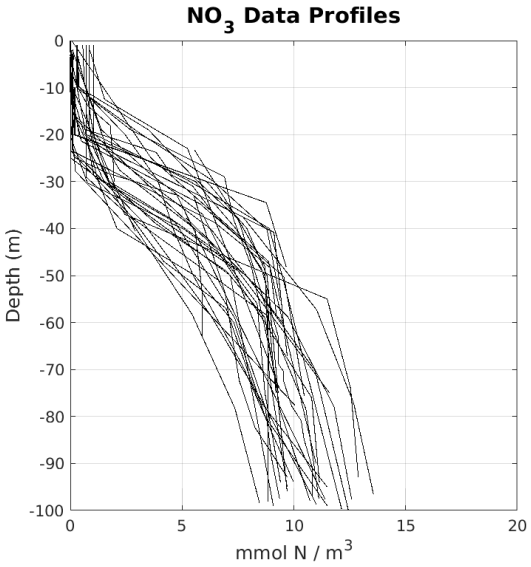
where $B \in \{N, P, Z, D, DIC, TA\}$ and valid at all times.



(a) Salinity

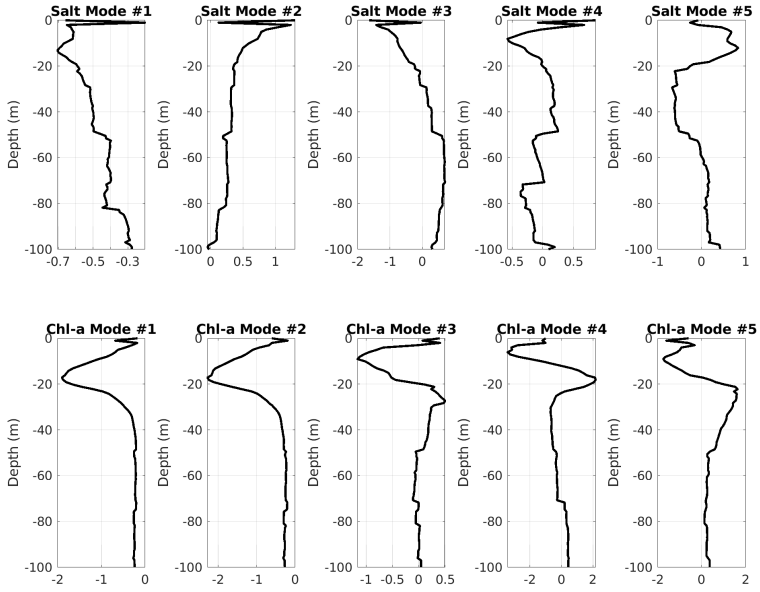


(b) Chlorophyll-a

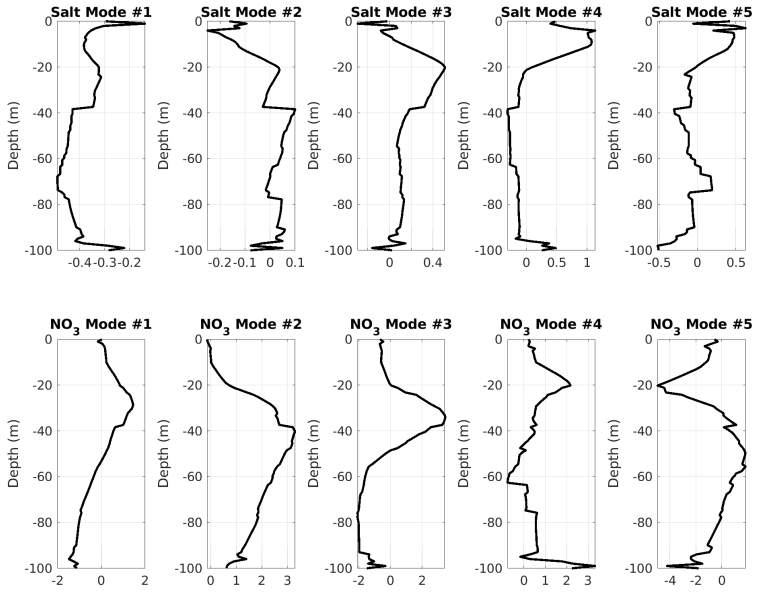


(c) Nitrate

Figure 4-3: World ocean database (WOD) data profiles observed in the months of July/August, in the area of interest, and used to create initial state uncertainty. For the corresponding data locations, see figure 4-2.

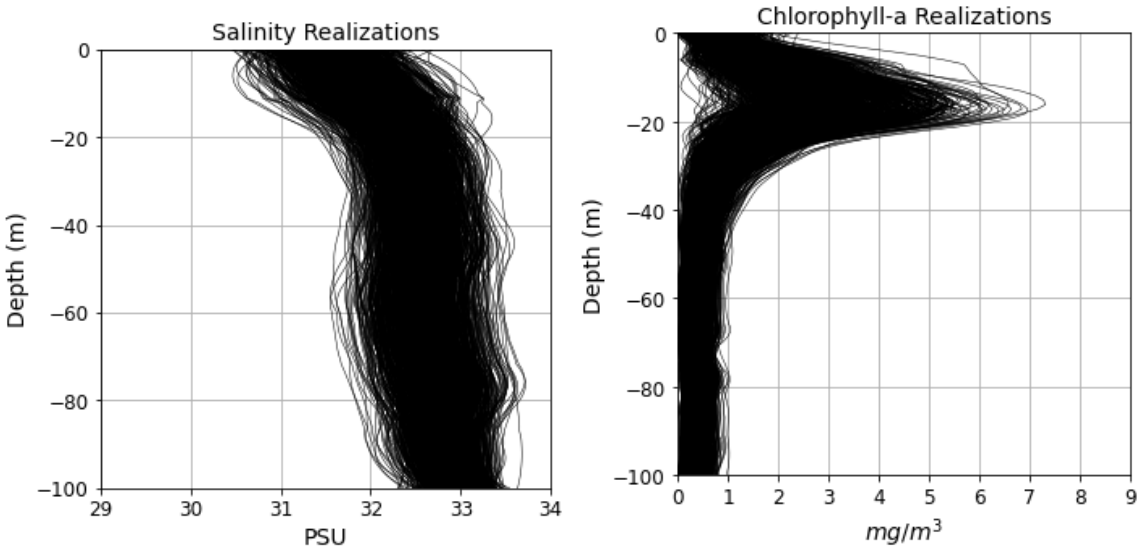


(a)



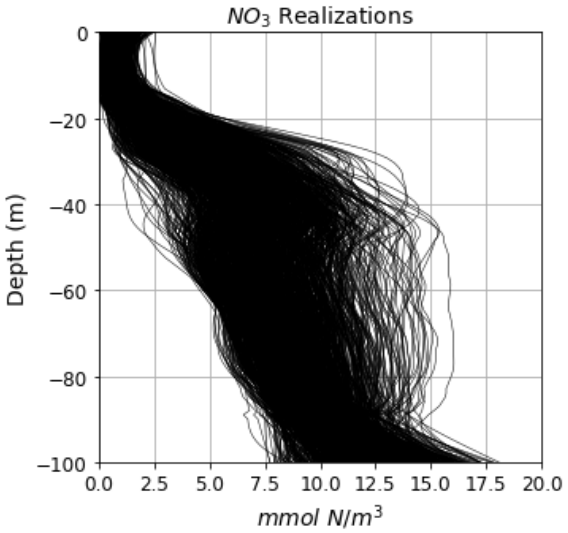
(b)

Figure 4-4: Joint vertical EOFs (empirical orthogonal functions) corresponding to different pairs of observed variables created using WOD data profiles (figure 4-3). Only the top 5 modes for each case are provided. *(a):* Observed variables, salinity (S) and chlorophyll-a (Chl-a); *(b):* Observed variables, S and nitrate (NO_3).



(a) Salinity

(b) Chlorophyll-a



(c) Nitrate

Figure 4-5: The created ensemble of realizations for different observed variables, representing historical uncertainty for the months of July / August.

Table 4.2: Parameter definition, values, and units related to the coupled physical-biological-carbonate model and the experimental setup in general.

Parameters	**Values**
Physical Model	
Total depth, D_z (m)	100
Diffusion coefficient at the top, K_{z_0} (m^2/day)	43.2
Diffusion coefficient at the bottom, K_{z_b} (m^2/day)	0.1728
Thermocline sharpness, γ_t (dimensionless)	3
Biological Model	
Temperature coefficient, a $(^\circ C^{-1})$	unif(0, 1)
Maximum grazing, g_{max} (day^{-1})	0.47
Light attenuation by phytoplankton, k_P $(m^2(mmol\ N)^{-1})$	0.06
Light attenuation by detritus, k_D $(m^2(mmol\ N)^{-1})$	0.01
Light attenuation by pure water, k_W (m^{-1})	0.08
Half-saturation constant for nitrogen uptake, K_N $(mmol\ N\ m^{-3})$	0.5
Half-saturation constant for grazing, K_P $(mmol\ N\ m^{-3})$	0.25
Phytoplankton mortality, m_P $(day^{-1}(mmol\ N\ m^{-3})^{-1})$	0.08
Zooplankton mortality, m_P $(day^{-1}(mmol\ N\ m^{-3})^{-1})$	0.06
Phytoplankton sinking speed, s_P $(m\ day^{-1})$	0.3
Detritus sinking speed, s_D $(m\ day^{-1})$	3
Optimal temperature, T_{opt} $(^\circ C)$	20
Phytoplankton maximum growth rate, μ_{max} (day^{-1})	2.808
Light-photosynthesis slope, α $(m^2 day^{-1} W^{-1})$	0.14
Light-inhibition slope, β $(m^2 day^{-1} W^{-1})$	0.0028
Remineralization rate at $0^\circ C$, ϵ (day^{-1})	0.015
Active respiration, λ (dimensionless)	0.3
Zooplankton growth efficiency, γ (dimensionless)	0.4

Carbonate Model	
Carbonate precipitated per unit of primary production, γ_c (dimensionless)	0.01
Carbon:Nitrogen ratio of phytoplankton, C_P (dimensionless)	6.625
Carbon:Nitrogen ratio of zooplankton, C_P (dimensionless)	5.625
Carbon:Nitrogen ratio of detritus, C_D (dimensionless)	7.5
Others	
Carbon:Nitrogen ratio C_N (dimensionless)	6.625
Carbon:Chlorophyll-a ratio C_{Chl} (dimensionless)	40
Sea-water density ρ_w ($kg\ m^{-3}$)	1025
Number of realizations, N_{MC}	1000

4.1.4 Numerical Method

We evolve each of the realizations individually in the Monte-Carlo sense. We discretize the 1-D domain using 100 equally-spaced grid points, and use simple 2^{nd} order central difference schemes are for all the spatial derivatives with one-sided schemes only at the boundaries. The discretized system is evolved in time using a *dopri-5* [134] adaptive time-integration scheme. Numerically solving biogeochemical models is prone to states becoming negative, and often the solution exploding. In order to avoid this problem, during every evaluation of the right-hand-side, any negative state values were reset to 0.

4.1.5 Observations and Inference

For all our data assimilation needs, we will use the GMM-DO filter [16, 17] with state augmentation. The GMM-DO filter performs a Gaussian Mixture Model (GMM) based Kalman-like update in a reduced-order space, thus, rendering non-Gaussian Bayesian inference computationally feasible. We will use 30 modes and 10 GMM components. For more algorithmic details, please see appendices B and C. Further,

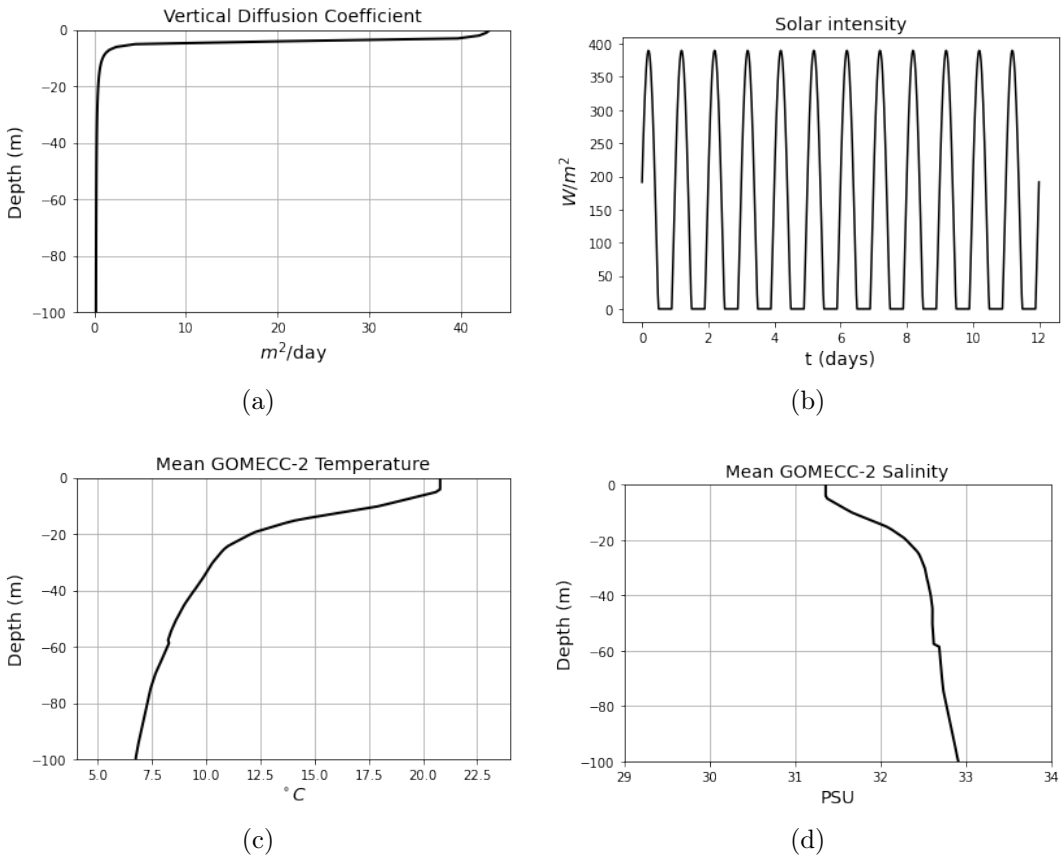


Figure 4-6: Physical features relevant to the time and area of interest, and used in the experiment. *(a):* Vertical diffusion coefficient, $K_z(z,t)$, corresponding to a stationary mixed-layer-depth of $4m$ and $\gamma_c = 3$; *(b):* Time variation of photosynthetically active radiation (PAR) at the location, $(70°W, 42.85°N)$; *(c):* Mean temperature computed from the observed GOMECC-2 data profiles; *(d):* Mean salinity computed from the observed GOMECC-2 data profiles.

all the observation are considered independent of each other, and the estimates of the sensor noise are represented through the error standard deviation matrix ($\sqrt{\boldsymbol{R}}$ in equation 2.3). Exact sensor noise values are unavailable, thus, to be on the conservative side, the standard deviation of the observation noise is chosen to be approximately 10 times the precision of the measured values for all our assimilation needs. The observed variables are first converted to the corresponding unobserved variables using the relationships in table 4.1, and then, a linear observation matrix ($\boldsymbol{H}$ in equation 2.3) is specified such that it identifies and maps the variable and the observation locations to the concatenated state space.

4.2 Experiment Overview

We first describe the various sources of uncertainties in our setup, followed by details about the model run, and finally our learning objectives.

Model uncertainty: As mentioned earlier in section 4.1.1, the change in total alkalinity (TA) can be decomposed into two components, diagnostic and prognostic. In the HadOCC based model used in the current study, the prognostic component of TA is modeled by coupling the RHS of $d(TA)/dt$ ODE (equation 4.3) with the diffusion-reaction PDE (equation 4.4). This model captures the effects of diffusion and biological processes on the variability of TA, however, lacks the ability to account for changes due to advection of water masses of different salinity caused due to precipitation, riverine input, and other oceanographic processes. Thus, to represent these unmodeled effects, we propose to add an uncertain salinity based forcing term to the existing TA equation,

$$\begin{aligned}\frac{\partial(TA)}{\partial t} =& \frac{\partial}{\partial z}\left(K_z(z, M(t))\frac{\partial(TA)}{\partial z}\right) - \frac{U_P - \lambda G_Z - \varepsilon f_T(T(z))D}{\rho_w} \\ & - \frac{2\gamma_c C_P U_P}{\rho_w} + \frac{f(S(z);\omega)}{\rho_w}\,,\end{aligned} \tag{4.10}$$

where $f(S(z);\omega)$ acts as a closure term, and is parameterized using 4 continuous

and stochastic piece-wise linear functions (section 2.2.2). ω is the realization index belonging to a measurable sample space Ω. Based on historical TA variability and salinity profiles measured during GOMECC-2, we assume that each realization of $f(S(z);\omega)$ randomly varies between ± 5 $mmol\ m^{-3}\ day^{-1}$, and the range of values taken by salinity, $S(z) \in [31, 33]$ $PSU, \forall z \in [-D_z, 0]$. Let the interval $[31, 33]$ be divided into 4 equal non-overlapping sections, such that, $31 = S_L^0 < S_R^0 = 31.25 = S_L^1 < ... < S_R^3 = 32.75 = S_L^4 < S_R^4 = 33$. Hence, $f(S(z);\omega)$ can be represented as,

$$f(S(z);\omega) = \sum_{j=0}^{5} \chi_j(\omega)\phi_j(S(z)) \tag{4.11}$$

where,

$$\phi_0(S) = \begin{cases} \frac{1}{0.25}(31.25 - S) & \text{if } 31 \leq S \leq 31.25\ , \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_i(S) = \begin{cases} \frac{1}{(S_R^{i-1} - S_L^{i-1})}(S - S_L^{i-1}) & \text{if } S_L^{i-1} \leq S \leq S_R^{i-1}\ , \\ \frac{1}{(S_R^{i} - S_L^{i})}(S_R^{i} - S) & \text{if } S_L^{i} \leq S \leq S_R^{i}\ , \\ 0 & \text{otherwise} \end{cases} \qquad \text{for } i \in \{1, ..., 4\}\ . \tag{4.12}$$

$$\phi_5(S) = \begin{cases} \frac{1}{0.25}(S - 32.75) & \text{if } 32.75 \leq S \leq 33\ , \\ 0 & \text{otherwise} \end{cases}$$

Further, each set of realizations of stochastic expansion coefficients $\chi_i's,\ i \in \{0, ..., 5\}$ are sampled in such a way, that they do not lead to a prior with highly fluctuating function realizations.

Parameter uncertainty: In the GOMECC-2 data, we notice high phytoplankton concentrations in the upper $20m$ depth (figure 4-1e). However, some preliminary deterministic model runs indicated that the parameter values taken from Tian *et al.*, 2015 [77] and Beşiktepe *et al.*, 2003 [62], and combined with the physical conditions provided in figure 4-6 led to sharp decline in the modeled phytoplankton concentra-

tions in the top $20m$. Changing the magnitude of the temperature coefficient (a) was found to have the most impact on the phytoplankton concentration in the top $20m$. Thus, the temperature coefficient (a) parameter is made uncertain, and assumed to be uniformly distributed between 0 and 1 $^{\circ}C^{-1}$.

Initial condition uncertainty: We start with the state realizations of the historical uncertainty for the months of July / August created using the relevant WOD data and explained in section 4.1.3. Next, we utilize the GTSPP buoy data in order to make the initial state uncertainty representative of the year 2012. Looking at the wind stress magnitudes at the buoy in figure 4-8(b), we can notice a major wind event on July 23. Prior to the wind event, on July 21, the winds were weaker and variable. They did not support major upwelling which agrees with the SST (figure 4-9). On July 22, the winds began to pick up in strength and were somewhat more consistently to the northeast (an upwelling favorable direction). One may see some evidence of weaker upwelling at buoy (slight decrease in SST) on July 22. However, on July 23, the winds were strong and more consistently to the northeast, and the July 23 SST showed a strong upwelling response. Further, the de-tided barotropic velocities (figure 4-7) at the buoy showed an Ekman response to the wind event turning from a southwesterly flow to a southeasterly flow in response to the winds. Note that the southeasterly de-tided barotropic flow was weak (around 3 cm/s) indicating that the winds were fighting the density driven flow. After the wind event, calmer more variable winds prevailed during the period of July 25-30, and a couple of downwelling favorable minor wind events occurred on July 27 & 29. Followed by that, the de-tided barotropic velocities at the buoy slowly turned to the southwest and strengthened through July 29, and the winds at the buoy strengthened on July 31. The winds were somewhat variable over the day with upwelling favorable winds to the northeast only occurring at the end of the day. These winds were too late to remove the warming from the downwelling on July 27 & 29 in the morning SST image on July 31. These winds may be responsible for the decrease in the de-tided barotropic velocities on July 30-31. Based on the above arguments and analysis (provided by Dr. Patrick J.

Haley Jr., *pers. comm.*), we can expect the water from the buoy on July 23 being kicked southeast towards the first two GOMECC-2 profiles and making it there on August 01. Thus, we assume that assimilating salinity observations from the GTSPP buoy (figure 4-10) on July 23 into the historical state uncertainty for the months of July / August would lead to initial state uncertainty representative of the conditions on August 01 12Z.

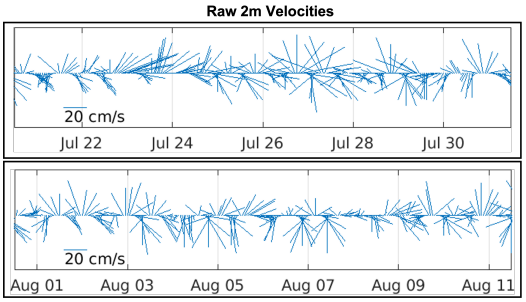
Model run: Starting with the initial condition, parameter, and model uncertainties described above, we evolve the ensemble of realizations using the coupled biogeochemical model for a period of 12 *days*, thus, ending the simulation on August 13, 2012 12Z. In order to make model simulation correspond to the period of August 01 - 13, 2012, we utilize specific physical conditions as described earlier in section 4.1.3. This allows for the states to adjust in response to the model and the physical conditions, nearly reaching an equilibrium. The evolved ensemble will act as the prior uncertainty estimate, with built-in relationships between corresponding realizations of states, parameter, and the salinity based forcing term.

Learning objectives: Our learning objectives include simultaneous estimation of states, parameter, and the functional form of salinity based forcing term, using the GOMECC-2 observations of NO_3, Chl-a, and TA.

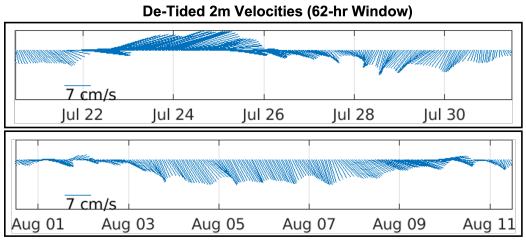
Combining all the steps described above, we provide the overview of the experiment in figure 4-11.

4.3 Application Results and Discussions

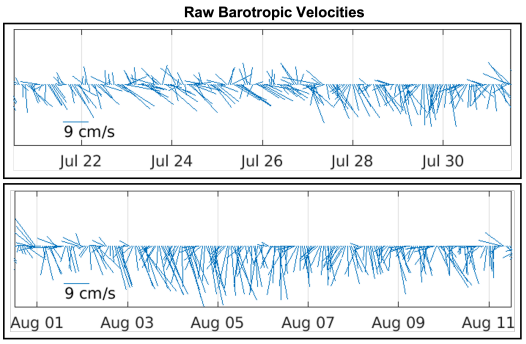
Following the experiment overview described in the section 4.2 and figure 4-11, we start with creating the initial state uncertainty representative of conditions on August 01, 2012 12Z. The GTSPP buoy salinity data measure on July 23, 2012 is first converted to TA using the empirical linear relationship given by equation 4.8. These equivalent TA observations are then assimilated into the historical uncertainty for the model states, which acts as the prior. We assume independent observation noise of



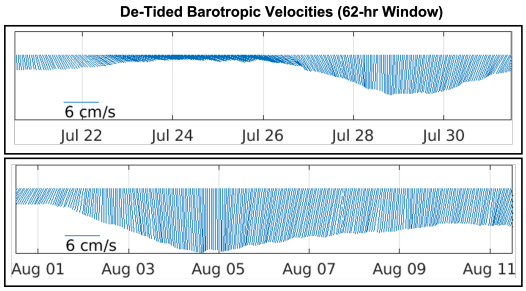
(a) Raw $2m$ velocities



(b) De-tided $2m$ velocities

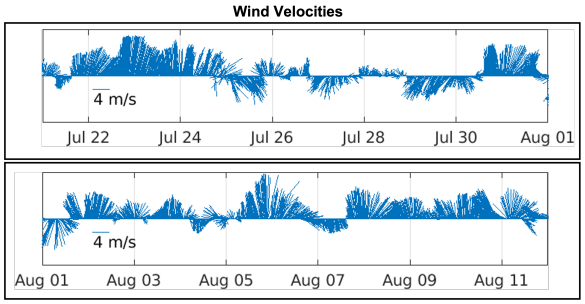


(c) Raw barotropic velocities

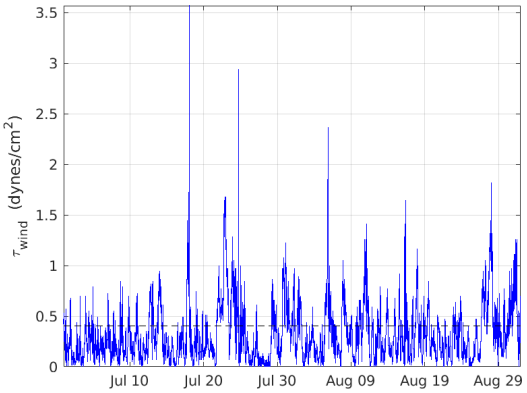


(d) De-tided barotropic velocities

Figure 4-7: Barotropic and $2m$ velocities observed at the GTSPP buoy (see figure 4-2 for location) between July 20, 2012 to August 11, 2012. We provide both, the respective raw and the 62-hour window de-tided velocities. These plots were prepared with the help of Dr. Patrick J. Haley Jr., *pers. comm.*

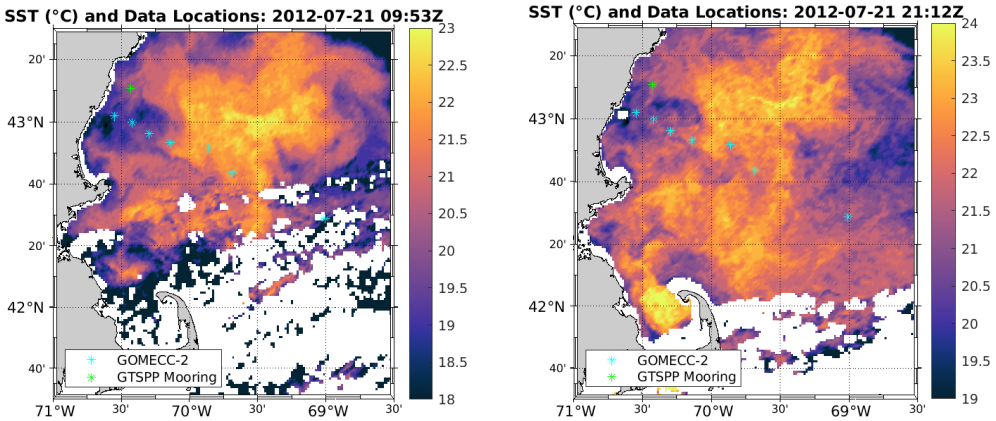


(a) Wind velocities



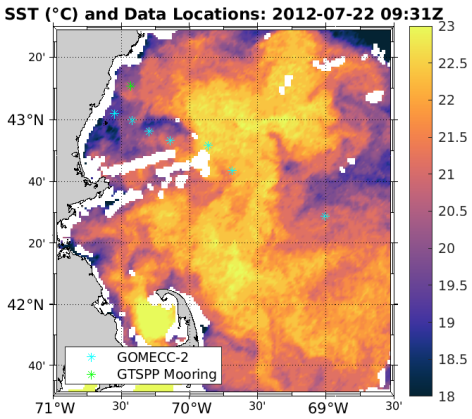
(b) Wind stress magnitudes

Figure 4-8: Wind conditions observed at the GTSPP buoy (see figure 4-2 for location). Wind velocities are provided between July 20, 2012 to August 11, 2012, while, the wind stress magnitude for the whole 2 months period. These plots were prepared with the help of Dr. Patrick J. Haley Jr., *pers. comm.*

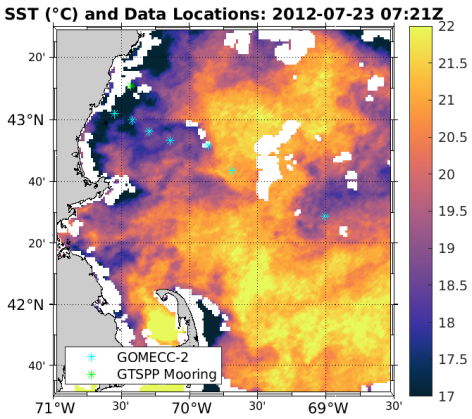


(a) July 21, 2012 09:53Z

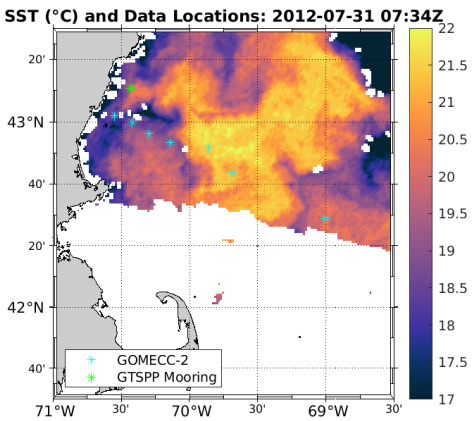
(b) July 21, 2012 21:12Z



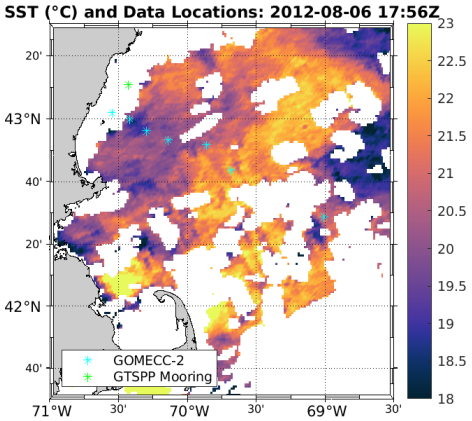
(c) July 22, 2012 09:31Z



(d) July 23, 2012 07:21Z

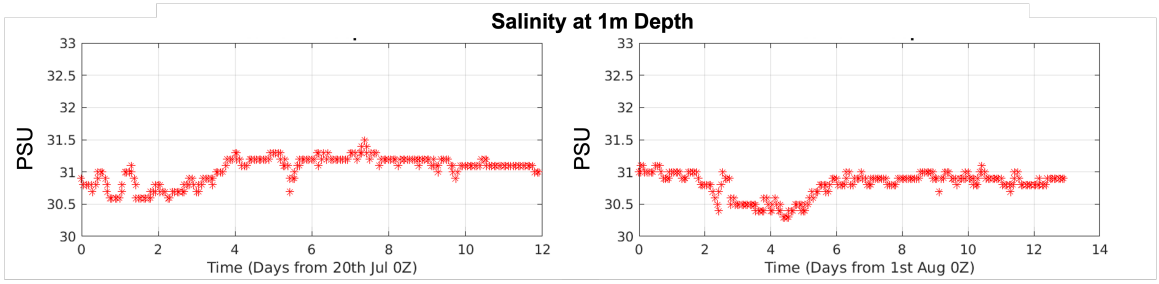


(e) July 31, 2012 07:34Z

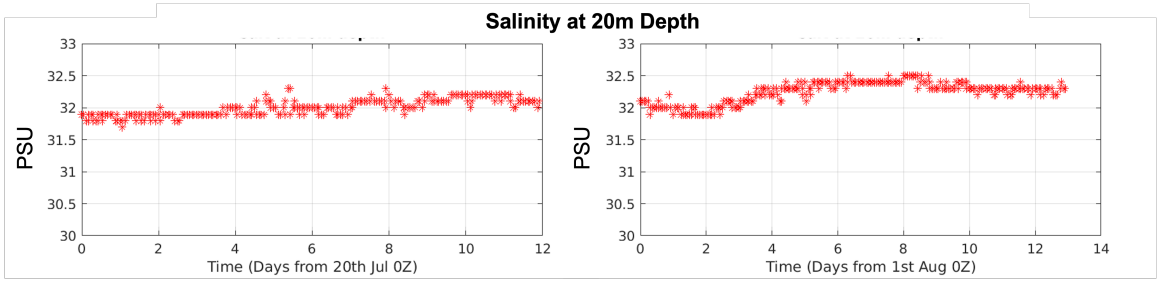


(f) August 06, 2012 17:56Z

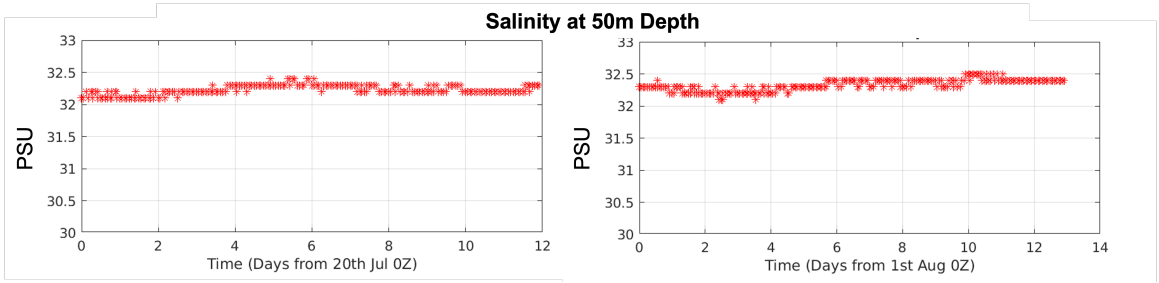
Figure 4-9: Remote sea surface temperature (SST) observed in the area and time-period of interest. Images corresponding to relatively clearer days are only provided. The *white* patches are due to cloud cover. All seven GOMECC-2 data locations, along with that of the GTSPP buoy are marked using the '*' symbol. These SST images were found and prepared with the help of Dr. Chris Mirabito, *pers. comm.*



(a) $1m$ depth



(b) $20m$ depth



(c) $50m$ depth

Figure 4-10: Hourly salinity values observed at the GTSPP buoy (see figure 4-2 for location) at 3 different depths between July 20, 2012 to August 13, 2012.

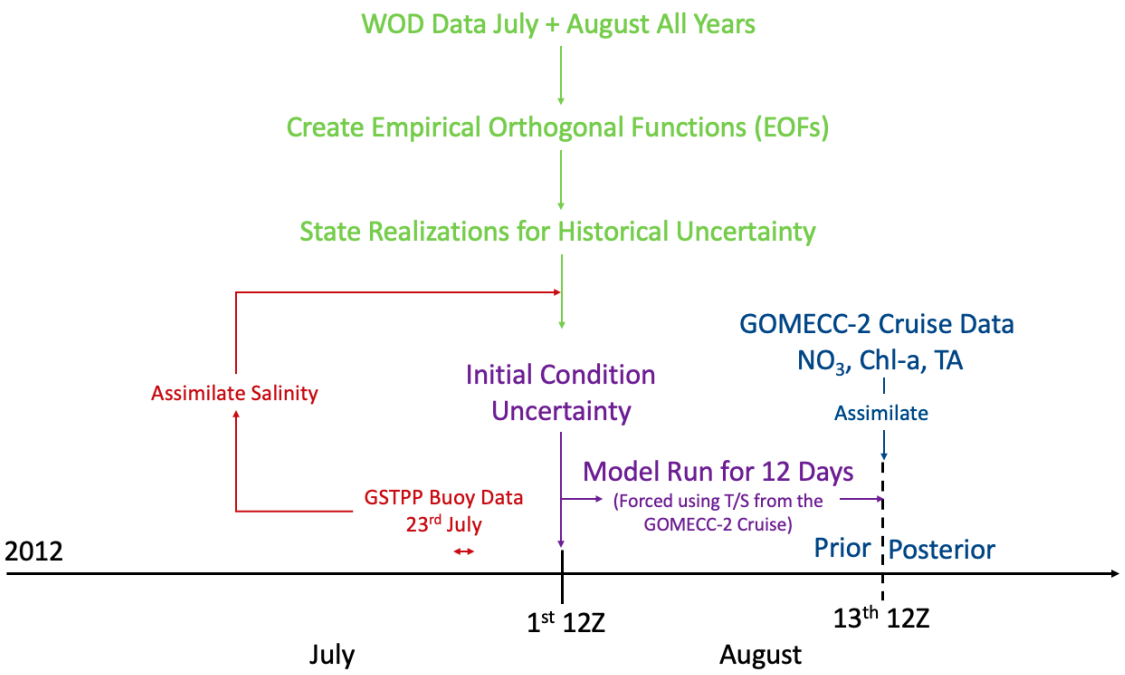
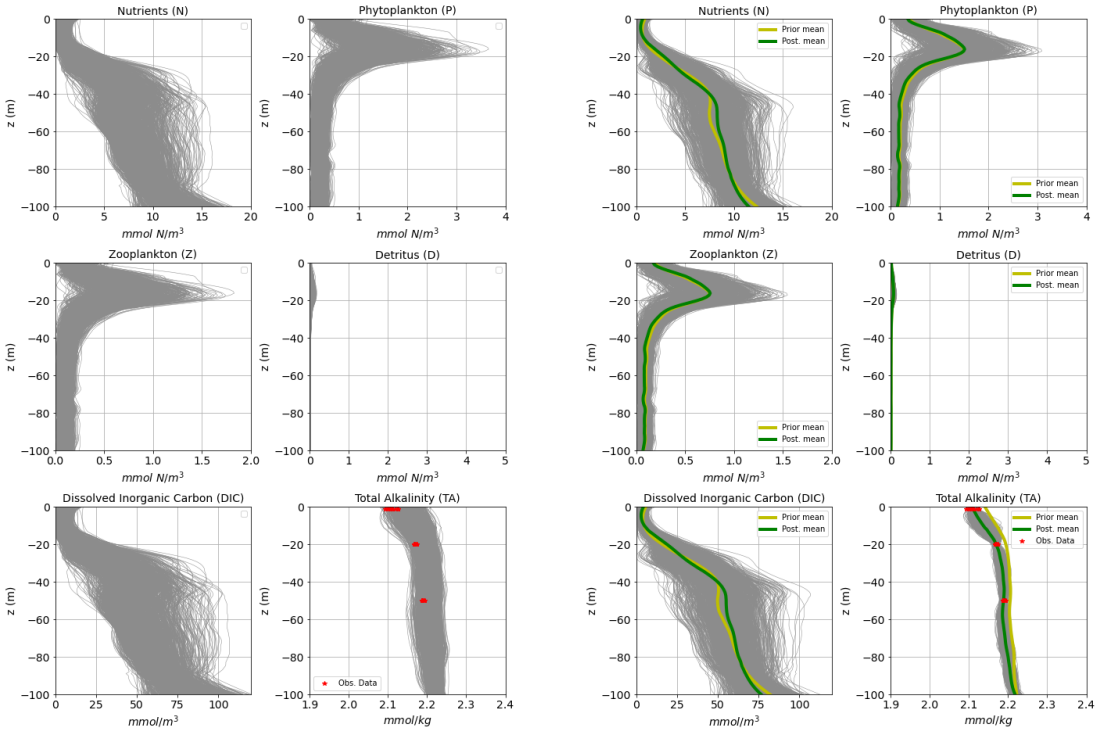


Figure 4-11: Diagram depicting the overview of the experiment. See figure 4-2 for abbreviations.

standard deviation, 0.02 $mmol/kg^{-1}$, and utilize the GMM-DO filter (section 4.1.5). The prior and the posterior state ensembles are provided in figure 4-12. We notice that the major uncertainty reduction is only obtained for TA, indicating weak coupling between historical observations of biological states and salinity.

Next, we evolve each member of the initial state ensemble (figure 4-13(a)) on August 01, 2012 12Z for 12 *days*. During the evolution, each state realization experiences a randomly selected member from the $f(S(z);\omega)$ and the temperature coefficient $a(\omega)$ ensemble. As mentioned earlier, $f(S(z),\omega)$ is parameterized using 4 continuous piece-wise linear functions nearly encompassing all possible functions with the range, $[-5,5]$ $mmol\ m^{-3}\ day^{-1}$, and domain, $[31,33]$ PSU. While, $a(\omega)$ is independently sampled from a uniform distribution between 0 and 1 $^{\circ}C^{-1}$. In figure 4-13(a), we can notice a clear discrepancy between initial TA realizations and the corresponding GOMECC-2 data. A positive $f(S(z),\omega)$ value leads to an increase in TA concentration, while a negative value decreases it. Thus, after 12 *days* of simulation, the TA ensemble members spread out, abridging the discrepancy and encapsulating the ob-

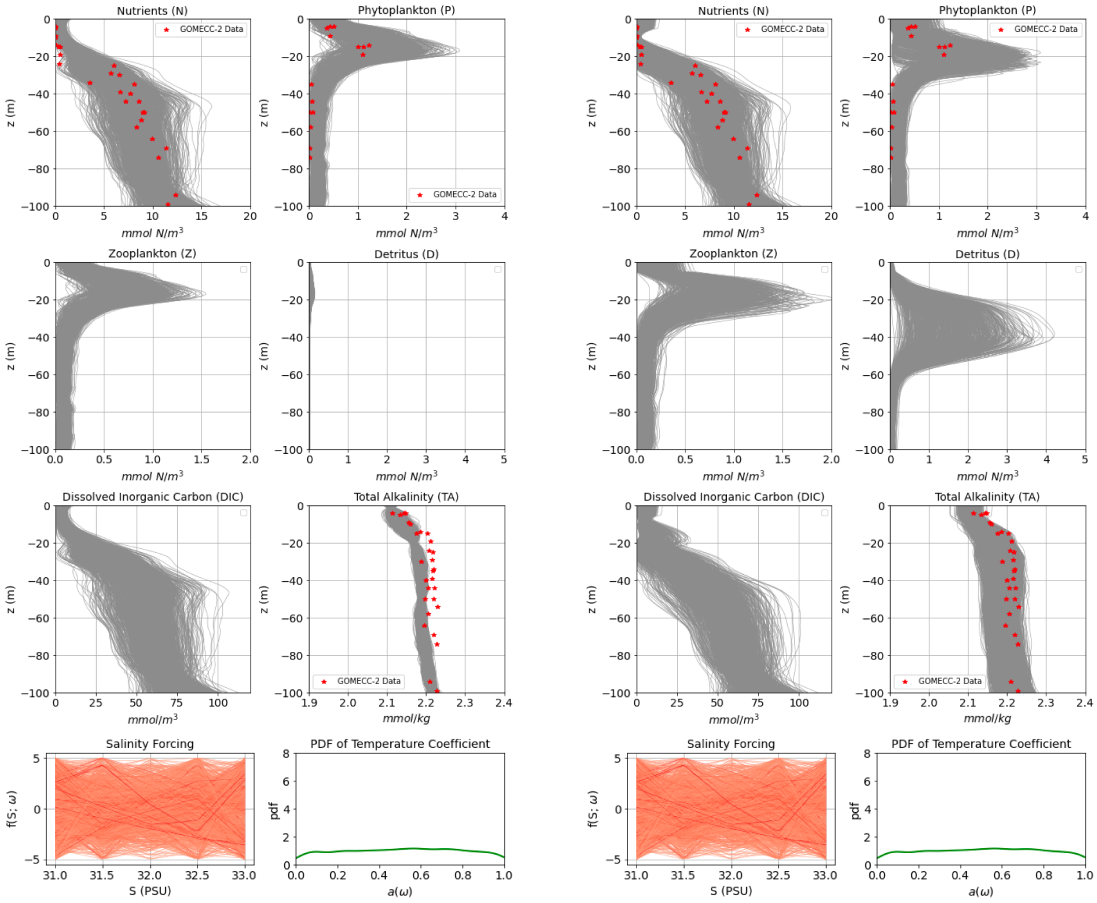


(a) Prior (b) Posterior

Figure 4-12: State ensembles (uncertainty estimates) before and after assimilating the GTSPP buoy salinity data observed on July 23, 2012. *(a):* July / August historical state uncertainty obtained from WOD data, and which acts as the prior; *(b):* Posterior obtained after assimilating the observed salinity data converted to total alkalinity (TA) using empirical relationship provided in equation 4.8 (marked with red '$\star$' symbol). Each state ensemble is overlayed with their corresponding prior and posterior means.

served TA data within their variability (figure 4-13(b)). Similarly, we also see increase in uncertainty of other modeled variables due to each realization reaching a different near equilibrium condition or a point on the limit cycle, and also getting affected by a different temperature coefficient value ($a(\omega)$). The final state uncertainty on August 13, 2012 12Z along with that for $f(S(z);\omega)$ and $a(\omega)$, is provided in figure 4-13(b). It should be noted that $f(S(z);\omega)$ and $a(\omega)$ are autonomous, thus, do not change in time.

The evolved state realizations along with that for $f(S(z);\omega)$ and $a(\omega)$ acts as the prior model uncertainty estimate for August 13, 2012 12Z in the region around the GOMECC-2 data locations. We will assimilate the GOMECC-2 observations and perform simultaneous estimation of uncertain states, model, and the parameter. Because of the lack of availability of additional data for the purpose of validation, we perform a validation study by holding out data corresponding one of the observed variables during assimilation, and repeating it thrice. The Bayesian inference step is performed by augmenting $\chi_i(\omega)$'s (equation 4.11), $a(\omega)$ and the all the discretized state variables, and using the GMM-DO filter. We convert the observed data for NO_3 and Chl-a to N and P, respectively, using the relationships provided in table 4.1. We assume independent observation noise of standard deviation, 0.5 $mmol\ N/m^{-3}$ for N, 0.1 $mmol\ N/m^{-3}$ for P, and 0.01 $mmol/kg^{-1}$ for TA. In figures 4-15, 4-16, and 4-14, we provide the posterior obtained after assimilating data in pairs of $\{N,\ TA\}$, $\{P,\ TA\}$, and $\{N,\ P\}$, respectively. In both the cases of $\{N,\ TA\}$, and $\{P,\ TA\}$ data pairs, especially for the variables whose data are being assimilated, we notice large uncertainty reductions and agreement with the data in the posterior. For their respective validation variables, we notice uncertainty reduction and the observed data still contained within the variability of the posterior realizations. However, for the $\{N,\ TA\}$ data pair case, we notice a peculiar behavior of the P posterior realizations. They show a sharp fluctuation between $10-15\ m$ depth, which also correspond to the posterior pdf of $a(\omega)$ getting concentrated around lower values (figure 4-15(b)). While for the $\{P,\ TA\}$ data pair, because P is directly being observed, the Bayesian inference picks the easiest explanation for the observed data, and eliminates the fluctuation



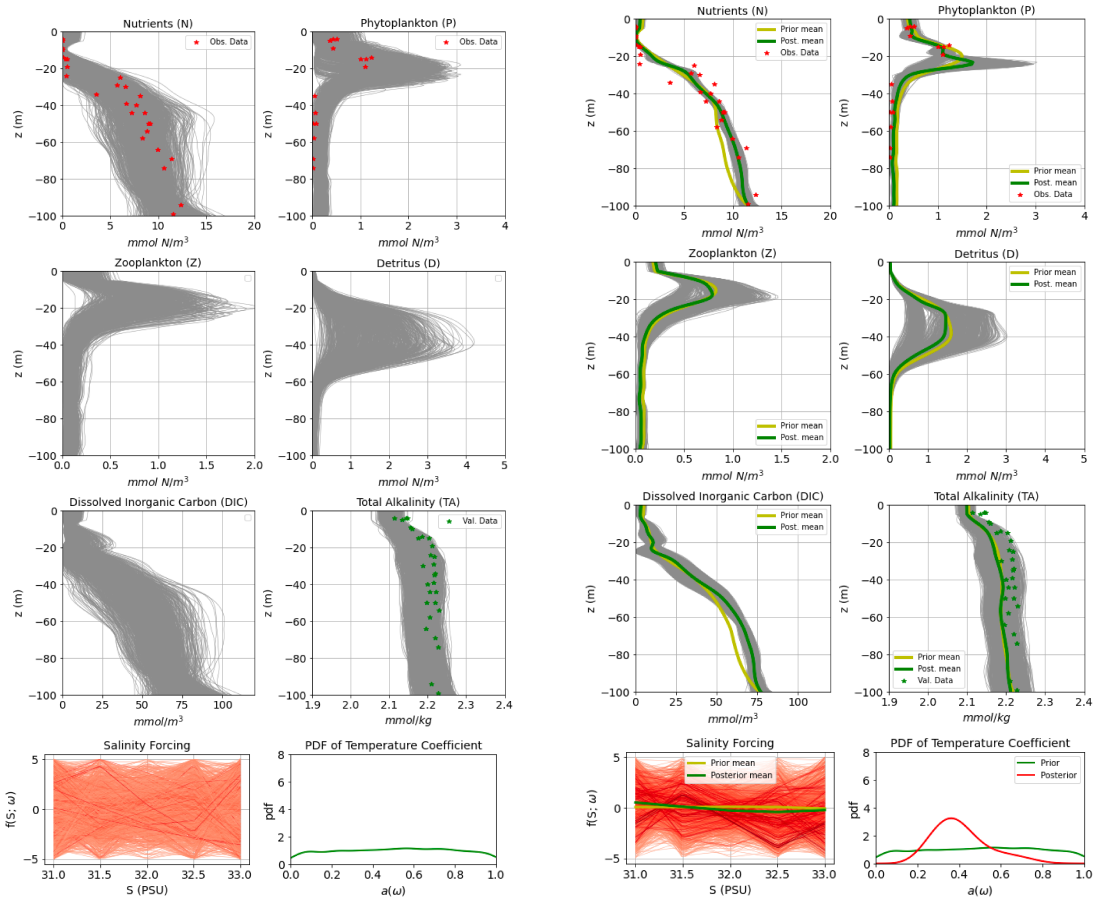
(a) August 01, 2012 12Z

(b) August 13, 2012 12Z

Figure 4-13: Evolved state ensemble members (uncertainty estimates) after 12 *days* of model run. *(a):* Initial state uncertainty (same as the posterior obtained after assimilating GTSPP data in figure 4-12(b)), realizations for the salinity forcing term ($f(S(z);\omega)$; colored according to their respective normalized probability density values (red for 1 and white for 0); *bottom-left*), and probability distribution for the temperature coefficient ($a(\omega)$; *bottom-right*); *(b):* State uncertainty at the end of model run. Realizations for the salinity forcing term ($f(S(z);\omega)$; *bottom-left*) and probability distribution for the temperature coefficient ($a(\omega)$; *bottom-right*) are exactly the same as that in *(a)*. GOMECC-2 nitrate, chlorophyll-a (converted to N and P, respectively, using the relationships provided in table 4.1), and total alkalinity data is also provided and marked with red '$\star$' symbol.

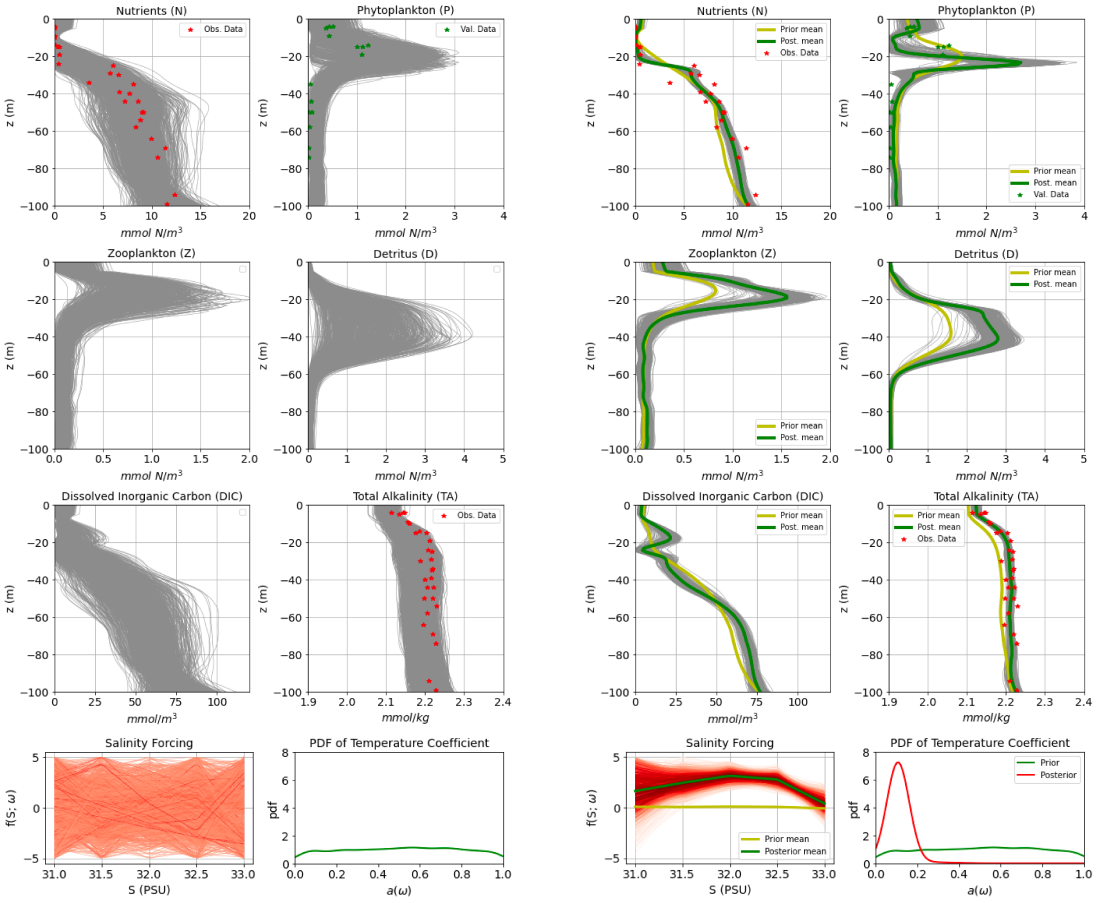
in the posterior. This also leads to posterior pdf of $a(\omega)$ concentrating around relatively higher values (figure 4-16(b)). In both the cases, we learn a positive $f(S(z);\omega)$ function with high amount of certainty for the three piece-wise segments between $31.25 \leq S \leq 33$. The low salinity values correspond to the upper depths (figure 4-6), where other processes such as high diffusion (mixing) in the mixed-layer dominate the TA equation and are approximately 10 times larger in magnitude than the salinity forcing term. This makes it harder to learn the $f(S(z);\omega)$ function with confidence. Due to little disagreement between our estimated initial state uncertainty of TA on August 01, 2012 12Z and the GOMECC-2 TA data at the surface and the bottom (figure 4-13(a)), and also the high salinity values corresponding to the bottom, we notice a near zero posterior mean for the $f(S(z);\omega)$ function. The learned positive $f(S(z);\omega)$ function corresponds to the need for increasing the TA concentration in order to fill the gap between TA initial condition and the data through the model evolution (figure 4-13). In the case of the $\{N,\ P\}$ data pair, we barely notice any reduction in uncertainty for TA, thus, indicating a weak influence of the biological states on the evolution of TA. Such weak coupling was also noticed earlier between the historical WOD data of NO_3 and Chl-a, with S (in general, S is highly correlated with TA). This also leads to the inability to learn the $f(S(z);\omega)$ function only using biological state data. However, because the $a(\omega)$ parameter directly influences biology, we notice a tightening of the posterior pdf. The posterior pdf concentrates around a mid value as compared to that obtained in the case of $\{N,\ TA\}$ and $\{P,\ TA\}$ data pairs. Overall, the validation study demonstrated the ability to make reasonable estimates for the hold-out data, thus, indicating our modeling methodology predicts representative estimates of uncertainty for the GOMECC-2 data.

Finally, we assimilate all the GOMECC-2 observations corresponding to N, P, and TA simultaneously. The posterior presented in figure 4-17 demonstrates a combination of features we pointed out in the above validation study. We notice reduction in uncertainty for all the biological and carbonate variables along with agreement with the observed data. The learned $f(S(z);\omega)$ is mostly positive, and the pdf of the $a(\omega)$ parameter is concentrated in the middle. Overall, we achieve our learning objective



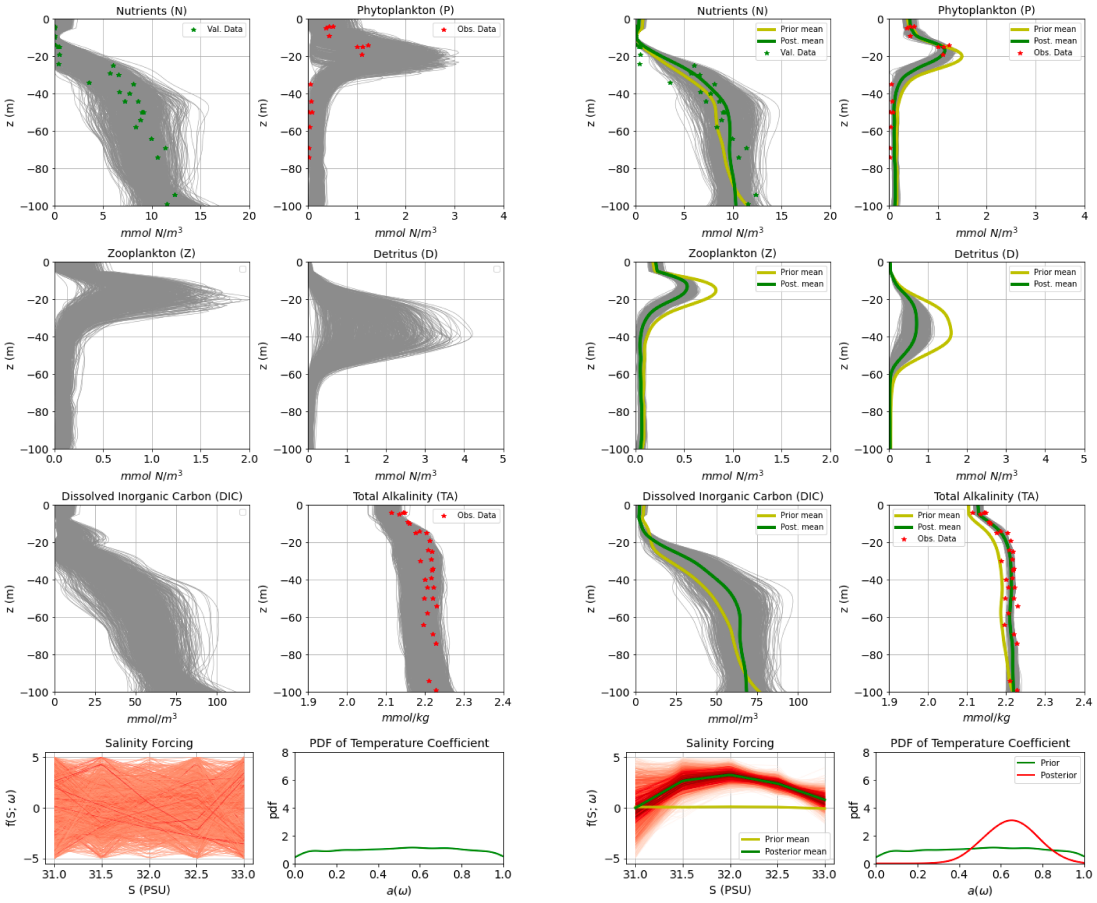
(a) Prior (b) Posterior

Figure 4-14: Uncertainty before and after assimilating the GOMECC-2 nitrate and chlorophyll-a data (converted to N and P, respectively, using the relationships provided in table 4.1). *(a):* State uncertainty at the end of model run, realizations for the salinity forcing term ($f(S(z);\omega)$; colored according to their respective normalized probability density values (red for 1 and white for 0); *bottom-left*) and probability distribution for the temperature coefficient ($a(\omega)$; *bottom-right*). These are exactly the same as those in figure 4-13(b) and acts as the prior; *(b):* Posterior obtained after assimilating the GOMECC-2 data (marked with red '$\star$' symbol). The GOMECC-2 total alkalinity (TA) data (marked with green '$\star$' symbol) is utilized for validation purposes. State and $f(S(z);\omega)$ ensembles are overlayed with their corresponding prior and posterior means. We also provide both prior and posterior PDF for $a(\omega)$ parameter for easy comparison (*bottom-right*).



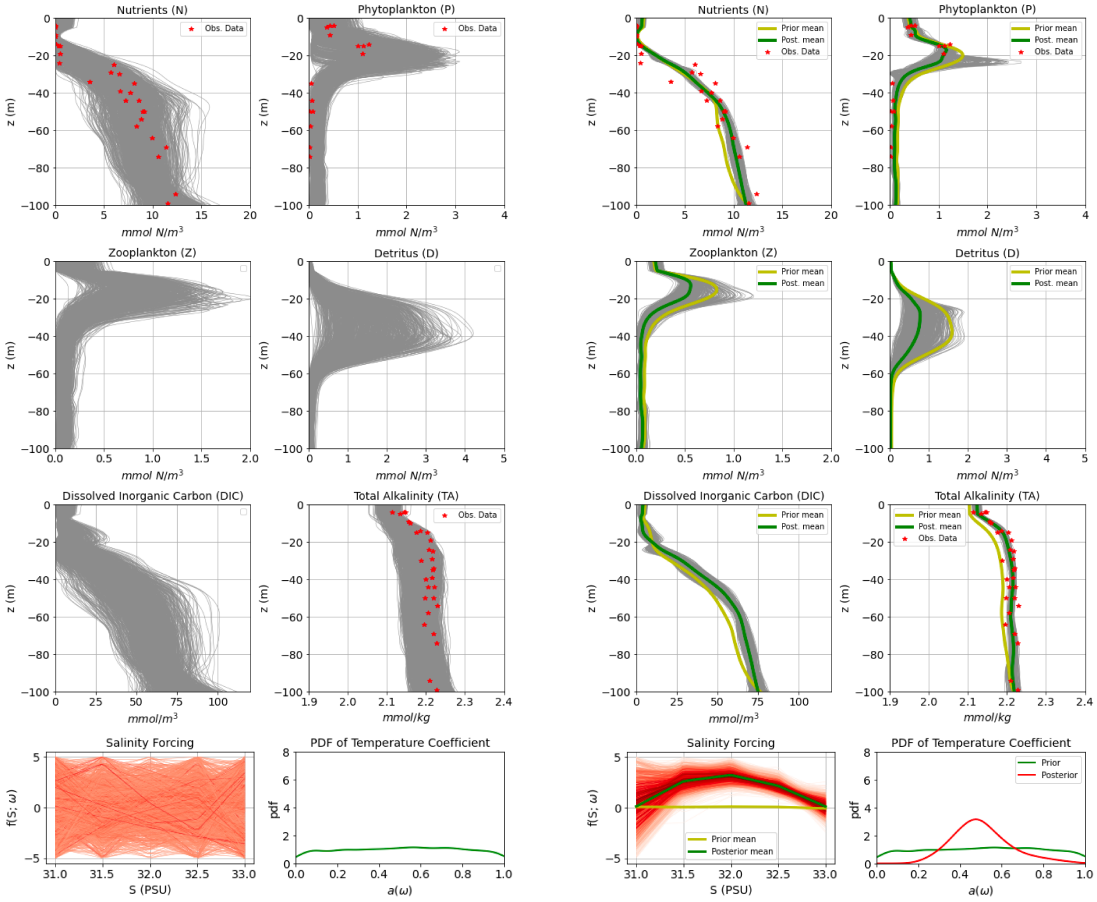
(a) Prior (b) Posterior

Figure 4-15: Uncertainty before and after assimilating the GOMECC-2 nitrate (converted to N using the relationship provided in table 4.1) and total alkalinity data. The GOMECC-2 chlorophyll-a data (converted to P using the relationship provided in table 4.1, marked with green '$\star$' symbol) is utilized for validation purposes. The rest of the description is the same as in figure 4-14.



(a) Prior (b) Posterior

Figure 4-16: Uncertainty before and after assimilating the GOMECC-2 chlorophyll-a (converted to P using the relationship provided in table 4.1) and total alkalinity data. The GOMECC-2 nitrate data (converted to N using the relationship provided in table 4.1, marked with green '$\star$' symbol) is utilized for validation purposes. The rest of the description is the same as in figure 4-14.



(a) Prior (b) Posterior

Figure 4-17: Uncertainty before and after assimilating the GOMECC-2 nitrate, chlorophyll-a (converted to N and P, respectively, using the relationships provided in table 4.1), and total alkalinity data. The rest of the description is the same as in figure 4-14.

of simultaneous estimation of states, parameter, and the functional form of salinity based forcing term in an existing biogeochemical model for the GoM.

4.4 Summary

In the present study, we demonstrated the application of our novel PDE-based Bayesian model learning framework for discovery of missing processes and interactions in existing ocean acidification (OA) models using real-world data. OA related *in-situ* measurements were made during the Gulf of Mexico and East Coast Carbon (GOMECC-

2) Cruise on its last day, August 13, 2012, in the Gulf of Maine (GoM) region. We configured a one-dimensional (1-D) coupled physical-biological-carbonate model optimized for the GoM region and utilized model-data misfits to simultaneously estimate states, an uncertain parameter, and discover functional form of a missing process with quantifiable uncertainty. The 1-D biogeochemical model consisted of a diffusion-reaction PDE for the physics part, a four-component nutrients (N), phytoplankton (P), zooplankton (Z), and detritus (D) model for the biological part, and a two-component dissolved inorganic carbon (DIC) and total alkalinity (TA) model for the carbonate part based on the Hadley Centre Ocean Carbon Cycle (HadOCC) model [123]. The lack of ability to account for changes in TA due to advection of water masses of different salinity arising from precipitation, riverine input, and other oceanographic processes was compensated by adding an uncertain salinity based forcing term to the existing TA equation. The salinity based forcing term was parameterized using 4 continuous and stochastic piece-wise linear functions, and discovered using the observed GOMECC-2 data. Other sources of uncertainty include initial conditions, and a coefficient in the term parameterizing temperature effect on phytoplankton growth and other biological rates. The initial condition uncertainty was estimated based on the historical *in-situ* observations made for the months of July / August for the variables of salinity, nitrate, and chlorophyll-a, and assimilating data from a nearby buoy operating during the time of interest. A model run quantifying these different sources of uncertainties provided the prior estimate of the state conditions around the GOMECC-2 observation locations on August 13, 2012. Observed GOMECC-2 data for nitrate, chlorophyll-a, and TA are assimilated for simultaneous estimation of all the multidisciplinary states, temperature coefficient parameter, and the salinity based forcing term. The learned salinity based forcing term was found to be positive, and corresponded to the need for increasing the TA concentration, in order to fill the gap between the TA initial condition estimate and the observed data through the model evolution. The states and the temperature coefficient experienced reduction in uncertainty in accordance with the observed data, and was validated by a data hold-out validation study.

Overall, our novel Bayesian model learning framework is effective in real-world data applications, and could be utilized to learn and discover parameterizations of missing processes and interactions with quantifiable uncertainty estimates in different scientific problems.

Chapter 5

Improving Uncertainty Quantification and Observation Planning

Obtaining an accurate and informative prior is an important first step in a Bayesian learning methodology. However, uncertainty evolution and quantification in high-dimensional systems, like ocean flows, are in general computationally prohibitive. To mitigate this issue in our Bayesian learning framework, we use the dynamically orthogonal (DO) equations [11, 12, 13, 14, 15], which propagates the uncertainty in a reduced subspace, thus rendering it computationally feasible. The computational speed-up occurs due to the truncation of relatively less important directions (modes) of uncertainty propagation. However, this comes at the expense of accuracy. It then becomes necessary to limit any sources of additional inaccuracies, such as incompatible numerical schemes. In this chapter, we will develop theory and apply existing techniques to properly handle stochastic boundary conditions, complex geometries, advection term, and augment the DO subspace as and when required to capture the effects of the truncated modes. Further, we will also discuss mutual information-based observation planning to answer *what*, *when*, and *where* to observe to best achieve our learning objectives in resource-constrained environments.

5.1 Stochastic Boundary Conditions[1]

Boundary conditions determine the uniqueness of the solution of a PDE system, thus, strongly affecting the dynamics of the solution. Unfortunately, boundary conditions (BCs) are also a major source of uncertainty, and it is imperative to account for them during uncertainty evolution. However, it is not straightforward to solve for a low-rank solution which also sufficiently satisfies the uncertain BCs. The existing approaches can be broadly classified into two categories, weak and strong imposition methods. In the weak imposition of stochastic BCs, it is ensured that the solution in the interior satisfies the BCs only up to 2^{nd} moment, and has been derived for the DO equations [11, 14, 9]. On the other hand, the strong imposition method ensures that each realization of the approximate solution in the interior satisfies the same boundary conditions as the exact solution, or a well controlled approximation of them, and has only been derived for the Dual DO equations [135]. Dual DO equations are a variant of the original DO equations, where the condition for dynamic orthogonality is imposed on the stochastic coefficients, instead of the modes, intending to better treat stochastic BCs. Section 4.5.5 in Lin, 2020 [9] provides a nice juxtaposition between the two. In this section, we will first visit the derivation of the weak imposition of BCs for the DO equations, followed by the derivation of the corresponding strong imposition of BCs, for a general stochastic dynamical system of the form,

$$\begin{aligned}
\frac{\partial \boldsymbol{u}(\boldsymbol{x},t;\omega)}{\partial t} &= \mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega], \qquad \boldsymbol{x}\in D,\ t\in\mathcal{T} \text{ and } \omega\in\Omega \\
\boldsymbol{u}(\boldsymbol{x},0;\omega) &= \boldsymbol{u}_0(\boldsymbol{x};\omega), \qquad \boldsymbol{x}\in D \text{ and } \omega\in\Omega \\
\mathcal{B}[\boldsymbol{u}]|_{\partial D} &= \boldsymbol{b}(\boldsymbol{x},t;\omega), \qquad \boldsymbol{x}\in\partial D,\ t\in\mathcal{T} \text{ and } \omega\in\Omega
\end{aligned} \tag{5.1}$$

where $\mathcal{B}$ is a boundary operator, such as a Dirichlet or Neumann operator. In what follows, we assume it is a linear operator for simplicity.

[1]Done with inputs from Dr. Jing Lin

5.1.1 Weak Imposition of BCs

Starting with the DO decomposition of the state variable denoted by,

$$\boldsymbol{u}(\boldsymbol{x},t;\omega) = \bar{\boldsymbol{u}}(\boldsymbol{x},t) + \sum_{i=1}^{S} Y_i(t;\omega)\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t), \qquad \boldsymbol{x} \in D,\ t \in \mathcal{T} \text{ and } \omega \in \Omega \tag{5.2}$$

we can derive the mean, modes, and coefficients evolution equations as done in appendix A,

$$\frac{\partial \bar{\boldsymbol{u}}(\boldsymbol{x},t)}{\partial t} = \mathbb{E}[\mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t;\omega),\boldsymbol{x},t;\omega]], \qquad \boldsymbol{x} \in D, t \in \mathcal{T} \tag{5.3}$$

$$\frac{\partial \tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)}{\partial t} = \sum_{j=1}^{S} C^{-1}_{Y_iY_j}\Pi^{\perp}_{\boldsymbol{u}}[\mathbb{E}[Y_j(t;\omega)\mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t;\omega),\boldsymbol{x},t;\omega]]] \qquad \boldsymbol{x} \in D, t \in \mathcal{T}, \forall i \in \{1,...,S\}, \tag{5.4}$$

$$\frac{\mathrm{d}Y_i(t;\omega)}{\mathrm{d}t} = \langle \mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t;\omega),\boldsymbol{x},t;\omega] - \mathbb{E}[\mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t;\omega),\boldsymbol{x},t;\omega]], \tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)\rangle \qquad t \in \mathcal{T}, \forall i \in \{1,...,S\}\,. \tag{5.5}$$

The boundary conditions for the mean and modes equations will be derived using a second order moment matching. We start by plugging the DO decomposition (equation 5.2) into the boundary conditions (equation 5.1),

$$\mathcal{B}[\bar{\boldsymbol{u}}]|_{\partial D} + \sum_{i=1}^{S} Y_i\mathcal{B}[\tilde{\boldsymbol{u}}_i]|_{\partial D} = \bar{\boldsymbol{b}}(\boldsymbol{x},t) + (\boldsymbol{b}(\boldsymbol{x},t;\omega) - \bar{\boldsymbol{b}}(\boldsymbol{x},t)), \qquad \boldsymbol{x} \in \partial D\,. \tag{5.6}$$

Equating the mean and the stochastic part, we obtain,

$$\mathcal{B}[\bar{\boldsymbol{u}}]|_{\partial D} = \bar{\boldsymbol{b}}(\boldsymbol{x},t), \qquad \boldsymbol{x} \in \partial D\,, \tag{5.7}$$

and,

$$Y_i\mathcal{B}[\tilde{\boldsymbol{u}}_i]|_{\partial D} = (\boldsymbol{b}(\boldsymbol{x},t;\omega) - \bar{\boldsymbol{b}}(\boldsymbol{x},t)), \qquad \boldsymbol{x} \in \partial D\,. \tag{5.8}$$

Projecting equation 5.8 onto the space spanned by the coefficients,

$$\begin{aligned}
&\mathbb{E}[Y_j Y_i]\mathcal{B}[\tilde{\boldsymbol{u}}_{\boldsymbol{i}}]|_{\partial D} = \mathbb{E}[Y_j(\boldsymbol{b}(\boldsymbol{x},t;\omega) - \bar{\boldsymbol{b}}(\boldsymbol{x},t))], \qquad \boldsymbol{x} \in \partial D \\
&\implies \mathcal{B}[\tilde{\boldsymbol{u}}_{\boldsymbol{i}}]|_{\partial D} = C^{-1}_{Y_i Y_j}\mathbb{E}[Y_j(\boldsymbol{b}(\boldsymbol{x},t;\omega) - \bar{\boldsymbol{b}}(\boldsymbol{x},t))], \qquad \boldsymbol{x} \in \partial D \ , \forall i \in \{1,...,S\} \ .
\end{aligned} \tag{5.9}$$

Equations 5.7 and 5.9 provide the BCs for the mean and the modes equations, respectively. A similar approach for handling stochastic BCs is also mentioned in Cheng *et al.* 2013 [136] and Maître *et al.* 2002 [137]. However, in this method, the orthonormality of the modes is only maintained in the interior, and there is no guarantee of a good alignment of the stochastic subspace of the BCs and the DO coefficients space, which determines the accuracy of this approach.

5.1.2 Strong Imposition of BCs

The strong imposition of BCs requires a realization-wise match between a controlled approximation of the interior and the boundary realizations. In order to achieve this, we start by splitting the DO modes into two categories, unforced and forced. Let R be the number of unforced modes ($\boldsymbol{u}_i$'s) with zero boundary conditions, and M be the number of forced modes ($\boldsymbol{v}_i$'s) with non-zero boundary conditions. To enforce the connection between the interior and the boundaries, the overall S-rank ($S = R + M$) approximation is,

$$\boldsymbol{u}(\boldsymbol{x},t;\omega) = \bar{\boldsymbol{u}}(\boldsymbol{x},t) + \sum_{i=1}^{R} \tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)Y_i(t;\omega) + \sum_{i=1}^{M} \tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)Z_i(t;\omega)\,. \tag{5.10}$$

Plugging this modified DO decomposition into the stochastic BCs (equation 5.1), and assuming a linear boundary operator $\mathcal{B}$, we obtain,

$$\begin{aligned}
\mathcal{B}[\bar{\boldsymbol{u}}(\boldsymbol{x},t)]|_{\partial D} + \sum_{i=1}^{R} \mathcal{B}[\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)]|_{\partial D} Y_i(t;\omega) + \sum_{i=1}^{M} \mathcal{B}[\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)]|_{\partial D} Z_i(t;\omega) \\
= \boldsymbol{b}(\boldsymbol{x},t;\omega), \quad \boldsymbol{x} \in \partial D
\end{aligned} \tag{5.11}$$

thus, the mean field is subject to,

$$\mathcal{B}[\bar{\boldsymbol{u}}(\boldsymbol{x},t)]|_{\partial D} = \bar{\boldsymbol{b}}(\boldsymbol{x},t), \qquad \boldsymbol{x} \in \partial D\,, \tag{5.12}$$

and, as we assumed $\mathcal{B}[\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)]|_{\partial D} = 0,\ \boldsymbol{x} \in \partial D$, we get,

$$\sum_{i=1}^{M} \mathcal{B}[\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)]|_{\partial D} Z_i(t;\omega) = \boldsymbol{b}(\boldsymbol{x},t;\omega) - \bar{\boldsymbol{b}}(\boldsymbol{x},t)\,. \tag{5.13}$$

We further assume a KL decomposition for the stochastic component of the boundary condition in the basis of $Z_i(t;\omega),\ \forall i \in \{1,...,M\}$, and in order to get an unique decomposition, impose $\mathbb{E}[Z_i(t;\omega)Z_j(t\ \omega)] = \delta_{ij}$. Thus, we get,

$$\mathcal{B}[\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)]|_{\partial D} = \mathbb{E}[(\boldsymbol{b}(\boldsymbol{x},t;\omega) - \bar{\boldsymbol{b}}(\boldsymbol{x},t))Z_i(t;\omega)], \qquad \boldsymbol{x} \in \partial D,\ \forall i \in \{1,...,M\}\ . \tag{5.14}$$

It should be noted that $Z_i(t;\omega)$ are pre-known and need not to be solved for. Substituting the modified DO decomposition (equation 5.10) into the generic stochastic PDE (equation 5.1), we get,

$$\begin{aligned}\frac{\partial \bar{\boldsymbol{u}}(\boldsymbol{x},t)}{\partial t} + \sum_{i=1}^{R}\left[\frac{\partial \tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)}{\partial t}Y_i(t;\omega) + \tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)\frac{dY_i(t;\omega)}{dt}\right] \\ + \sum_{i=1}^{M}\left[\frac{\partial \tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)}{\partial t}Z_i(t;\omega) + \tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)\frac{dZ_i(t;\omega)}{dt}\right] = \mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega]\,.\end{aligned} \tag{5.15}$$

Applying the expectation operator, we get the mean equation,

$$\frac{\partial \bar{\boldsymbol{u}}(\boldsymbol{x},t)}{\partial t} = \mathbb{E}[\mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t;\omega),\boldsymbol{x},t;\omega]]\,, \tag{5.16}$$

and substituting it back, we obtain,

$$\begin{aligned}
&\sum_{i=1}^{R}\left[\frac{\partial\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)}{\partial t}Y_i(t;\omega)+\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)\frac{dY_i(t;\omega)}{dt}\right]+\sum_{i=1}^{M}\left[\frac{\partial\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)}{\partial t}Z_i(t;\omega)+\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)\frac{dZ_i(t;\omega)}{dt}\right]\\
&=\mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega]-\mathbb{E}[\mathcal{L}[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t;\omega),\boldsymbol{x},t;\omega]]\\
&\equiv\mathcal{L}^*[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega]\,.
\end{aligned}\tag{5.17}$$

In order to close the system, we need to impose some additional constraints. We impose the original DO conditions on the unforced modes,

$$\left\langle\frac{\partial\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)}{\partial t},\tilde{\boldsymbol{u}}_j(\boldsymbol{x},t)\right\rangle=0\;,\quad\forall i\neq j,\;i,j\in\{1,...,R\}\;,\tag{5.18}$$

and also impose the condition that the stochastic coefficients corresponding to the forced and unforced modes are orthogonal to each other,

$$\mathbb{E}[Y_i(t;\omega)Z_j(t;\omega)]=0\;,\quad\forall i\in\{1,...,R\},\;j\in\{1,...,M\}\;.\tag{5.19}$$

It should be noted that no constraints were imposed on the forced modes. We can derive the evolution equations for $\tilde{\boldsymbol{u}}_i$'s by simply projecting equation 5.17 onto $Z_j(t;\omega)$,

$$\frac{\partial\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)}{\partial t}=\mathbb{E}[\mathcal{L}^*[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega]Z_i(t;\omega)],\quad\forall i\in\{1,...,M\}\;.\tag{5.20}$$

Substituting this back in equation 5.17,

$$\begin{aligned}
\sum_{i=1}^{R}\left[\frac{\partial\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)}{\partial t}Y_i(t;\omega)+\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)\frac{dY_i(t;\omega)}{dt}\right]=&\Pi_Z^{\perp}\mathcal{L}^*[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega]\\
&-\sum_{i=1}^{M}\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)\frac{dZ_i(t;\omega)}{dt}\;,
\end{aligned}\tag{5.21}$$

and taking inner product with $\tilde{\boldsymbol{u}}_j(\boldsymbol{x},t)$, we get,

$$\begin{aligned}\frac{dY_j(t;\omega)}{dt} =& \left\langle \Pi_Z^{\perp}\mathcal{L}^*[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega],\tilde{\boldsymbol{u}}_j(\boldsymbol{x},t)\right\rangle \\ &- \sum_{i=1}^{M}\left\langle \tilde{\boldsymbol{v}}_i(\boldsymbol{x},t),\tilde{\boldsymbol{u}}_j(\boldsymbol{x},t)\right\rangle \frac{dZ_i(t;\omega)}{dt}\,, \quad \forall j\in\{1,...,R\}\,.\end{aligned} \tag{5.22}$$

Finally, substituting equation 5.22 into equation 5.21, followed by projecting onto the subspace of Y_i's, we get the evolution equations for the unforced modes,

$$\begin{aligned}\sum_{i=1}^{R}\mathbb{E}[Y_i(t;\omega)Y_j(t;\omega)]\frac{\partial\tilde{\boldsymbol{u}}_i(\boldsymbol{x},t)}{\partial t} =& \mathbb{E}[\Pi_{\tilde{u}}^{\perp}\Pi_Z^{\perp}\mathcal{L}^*[\boldsymbol{u}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(t,\omega),\boldsymbol{x},t;\omega]Y_j(t;\omega)] \\ &- \Pi_{\tilde{u}}^{\perp}\left[\sum_{i=1}^{M}\tilde{\boldsymbol{v}}_i(\boldsymbol{x},t)\frac{dZ_i(t;\omega)}{dt}\right]\,.\end{aligned} \tag{5.23}$$

In case of stationary stochastic BCs, we have $\frac{dZ_i(t;\omega)}{dt}=0$, thus decoupling the equations for the forced modes, and the unforced modes and stochastic coefficients.

5.1.3 Application Results and Discussions

In order to compare the performance of the proposed method of strong imposition (section 5.1.2) of stochastic BCs, with their weak imposition (section 5.1.1), we use the flow past cylinder case with a stochastic inlet BC for our experiments. The domain setup is exactly the same as that used in section 2.3.4, but with the seamount replaced by a cylinder. The flow is governed by the Navier-Stokes equations with initial condition (IC) and inlet Dirichlet BC uncertainties. Stochastic initial conditions are specified for the velocity field, with the mean velocity field initialized using a strong perturbation, and the modes initialized using a boundary-mollified spatial covariance function [11]. A mean perturbation which is exponentially decaying from its origin is specified upstream and is asymmetric w.r.t. to the centerline to help induce the vortex shedding. While the stochastic inlet Dirichlet BC determines the effective Reynolds number for each realization. To create the inlet Dirichlet bound-

ary profiles, we use an ensemble of fourth order polynomial, such that all the profiles are confined between parabolas corresponding to a Reynolds number between 40 and 140, as shown in figure 5-1. In our experiments, we pick the realizations corresponding to representative low and high inlet velocities, and rely on comparisons with the corresponding Monte Carlo (MC) runs.

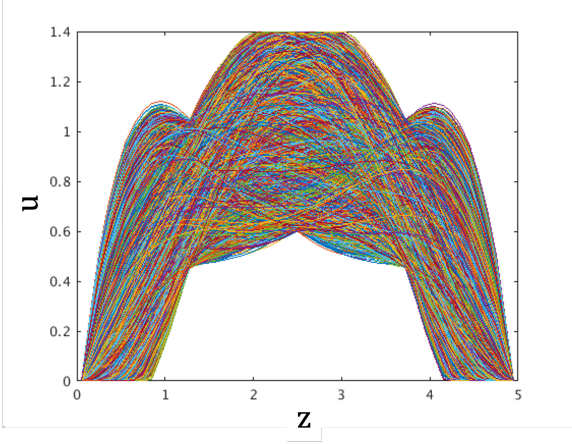


Figure 5-1: Realizations of the stochastic boundary condition for the inlet Dirichlet for the u-velocity field. The horizontal axis corresponds to the z-axis of the domain (figure 2-1), and vertical axis is the magnitude of velocity.

First, we analyze the performance of the weak imposition of stochastic BCs. For the DO simulation, 10 modes and 10,000 realizations were used. The initial statistics of the setup are presented in figure 5-2. The flow is allowed to develop until $T = 50$, and the evolved statistics are shown in figure 5-3. Focusing on the u and v velocity standard deviation fields in figure 5-3, we can notice that there is only uncertainty in the wake of the cylinder, and none near the inlet, thus pointing towards the fact that the inlet BC uncertainty is misaligned w.r.t. interior uncertainty. We also pick two realizations for our comparison, one corresponding to a low inlet velocity (realization #496) and another for a high inlet velocity (realization #7203). The inlet velocities are also directly proportional to the strength of the vortex shedding. As compared to their MC counterparts started from exactly the same ICs, in figure 5-4, we notice vortex shedding of the wrong strengths for either of these realizations in their approximate solutions.

Second, we analyse the performance of strong imposition of stochastic BCs. Once

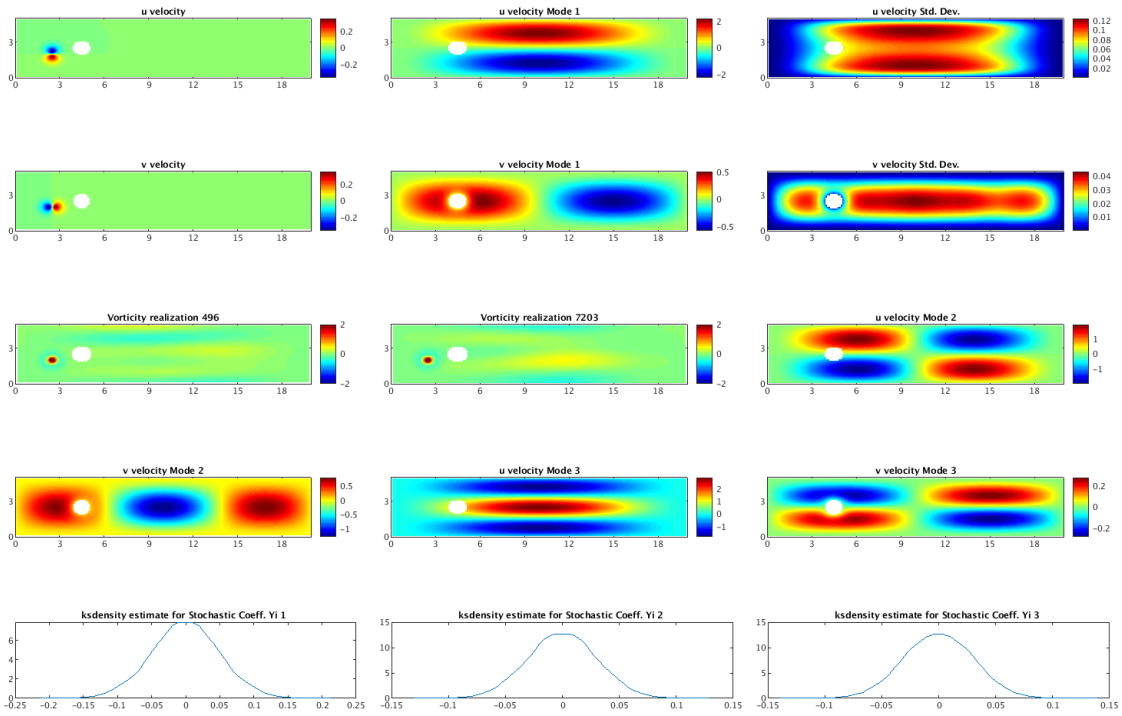


Figure 5-2: Initial condition statistics for the experiment corresponding to the weak imposition of stochastic boundary conditions. The first and the second rows correspond to $u-$ and $v-$ velocities respectively, with mean field, first mode, and standard deviation fields going left to right. In the third row, going from left to right, the first two corresponds to vorticity for reconstructed DO realizations #496 and #7203, and the third is $u-$velocity second mode. In the fourth row, going from left to right, the first is second $v-$velocity mode, followed by the third modes for $u-$ and $v-$velocities respectively. The last row corresponds to kernel density fits for the first three stochastic coefficients.

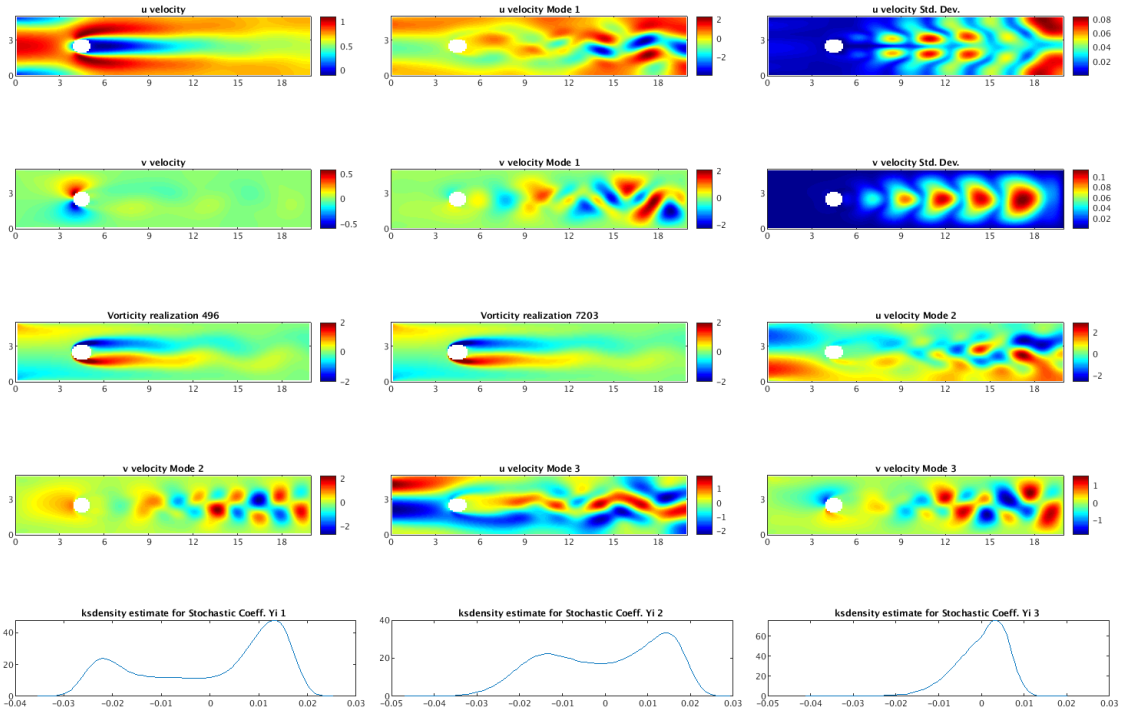
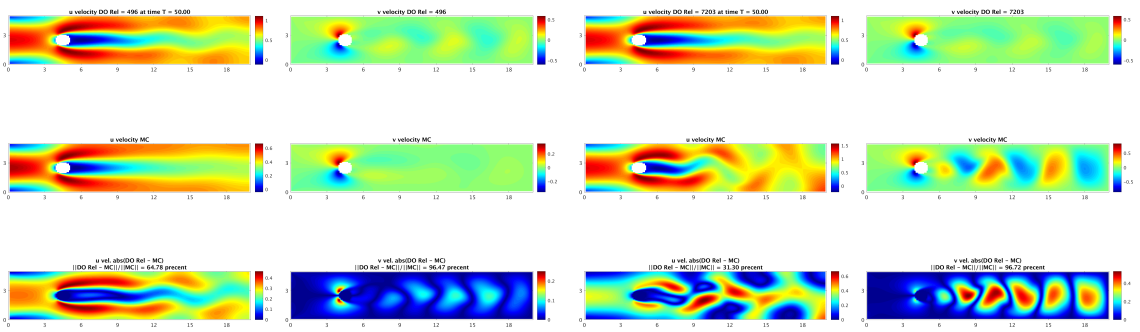


Figure 5-3: Statistics for the experiment corresponding to the weak imposition of stochastic boundary conditions, at time $T = 50$. Description is same as figure 5-2.



(a) Realization #496 (b) Realization #7203

Figure 5-4: Comparison between reconstructed DO realizations and the corresponding monte carlo runs for the experiment with weak imposition of stochastic boundary conditions, at time $T = 50$. The left and right columns corresponds to the $u-$ and $v-$ velocities respectively. The first row corresponds to the reconstructed DO realization, the second to the monte carlo run, and the third is their absolute difference, along with the relative % of spatial average of L_2 error in the title.

again, we choose 10,000 realizations, and a total of 10 modes, with 7 unforced and 3 forced modes. The initial conditions are presented in figure 5-5, and are the same as the previous experiment, for the 7 unforced modes. The forced modes are zero in the interior initially, but with non-zero boundary values which are assigned based on the dominant singular vectors of the BC ensemble. Thus, there is indiscernible difference between the initial conditions for the previous and the current experiments in figures 5-2 and 5-5, respectively. The flow is allowed to develop until $T = 50$, and the statistics are presented in figure 5-6. In this experiment, we can notice a comparatively higher strength of uncertainty throughout the domain. Specifically, the u velocity standard deviation near the inlet shows the impression of the BC uncertainty. In figure 5-7, the instantaneous flow of realizations #496 and #7203 also qualitatively matches their MC counterparts. The difference between the MC realizations and the approximate solutions are significant, however, and could be attributed to the phase mismatch in the shedding.

We also plot the temporal variation of the space-averaged standard deviation value for both u-, and v-velocities in figure 5-8. For the strong stochastic BC imposition experiment, we compute both, the standard deviation due to only the contribution

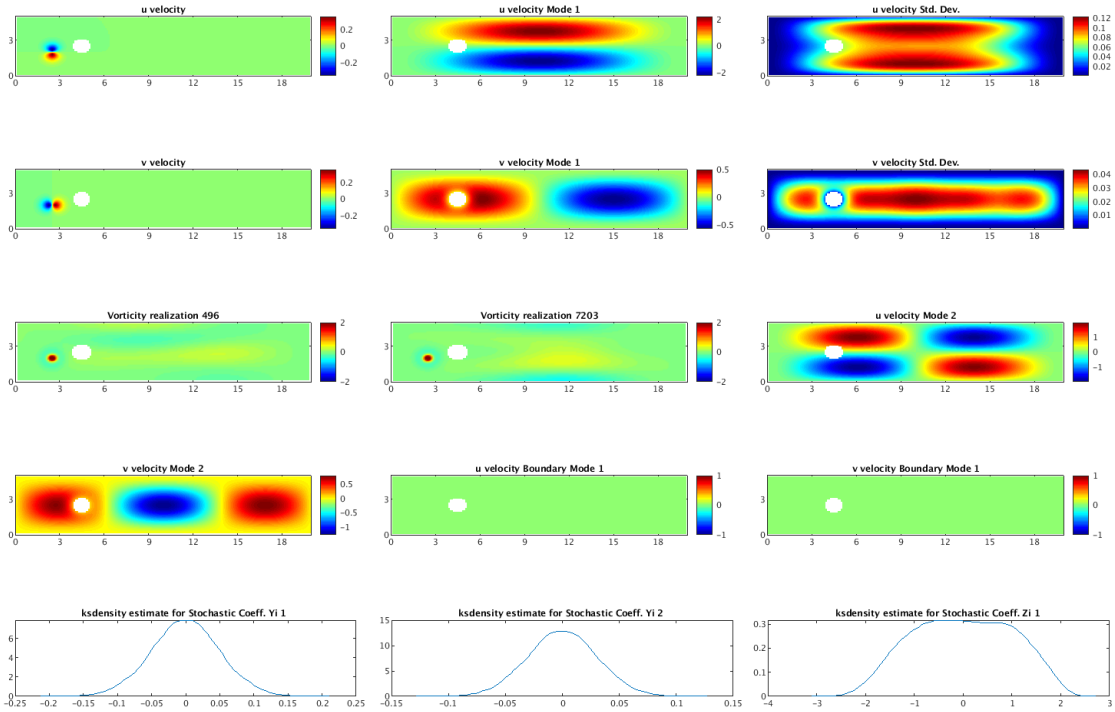


Figure 5-5: Initial condition statistics for the experiment corresponding to the strong imposition of stochastic boundary conditions. Description is same as figure 5-2.

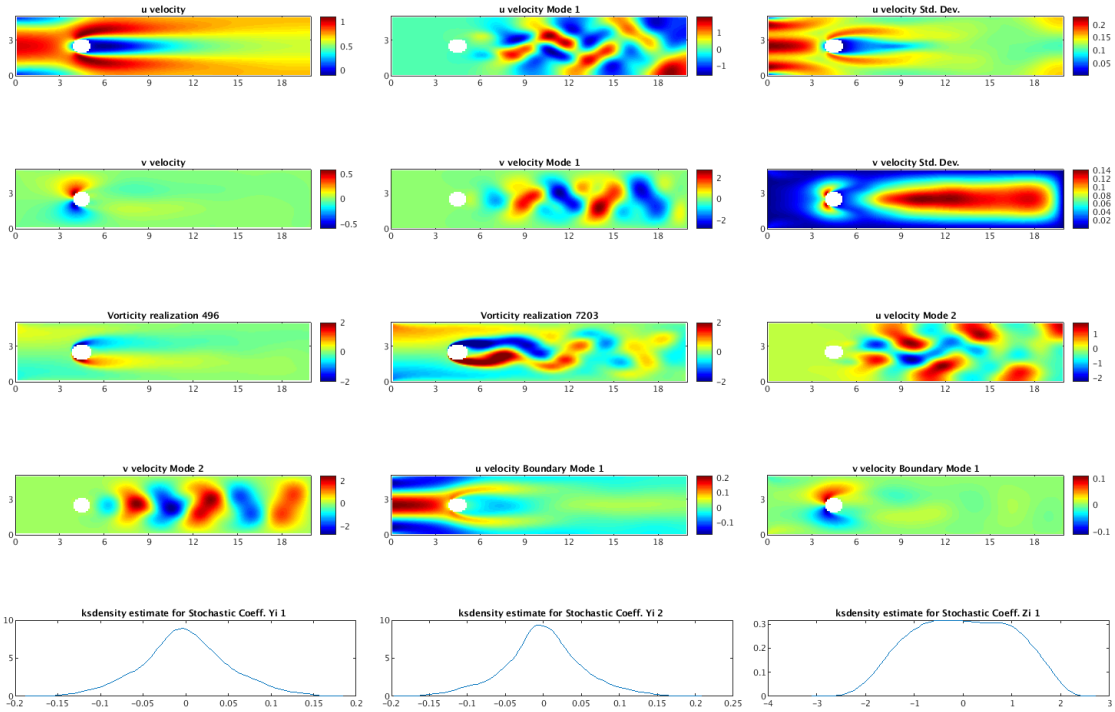
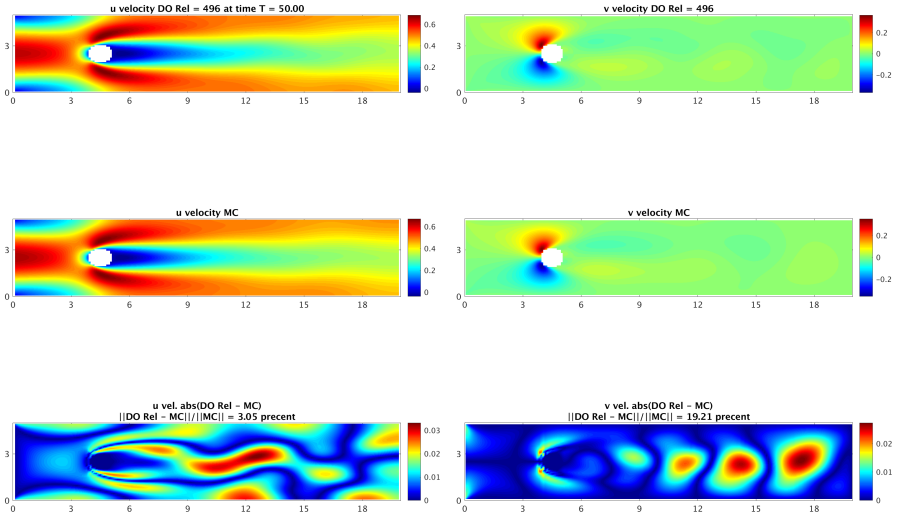
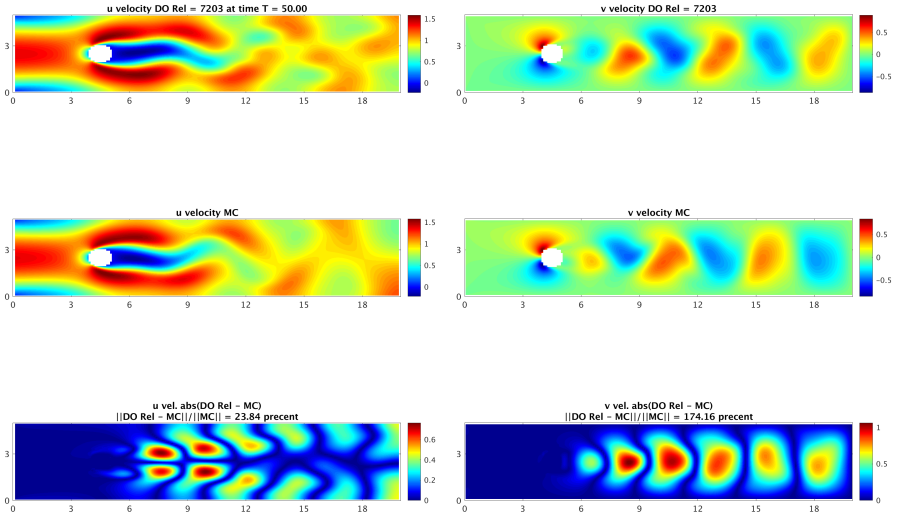


Figure 5-6: Statistics for the experiment corresponding to the strong imposition of stochastic boundary conditions, at time $T = 50$. Description is same as figure 5-2.



(a) Realization 496

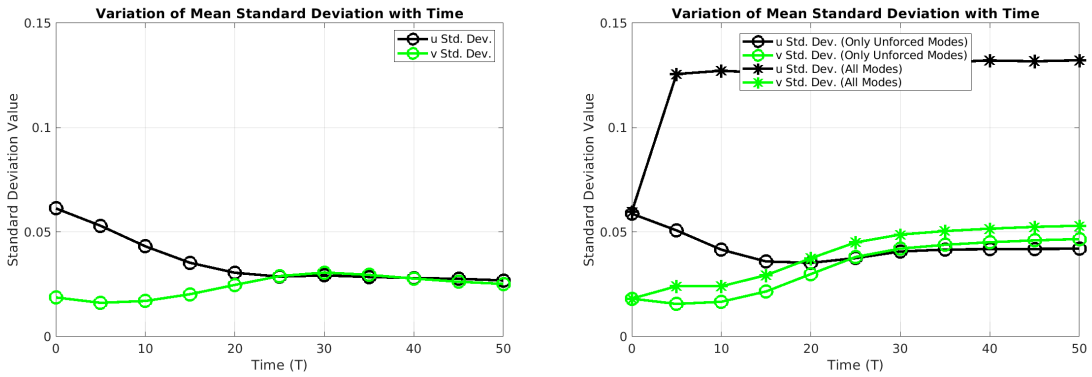


(b) Realization 7203

Figure 5-7: Comparison between reconstructed DO realizations and the corresponding monte carlo runs for the experiment with strong imposition of stochastic boundary conditions, at time $T = 50$. Description is same as figure 5-4.

from unforced modes, and from both the unforced and forced modes combined. It can be noticed that the contribution to standard deviation from just the unforced modes in case of strong imposition is of similar magnitude as the contribution of all the

modes in case of weak imposition. Thus, as expected, all the additional uncertainty due to the stochastic inlet is captured by the forced modes when the method of strong imposition of stochastic BCs is used.



(a) Weak imposition of stochastic BCs

(b) Strong imposition of stochastic BCs

Figure 5-8: Variation of spatially averaged standard deviation over time. *Black* lines corresponds to $u-$velocity, and *green* lines corresponds to $v-$velocity. In *(b)*, the lines marked with "$\circ$" only accounts for standard deviation computed using the unforced modes, while the lines marked with "$*$" accounts for both unforced and forced modes.

5.2 Numerical Challenges and Implementations

In the present thesis, the numerical experiments involve complex geometries and multidisciplinary dynamics. Thus, in this section, we elaborate on the methods adopted to address the numerical issues pertaining to these specific requirements.

5.2.1 Ghost Cell Immersed Boundary Method

For most of the Bayesian learning experiments in this thesis, we use a 2D finite-volume (FV) framework [80] which uses an uniform staggered Cartesian grid for the spatial discretization. Due to the rectangular nature of the grid, boundaries with complex and smooth geometries are approximated as a staircase. This means that the boundary conditions are actually enforced on the approximate staircase boundary which has ramifications on the accuracy and conservation properties of the numerical schemes. The literature is abound with methods to accurately handle complex boundaries

with structured Cartesian grids, and are popularly termed as immersed boundary (IB) methods. Mittal and Iaccarino, 2005 [138] provides a comprehensive review of IB methods.

Based on the code structure of the existing FV framework, ease of implementation, and current requirements, the ghost-cell immersed boundary method (GCIBM) proposed in Tseng and Ferziger, 2003 [139] was chosen to be implemented to increase the accuracy in handling complex boundary shapes. In this method, based on the location of the actual boundaries, the center of FV cells are divided in to fluid, ghost, and solid cells, as shown in figure 5-9. For each ghost cell, we identify an image point, and then use biliniear interpolation to calculate the state variable at that image point. Followed by this, we simply enforce the exact boundary condition at center of the ghost cell and the image point, which also happens to lie at the actual boundary. We implemented the GCIBM method to handle both Dirichlet and Neumann BCs, and for both DO mean and modes equations. This method preserves up to second order accuracy of the numerical schemes. For more implementation and algorithmic details, the reader is referred to Tseng and Ferziger, 2003 [139].

We utilize a deterministic Couette flow experiment to test the implementation of the GCIBM in the 2D FV framework [80]. The setup consists of two concentric rings rotating at different angular velocities and no slip boundary conditions, as shown in figure 5-10. This is a challenging experiment due to the circular boundaries embedded in a uniform Cartesian grid. The analytical velocity field in cylindrical coordinates is given by,

$$u_\theta = \left(\frac{R_2^2\omega_2 - R_1^2\omega_1}{R_2^2 - R_1^2}\right) r + \left(\frac{\omega_1 - \omega_2}{R_2^2 - R_1^2}\right) \frac{R_1^2 R_2^2}{r}\,, \quad \text{and} \quad u_r = 0\,, \tag{5.24}$$

where R_1 and ω_1 are respectively the radii and the angular velocity of the inner cylinder, while R_2 and ω_2 are that of the outer cylinder. We solve the Navier-Stokes equations using numerical schemes that are second order accurate both in space and time. For comparison, the analytical $u-$velocity, along with the absolute difference w.r.t. to the staircase approximation of the boundary and GCIBM at a grid size of

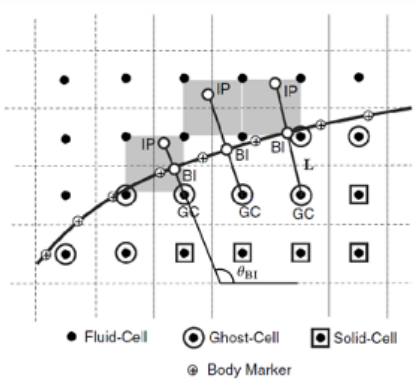


Figure 5-9: Schematic denoting the split of underlying Cartesian grid cells into fluid-, ghost-, and solid-cells based on the location of the cell-center relative to the actual boundary denoted by the solid curve with "$\oplus$" symbol. Symbol "$\circ$" with "IP" marks the location of the image points for the corresponding ghost-cells, and lies on the perpendicular drawn from the ghost-cell to the actual boundary curve.

30×30, are presented in figure 5-11. We can clearly see that the maximum error occurs near the boundaries in the case of the staircase approximation. We further perform a convergence study by computing the average error incurred at three different grid resolutions. A log-log plot between the average error and the number of grid-points in one direction is provided in figure 5-11d, and as expected, the implementation of GCIBM maintains the original second-order accuracy of the spatial discretization of the existing numerical schemes in the FV framework.

5.2.2 Advection Schemes

Advection is the most problematic and tricky term to handle in the Navier-Stokes or the advection-diffusion-reaction equations, due to its non-linear nature. Advection schemes which remain both stable and accurate in the presence of steep gradients and shocks, such as upwinding, total variation diminishing (TVD), and essentially non-oscillatory (ENO) schemes, invariably use diverse rules depending on the sign of the advecting velocity. This does not pose any challenge in the case of evolution of stochastic PDEs using DO methodology in which only tracer fields are uncertain and one could easily use any of the existing non-linear advection schemes. However, when uncertainty is introduced in the velocity fields, the use of these non-linear ad-

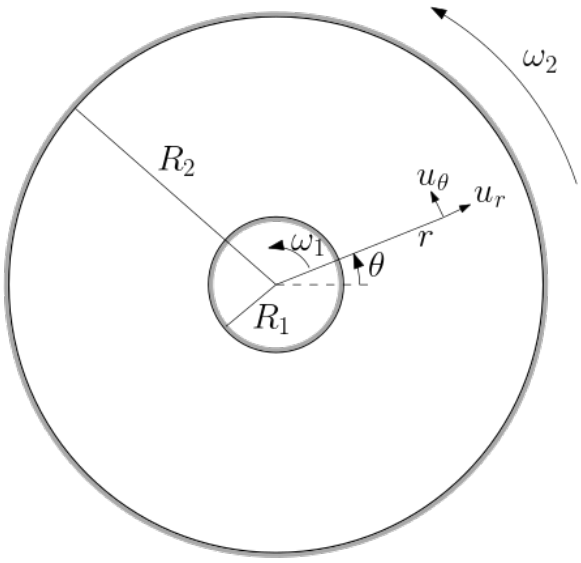
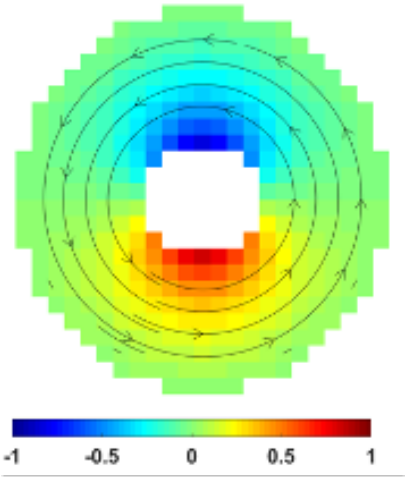


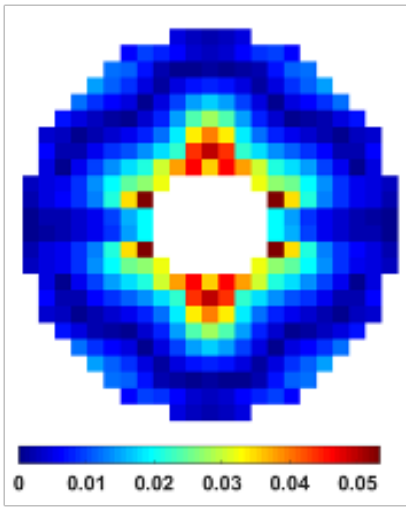
Figure 5-10: Schematic for two concentric cylinders rotating at different angular velocities with fluid in between. "r" and "θ" denotes the cylindrical coordinate system.

vection schemes poses a challenge. This is due to the fact that the velocity modes do not contain any directional information, and it would be computationally ineffiient to recreate all realizations and explicitly examine their individual velocity directions. As investigated by Feppon and Lermusiaux, 2018 [14], only linear advection schemes preserve the DO decomposition. Thus, we use central difference (CD) schemes. However, CD schemes are unstable with the simple Euler time stepping, hence, we implement them in conjugation with Heun's method for time-integration in the FV framework [80]. In addition, the use of CD schemes for advection is known to cause spurious oscillations. To remove spurious oscillation to stabilize the system, we use a popular spatial filter, called the Shapiro filter. Apart from being easy to use and implement, Shapiro filters are also linear, thus preserving the DO decomposition.

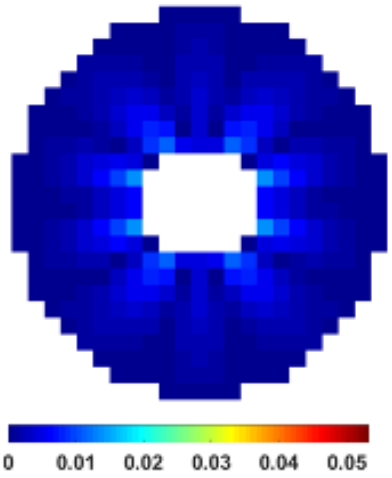
We utilize a simple experiment of flow past a seamount with initial condition and Reynolds number uncertainties in the Navier-Stokes equations, and coupled with a 3-component NPZ model. We use 60 DO modes and 10,000 realization. Other details of the setup are similar to that described earlier in sections 2.3 & 3.3. The flow is evolved using a previous version of the FV framework [80] which utilizes a TVD scheme for both DO mean and modes equations, along with Euler time-stepping. For the terms being advected by the velocity modes, an average value is used, computed using the



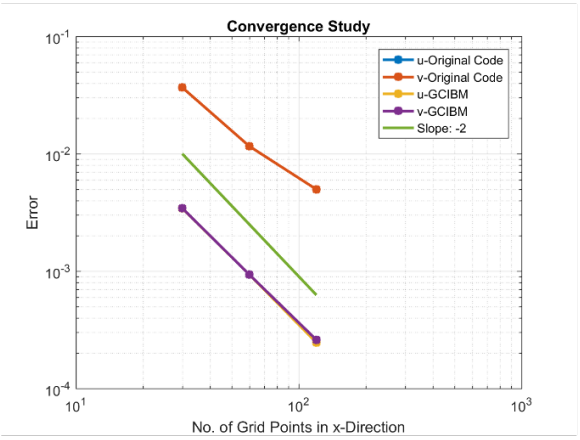
(a) Analytical $u-$velocity



(b) |Analytical - Staircase| $u-$velocity

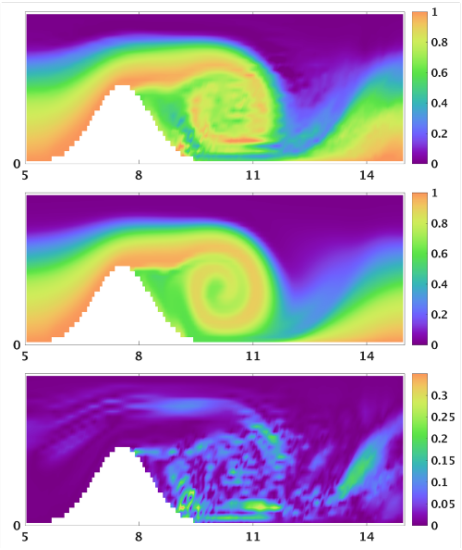


(c) |Analytical - GCIBM| $u-$velocity

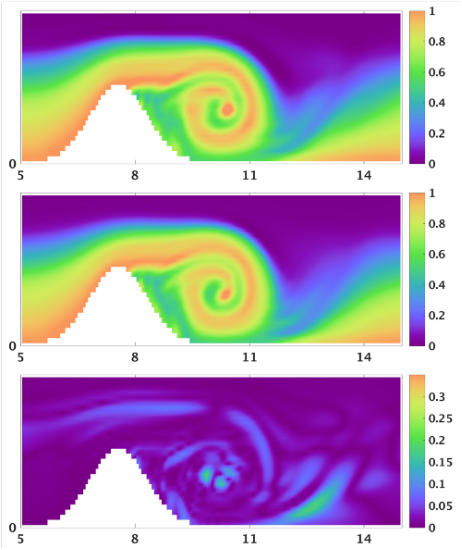


(d) Convergence Study

Figure 5-11: Experiment with deterministic Couette flow between two concentric cylinders rotating at different angular velocities. *(a)* Analytical $u-$velocity in the Cartesian coordinates (equation 5.24); *(b)* Absolute difference between the analytical $u-$velocity and that computed numerically with staircase approximation of the curved boundaries; *(c)* Same as (b) but with numerical solution computed with ghost-cell immersed boundary method (GCIBM) for the boundaries; *(d)* Variation of spatially averaged error in $u-$ and $v-$velocities w.r.t. the analytical solution, and corresponding to different grid-resolutions. "Original code" refers to the use of staircase approximation.



(a) Total variation diminishing (TVD) scheme and Euler time-stepping



(b) Central difference scheme, Shapiro filter (8, 1, 5), and Heun's method for time-integration

Figure 5-12: Reconstructed nutrient field realization at time $T = 5$ and $Re = 100$, computed using different numerical schemes for advection and time-integration. The *top* plots corresponds to the approximate DO solution; the *middle* plots to the monte carlo simulation; and the *bottom* plots to their absolute difference. The numerical schemes used are mentioned in the respective captions. The trio of (8, 1, 5) denotes the order, number of applications, and time-step frequency of application of the Shapiro filter.

TVD scheme with original and opposite signs of the velocity modes. Using the same initial conditions, and grid resolution, the flow is also evolved using a second-order CD scheme for space, Heun's method for time integration, and a 8^{th} order Shapiro filter applied once every 5 time-steps. In figure 5-12, we compare a reconstructed nutrient field realization after the evolution of the flow up until time $T = 5$, and corresponding to the Reynolds number 100. We notice the presence of spurious oscillations only in the case of TVD and Euler schemes, thus demonstrating both the need and advantage of using linear advection and higher-order time-integration schemes, with filtering.

5.3 Data Assimilation with Subspace Augmentation and Adaptive Covariance Inflation

A data assimilation step is basically an application of the Bayes rule under some assumptions, thus, it follows the constraint that the posterior of the augmented states and parameters should be contained within the support of the corresponding prior. In the case of data assimilation in a low-dimensional subspace, this also translates to the fact that the subspace cannot be modified based on the observations in a Bayesian sense. If the subspace is relatively off w.r.t. to the ground truth, it leads to discarding some of the information contained in the observations. However, this discarded information could inform us about the mismatch between the ground truth and the subspace. The goal of subspace augmentation is to utilize such information to correct the subspace.

We will utilize the methodology for data-driven subspace augmentation developed in Lin, 2020 [9] at every data assimilation step, to demonstrate the ability for meeting our learning objective even when starting with a subspace of insufficient size to capture the uncertainty evolution of a high-dimensional nonlinear system. Let the observation be denoted by $\hat{\boldsymbol{y}} \in \mathbb{R}^{N_Y}$, while the observation space is as defined by the observation model equation 2.3 and is denoted by $\mathcal{O} \in \text{span}\{\boldsymbol{H}\tilde{\boldsymbol{U}}\} \in \mathbb{R}^{N_Y \times S}$. The methodology for subspace augmentation first involves finding an equivalent observation $\hat{\boldsymbol{y}}_* \in \mathcal{O}$, such that replacing $\hat{\boldsymbol{y}}$ with $\hat{\boldsymbol{y}}_*$ in the Bayesian update will yield the exact same posterior. The unused part of the innovation vector, $(\hat{\boldsymbol{y}} - \hat{\boldsymbol{y}}_*)$, is then fed to a goodness-of-fit test to determine if the existing subspace is sufficiently incorrect to warrant an augmentation or not. If the p-value (popular statistical measure used for hypothesis testing; [140]) is small, the existing subspace is amended with a rank-1 update such that the modified $\mathcal{O}$ contains $\hat{\boldsymbol{y}}$. The DO coefficient corresponding to this new mode is initialized using uncorrelated Gaussian noise with zero mean and with a variance equal to that of the last mode of the existing prior. Further, most of the times it is also the case that the uncertainty in our prior is undermined by the model errors we have not taken into account. Thus, to avoid over-confidence in what the

truth should be, we incorporate a safety measure by inflating the prior uncertainty using the adaptive covariance inflation methodology developed in Lin, 2020 [9]. For algorithmic details, the reader is referred to the sections 5.4.4 and 5.5.1 of Lin, 2020 [9].

For our experiments, we consider the 3-component NPZ model with uncertainties in the initial conditions of the biological states, Ivlev grazing parameter, and the functional form of the zooplankton mortality term, whether it is quadratic or not. The setup is the same as earlier in sections 2.3 & 2.4.1, a flow past a seamount with sparse observations collected downstream. The goal is to learn all the biological states, regular parameter Λ, and the functional form of the zooplankton mortality using a special stochastic parameter, simultaneously, by assimilating the observations. The methodology to initialize the state fields, parameters, etc. is the same as earlier. However, we only use 2 DO modes, which are insufficient to quantify the evolving uncertainty in the system. We utilize noisy observations measured at 8 locations. Values of the various parameters associated with the experiments are provided in table 5.1. We perform two experiments with an identical initial setup, however, one with and the other without data-driven subspace augmentation and adaptive covariance inflation. In figures 5-13 and 5-14, we provide the mean of the posterior biological states, pdf of the parameters after the last assimilation step ($T = 25$), in addition to the variation of normalized RMSE over time. We can notice that with subspace augmentation and covariance inflation, RMSE either decreases or more-or-less remains same at every assimilation step (except for $T = 13$ which corresponds to a dynamics related event; figure 5-13). However, without subspace augmentation and covariance inflation, the change in RMSE is quite random (figure 5-14). In the latter case, the pdf of Ivlev grazing parameter is bi-modal, thus demonstrating ambiguity in the learned value of the parameter. Using the subspace augmentation and covariance inflation however helps with more accurate evolution of uncertainty, hence leading to more confidence and better learning of the Ivlev parameter. We do not see any perceivable differences in the mean fields. We also provide the increase in the size of the subspace over time in figure 5-15, as the number of modes are raised to 8 starting

from 2.

For further comparison, we conduct another experiment, again without subspace augmentation and adaptive covariance inflation, however, this time we use 8 DO-modes. This is equal to the maximum size of the subspace reached at the end of the experiment with subspace augmentation and adaptive covariance inflation. The posterior state after the last assimilation step is provided in figure 5-16. Especially in the pdf of the Ivlev parameter, we again notice multiple peaks, and the highest peak is also clearly off than the true parameter value. This demonstrates that even a subspace of size 8 is insufficient to capture the evolving uncertainty correctly. An accurate evolution of uncertainty is essential for obtaining a representative prior, and in turn a sufficiently accurate posterior.

5.4 Observation Planning

In real-world applications where data collection is a luxury, having a scientifically sound knowledge about *what*, *when*, and *where* to observe is crucial for achieving the learning objectives. The use of *mutual information (MI)* for determining optimal observation locations has been previously explored in our group, and developed to work in conjugation with the GMM-DO filter [9, 141]. In the present work, apart from extending the use of MI-based optimal observation location selection for n multidisciplinary problems, we will also utilize MI for the purpose of identifiability and predictability. Finally, we will showcase applications to realistic ocean simulations.

5.4.1 Computing Mutual Information (MI)

MI can be computed between any two random variables (RVs), without any constraint on them being scalars, or vectors of the same dimensions. MI between random variables, $\boldsymbol{X}$ and $\boldsymbol{Y}$, is an information-theoretic measure of the amount of relevant information contained in one w.r.t. the other variable, and is mathematically given

Parameters	**Values**
Biogeochemical model	NPZ
Diffusion constants both horizontal and vertical, $\mathcal{K}$	0
Light attenuation due to sea water, k_w (m^{-1})	0.067
Initial slope of the P-I curve, α $((W\ m^{-2}\ day)^{-1})$	0.025
Surface photosynthetically available radiation, I_o $(W\ m^{-2})$	158.075
Phytoplankton maximum uptake rate, V_m (day^{-1})	1.5
Half-saturation for phytoplankton uptake of nutrients, K_u^* $(mmol\ N\ m^{-3})$	1
Phytoplankton specific mortality rate, Ξ (day^{-1})	0.1
Zooplankton specific excretion and mortality rate, Γ (day^{-1})	0.145
Presence of absence of quadratic zooplankton term, a	unif$\{0, 1\}$
Quadratic zooplankton specific excretion and mortality rate, Γ_2 (day^{-1})	0.2
Zooplankton maximum grazing rate, R_m (day^{-1})	0.52
Ivlev constant, Λ $((mmol\ N\ m^{-3})^{-1})$	unif(0.1, 0.2)
Fraction of zooplankton grazing egested, γ	0.3
Detritus decomposition rate, Φ (day^{-1})	1.03
Inverse of Eddy viscosity based Reynolds number, Λ_{Re}	1
Number of Modes, N_S	See text
Number of MC samples, N_{MC}	1000
State being observed	Z
Observation error standard deviation, $\sqrt{\boldsymbol{R}}$	0.05
Measurement noise standard deviation	0.03
Size of Observation vector, N_Y	8
Observation start time	1
Time interval between assimilation steps	2
Observation end time	25
p-value threshold	0.1%
Maximum covariance inflation, ρ_{max}	1.1

Table 5.1: Values of the various biological and hyper- parameters used in data-driven subspace augmentation experiments. $N_T = 30\ mmol\ N\ m^{-3}$, $H = 50\ m$ and time-scale of 1 day, are the scales used for non-dimensionalization.

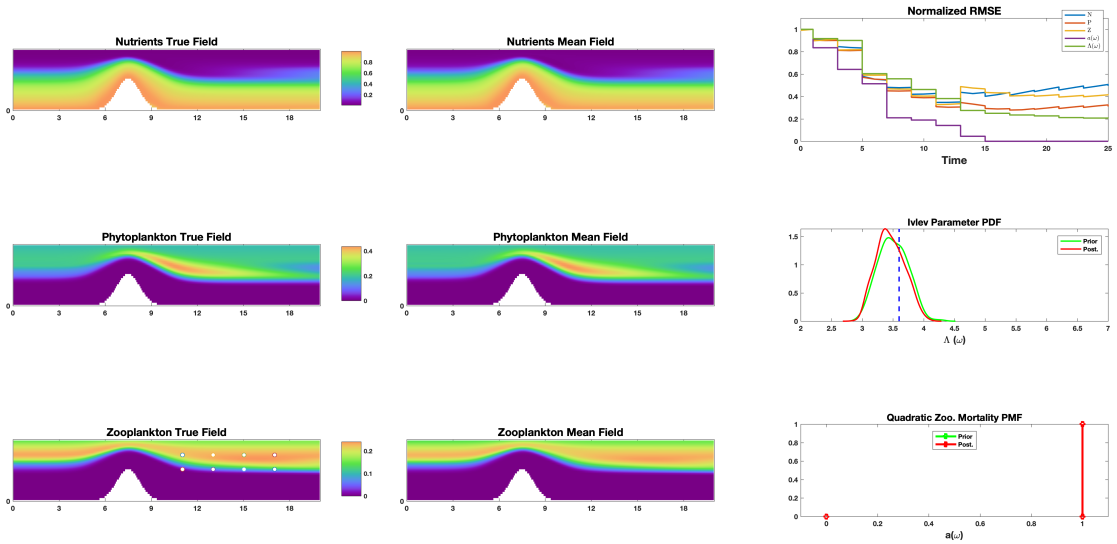


Figure 5-13: The posterior state of the NPZ model based stochastic dynamical system used in experiment **with** subspace augmentation and adaptive covariance inflation, and starting with only 2 DO-modes, at $T = 25$ (i.e. after 13 observational episodes). The first two columns consist of the true (left) and mean (right) field of the N, P and Z tracer fields. In the third column, the first plot will show the variation of normalized root-mean-square-error (RMSE) with time for various stochastic state variables and parameters. The next two plots contain the probability distribution of $\Lambda(\omega)$, and $a(\omega)$ (to learn presence or absence of quadratic zoo. mortality), with their true values marked with blue dotted lines. The velocity field is deterministic with $Re = 1$. The white circles on the zooplankton true field marks the observation locations.

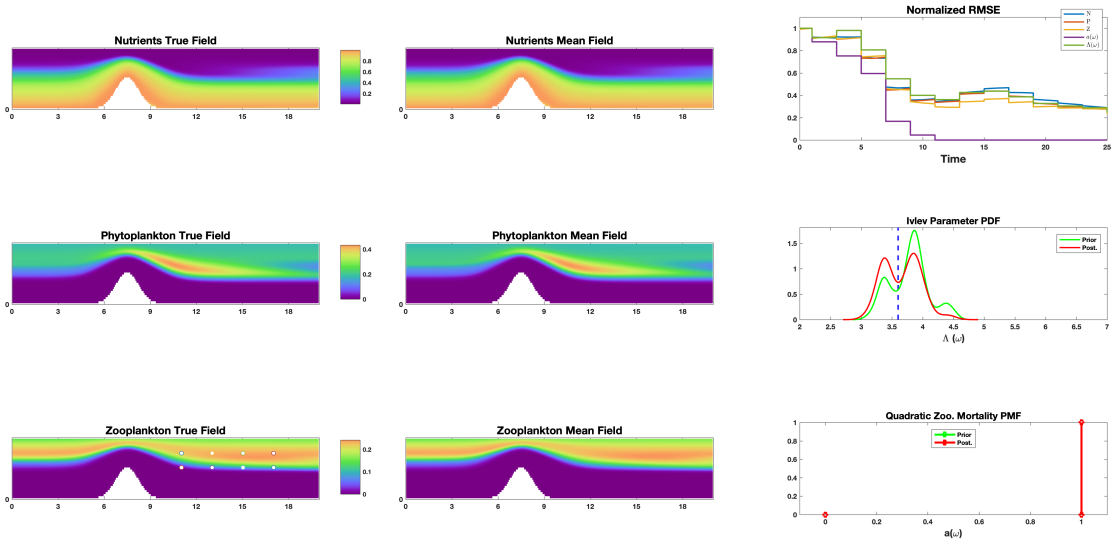


Figure 5-14: The posterior state of the NPZ model based stochastic dynamical system used in experiment **without** subspace augmentation and adaptive covariance inflation, and with only 2 DO-modes, at $T = 25$ (i.e. after 13 observational episodes). Description same as figure 5-13.

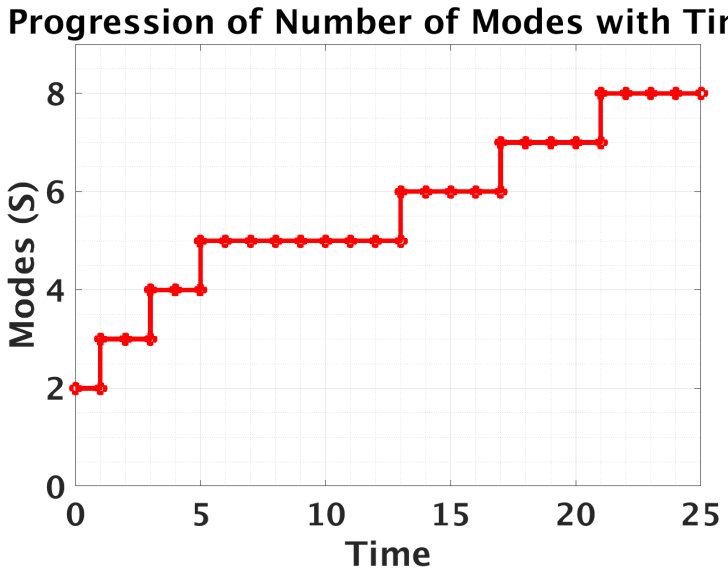


Figure 5-15: Increase in DO modes with time for the experiment with subspace augmentation and adaptive covariance inflation. The experiment is started with just 2 modes, and they increase up to 8 in number.

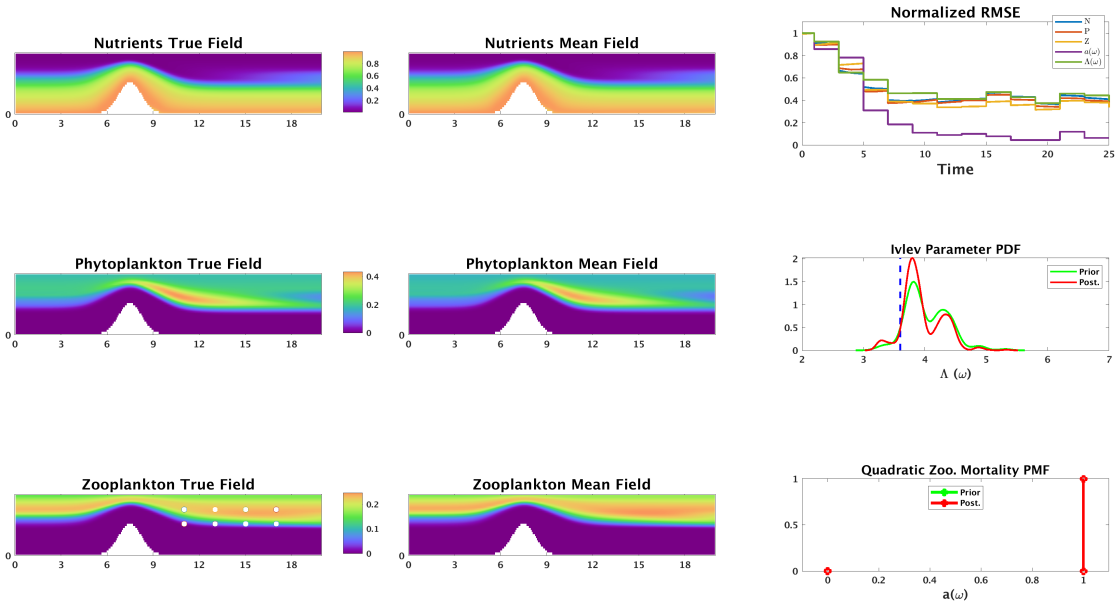


Figure 5-16: The posterior state of the NPZ model based stochastic dynamical system used in experiment **without** subspace augmentation and adaptive covariance inflation, and with 8 DO-modes, at $T = 25$ (i.e. after 13 observational episodes). Description same as figure 5-13.

by [142],

$$I(\boldsymbol{X};\boldsymbol{Y}) = \int_{x\in\boldsymbol{X}}\int_{y\in\boldsymbol{Y}} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy\,. \tag{5.25}$$

Computing MI is non-trivial, however, there exists a closed-form expression for Gaussian distributions. Thus, one could in principle fit a Gaussian to the joint of the RVs, and MI can be computed analytically [143]. This however does not lead to an accurate result in the case of non-Gaussian distributions. A GMM-based non-Gaussian method has been developed in Lolla, 2016 [141] and Lin, 2020 [9]. Starting with the samples from $\boldsymbol{X}$ and $\boldsymbol{Y}$, the methodology first involves fitting a GMM to the joint of $\boldsymbol{X}$ and $\boldsymbol{Y}$. After a GMM is fitted, expectation of the logarithm factor in the MI (equation 5.25) is approximated by drawing samples from the GMM, evaluating the logarithm factor at each realization using the GMM pdf, and computing the expectation by taking the average, also called the Monte Carlo method. Further, due to the transformation invariant property of MI, we can equivalently compute MI in the DO subspace which makes it computationally efficient.

We can use a simple experiment to demonstrate the advantage of using GMM-based MI computation. This experiment will also simultaneously demonstrate the advantage of using MI over covariance, and GMM-based non-Gaussian filter over the classic Kalman filter. We consider two scalar RVs with a parabolic joint distribution,

$$X \sim \mathcal{N}(0,1) \quad \text{and} \quad Y = X^2 + \mathcal{N}(0,\sigma^2)\,, \tag{5.26}$$

where $\mathcal{N}(0,1)$ is a normal distribution, and an uncorrelated noise with $\sigma << 1$ is added to avoid numerical issues. 10,000 random samples are plotted in figure 5-17, and intuitively, if we were to observe one of the RV, we would expect to gain knowledge about the other RV. However, covariance as an information measure for this distribution leads to the opposite conclusion, because of it being much smaller than 1. Along with this, computing MI using a Gaussian fit gives a value of 0.0258. Though, just knowing the magnitude of MI does not contribute to our intuition about the amount of information contained in one variable for the other, however, the value of 0.0258 does seem to be low and goes against our original intuition. On

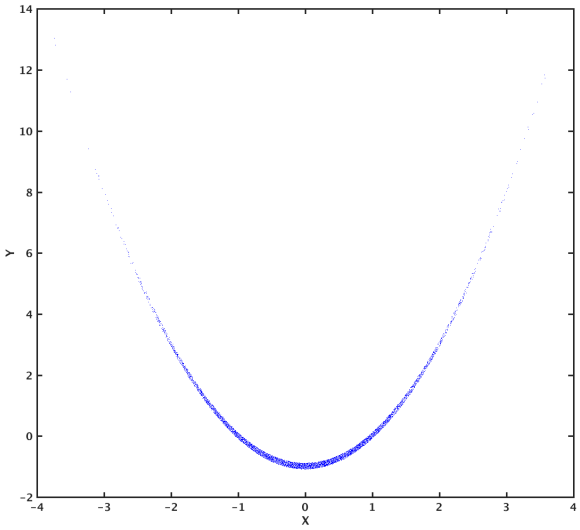
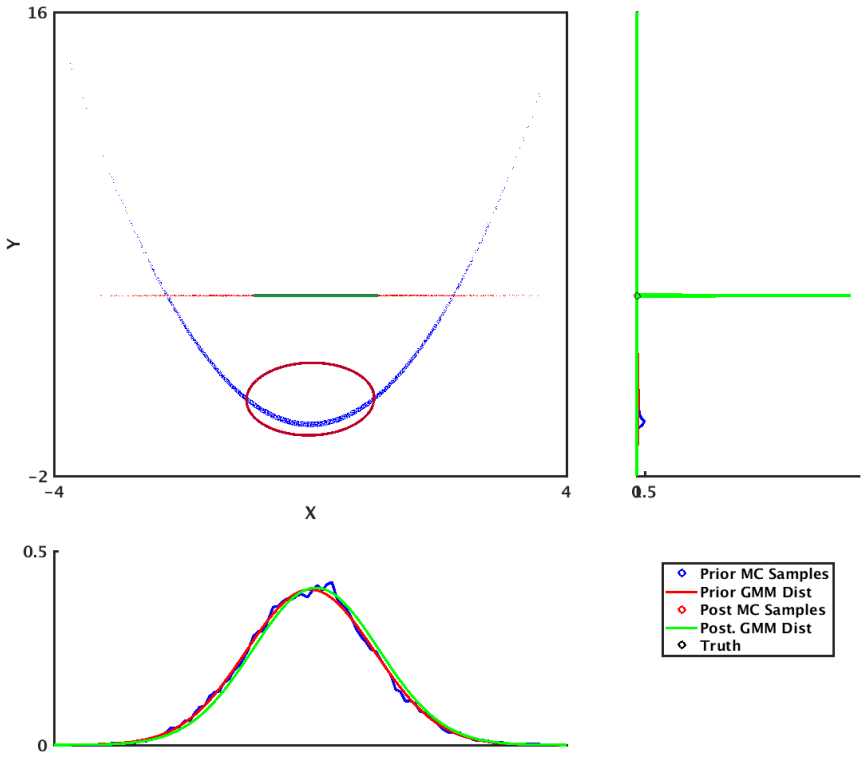


Figure 5-17: Joint sample distribution for $X \sim \mathcal{N}(0,1)$ and $Y = X^2 + \mathcal{N}(0,\sigma^2)$ with $\sigma << 1$.

the other hand, computing MI using a fit of 10 GMMs gives us a value of 1.251 for the same system, which seems more reasonable. In order to corroborate our original intuition, let us observe the RV $\boldsymbol{Y}$, and infer $\boldsymbol{X}$. For comparison, we assimilate the data using the classic Kalman filter and the GMM-DO filter, and the results are provided in figure 5-18. A fit of 10 GMM and 2 DO modes were used for the GMM-DO filter. Ideally, the information about the value of $\boldsymbol{Y}$ should make the posterior concentrate around the corresponding $\boldsymbol{X}$ intersections along the parabola, and this is only achieved with the GMM-DO filter as it is able to capture the non-linear and non-Gaussian relationship. The Gaussian assumption in the classic Kalman filter is insufficient for this experiment.

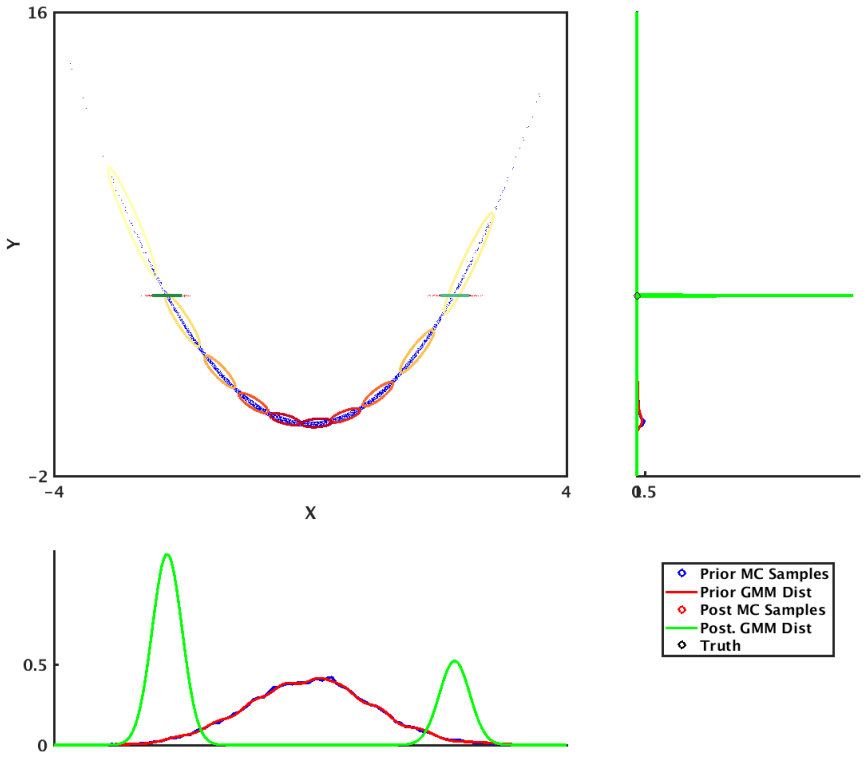
5.4.2 Optimal Locations

Due to the vastness of the ocean and the challenges associated with running *in-situ* experiments, ocean observations are both sparse in space and time. In such a resource-constrained environment, picking the observation locations with the most amount of information for your learning objectives at a given time is essential, yet a non trivial



(a) Kalman filter (GMM with 1 component)

Figure 5-18: The result of data assimilation after observing the value $y = 5$ and using different filters. The *top-left* plot corresponds to the joint distributions, while the *right* and the *bottom* ones showcase the marginals of Y and X respectively. The *dots* denotes the monte carlo samples and the *lines*, kernel density fits. The *ellipses* mark the 1st standard deviation of the Gaussians and the color intensity their individual normalized weights, with darker shades of *red* mapping to 1 and lighter to 0, for the prior Gaussian-Mixture-Model (GMM) fit. The shades of *green* marks the same, however, for the posterior GMM fits. The *black dot* marks the observed true value. *(Cont.)*



(b) GMM-DO filter

Figure 5-18: The result of data assimilation after observing the value $y = 5$ and using different filters. The *top-left* plot corresponds to the joint distributions, while the *right* and the *bottom* ones showcase the marginals of Y and X respectively. The *dots* denotes the monte carlo samples and the *lines*, kernel density fits. The *ellipses* mark the 1st standard deviation of the Gaussians and the color intensity their individual normalized weights, with darker shades of *red* mapping to 1 and lighter to 0, for the prior Gaussian-Mixture-Model (GMM) fit. The shades of *green* marks the same, however, for the posterior GMM fits. The *black dot* marks the observed true value.

task. Due to the fine computational grid used, the number of possibilities to test grow exponentially. For example, given the size of your state as $\boldsymbol{U} \in \mathbb{R}^{N_X}$, if one wants to pick a combination of $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{N_Y}$ number of locations, then a simple combinatorial search requires $\frac{N_X!}{(N_X-N_Y)!N_Y!}$ number of MI computations, which grows very quickly for large N_X. However, it is possible to use a greedy algorithm for our problem at hand, and using the submodularity property, theoretical bounds can be proven about the optimality of greedy search answer w.r.t. to the global maximum. The result for the bound on optimality of greedy search for a submodular function was first proven by Nemhauser et al., 1978 [144], while it was first used in the context of MI in Krause et al., 2008 [143]. Lin, 2020 [9] provides an in-depth analysis of submodularity of MI for data assimilation problems like ours. The goal of the work in this section is to perform greedy sub-modular maximization to find the set of best locations to observer nutrients so as to simultaneously learn all the uncertain biological states, Ivlev grazing parameter, and the presence or absence of the zooplankton mortality term. The initial setup of the experiment is the same as that described in section 2.4.1, the system is evolved till time $T = 5$. The goal is to make nutrient observations at four locations and find the set of best locations to learn the states, parameters and the model at time $T = 5$ itself. The target RVs or often called the verification variables, consist of variables $\boldsymbol{V}(t;\omega) = [a(\omega), \Lambda(\omega), \boldsymbol{N}(t;\omega), \boldsymbol{P}(t;\omega), \boldsymbol{Z}(t;\omega)]$ at $t = 5$. The greedy submodular maximization is a simple algorithm which will start with an empty set of observation locations and computes the MI between $\boldsymbol{V}(t;\omega)$ and $N(\boldsymbol{x}, t;\omega)$ at every grid location which gives us an MI field. The location of the grid point with highest MI value is added to the set of observation locations. We recompute the MI field by computing MI between $\boldsymbol{V}(t;\omega)$ and $N(\boldsymbol{x}, t;\omega)$ conditioned on the already selected observations, at every grid location apart from the locations in the existing observation set. We repeat this process until we find the required number of observations. We compare the set of four observations locations found by the greedy algorithm, with those randomly picked throughout the domain. We compute their combined MI content for the verification variable, and we also perform one step of data assimilation using the GMM-DO filter and compute the normalized root-mean-

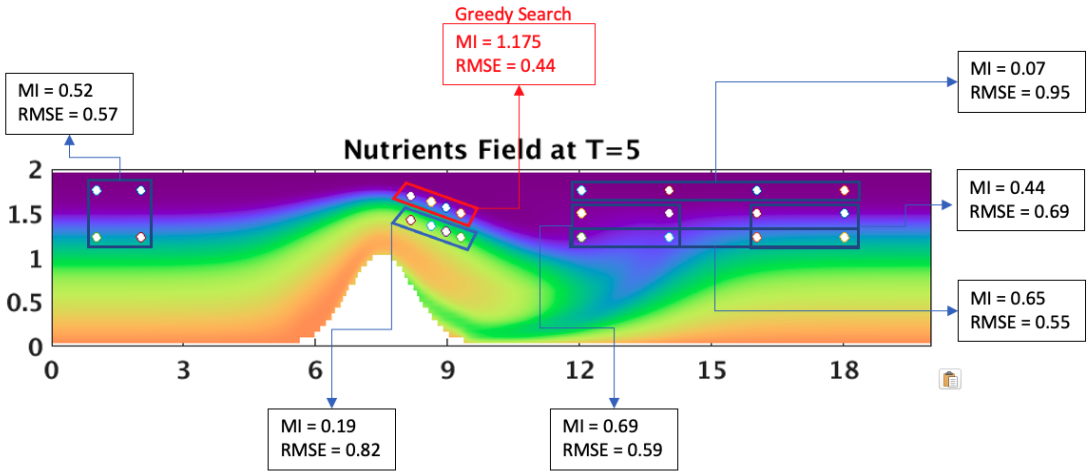


Figure 5-19: The background consists of the true nutrient field at $T = 5$ from which observations are extracted. Overlayed are different sets of four observation locations, and their mutual information content, and normalized posterior RMSE if they were assimilated. The *red box* and *arrow* marks the set of locations found using the greedy submodular maximization.

square-error (RMSE) of the combined posterior biological states, $\Lambda(\omega)$, and $a(\omega)$. The results are provided in the figure 5-19, and we can easily notice that the set of four locations found using the greedy submodular maximization has a significantly higher MI and leads to the smallest RMSE for the posterior as compare to the other randomly picked sets of observation locations. It is also interesting to note that the location of the observation set found by the greedy search lies inside the phytoplankton bloom, and the location and extent of the phytoplankton bloom is what is most sensitive to the value of the Ivlev parameter, and zooplankton mortality. Picking observations even just slightly below contains significantly lowers the amount of information.

5.4.3 Identifiability

Especially in ecosystem models, due to multidisciplinary dynamics, it is the case that the available data is not informative enough for some of the learning objectives, and is known as the problem of identifiability in the ocean ecosystem modeling scientific community [145]. Apart from selecting the observation locations, one first needs to

decide the state variable to be measured. As the instruments needed to measure each of the biological or physical variables are drastically different and expensive, it is not practical to carry all of them when embarking on a scientific expedition in the ocean to make *in-situ* observations. Thus, it becomes important to make a scientifically-sound decision about the instruments to carry. We propose to leverage MI for selecting the candidate variables which will definitely be informative about the learning objectives we really care about. Let us consider the NPZ system with Ivlev grazing parameter ($\Lambda(\omega)$) and phytoplankton mortality rate parameter ($\Xi(\omega)$) uncertainties. We initialize and evolve our system in the same fashion as done in earlier experiments (section 2.3). The goal is to quantify the identifiability of the $\Lambda(\omega)$ and $\Xi(\omega)$ parameters from phytoplankton observations. We compute MI between phytoplankton state at every grid location, $P(\boldsymbol{x}, t; \omega)$, and either of these uncertain parameters individually. The MI fields are provided in figure 5-20, and we can notice that the phytoplankton barely contains any information about $\Xi(\omega)$ anywhere in the domain, and thus, its observations will be ineffective in reducing its uncertainty. Once we have identified the candidate variables which would be the most effective for our learning objectives, we can utilize the greedy submodular maximization (section 5.4.2) to decide the locations at which observations should be made.

5.4.4 Predictability

The lack of ability to represent all the processes and interactions in the real-world leads to the need to introduce various sources of uncertainties in our dynamical system based models, followed by their accurate quantification and evolution. A system is called to have lost its predictability when the errors have grown so much that the forecast offers no better prediction than a randomly chosen field for the system [146]. Alternatively, it could be interpreted as the amount of time taken for the information to decay in a system. The knowledge of predictability could help us guide the temporal frequency of the of the observations needed, which could help with determining the operational needs and constraints of the observation experiments.

There are numerous ways in which scientists try to measure and quantify the pre-

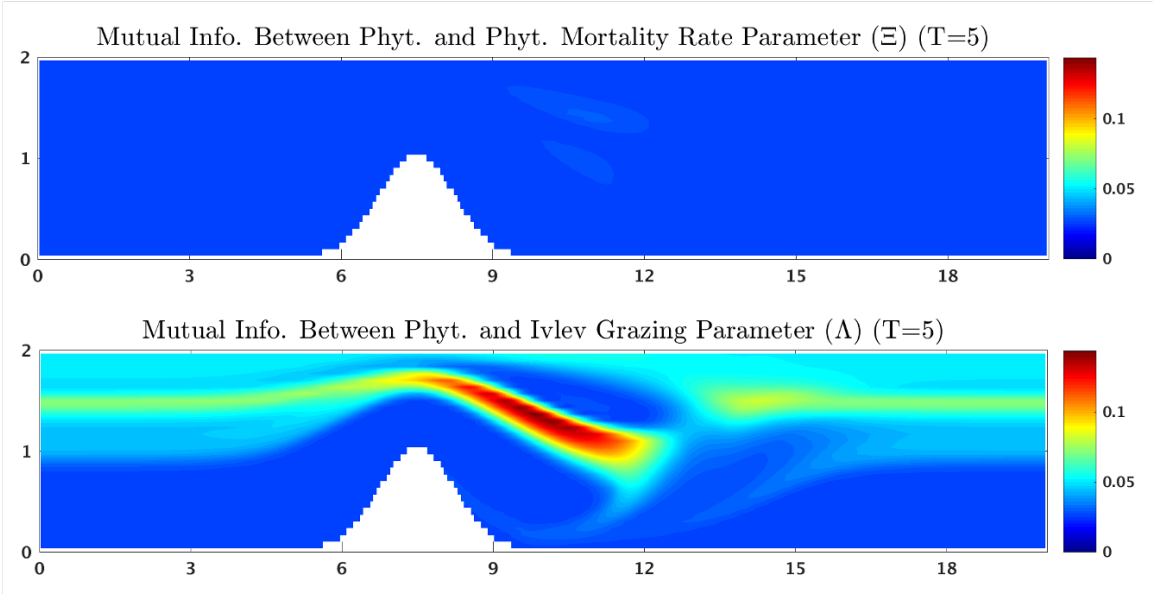


Figure 5-20: Mutual information fields consisting of mutual information computed between phytoplankton at each grid point and phytoplankton mortality rate parameter ($\Xi(\omega)$) in the *top*, and between phytoplankton at each grid point and Ivlev grazing parameter ($\Lambda(\omega)$) in the *bottom*, at time $t = 5$.

dictability limits. For example, using the evolution of mean-squared error, however, this requires the knowledge of the truth for comparison; the decay of auto-correlation, which is more relevant to deterministic systems; or using the Mahalanobis distance [147]. The use of MI as a measure for quantifying predictability of a system with knowledge of only the initial conditions and fixed boundary conditions, also called predictability of the first kind, was first proposed long ago by Leung and North, 1989 [148]. However, it is discarded especially in case of high-dimensional multidisciplinary systems due to the computational challenges associated with computing MI [147].

To demonstrate the use of MI for estimating the predictability limit for a high-dimensional multidisciplinary systems, we utilize the experimental setup of NPZ system with Ivlev grazing parameter, and quadratic zooplankton mortality uncertainty (section 2.4.1). Along with this, we also consider the Reynolds number to be uncertain along with the initial velocity as described in section 3.4.1. No data is assimilated in the current experiment. MI is computed between augmented state variables and the uncertain parameters at the initial time ($T = 0$) with those at later times. The

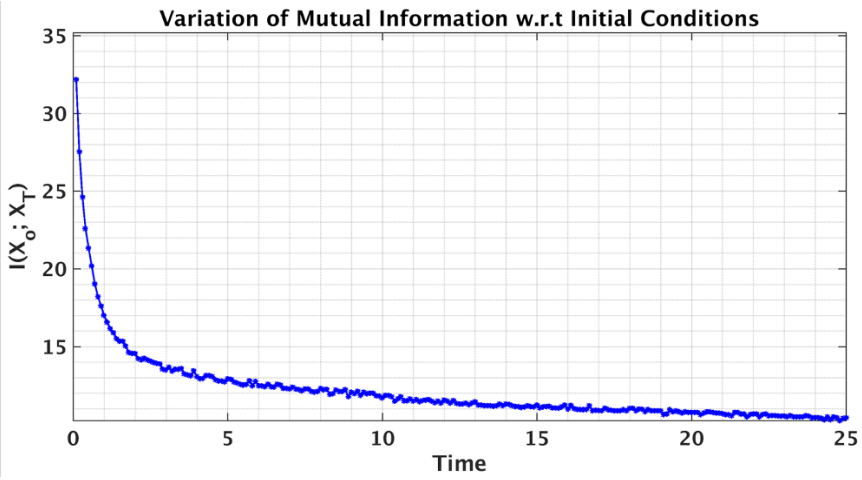


Figure 5-21: Variation of mutual information computed between augmented states and parameters at initial time ($t = 0$), and at later times. See section 5.4.4 for more details.

variation of MI with time is presented in figure 5-21. We notice an exponential decay in the amount of information retained by the system about the initial conditions at the later time, with a decay scale of the order of 2-3 non-dimensional times. This corroborates with our personal experience while setting up the corresponding learning experiment. In order to successfully learn the system, the observational episode had to be started relatively earlier than experiments with uncertainty only in the biogeochemical part, at $t = 1$ vs. $t = 5$, and subsequent observations were assimilated after every 2 non-dimensional times. The predictability limit is in general sensitive to the initial uncertainty, numerical error, and in our case would also be dependent on the error introduced due to truncating the DO modes and evolving the uncertainty in an reduced space. A detailed sensitivity study is required. It is however beyond the scope of the current thesis.

5.4.5 Applications to Realistic Ocean Simulations

In this section, we briefly mention two collaborative research projects involving realistic ocean simulations, for which MI-based analyses were performed to identify optimal observation locations.

Northern Arabian Sea Circulation autonomous research (NASCar) program[2]: This US Office of Naval Research (ONR) funded program employs a variety of autonomous and Lagrangian platforms and sensor systems to investigate the dynamics of the northern Arabian Sea. One of the goals of our group's NASCar contribution was to apply our theory and schemes for optimal path planning and optimal ocean sampling with swarms of autonomous vehicles. In what comes next, the finite-time Lyapunov exponent (FTLE) computations were performed by Dr. Chinmay S. Kulkarni and Mr. Arkopal Dutt, and the text is copied verbatim from Lermusiaux *et al.*, 2017 [1]. In Figure 5-22, we illustrate how MI fields forecasts can be used to identify the locations for observing different types of data that would be most informative about the velocity field or Lagrangian coherent structures. A particular realization of the ensemble forecast of the forward-time FTLE field over an interval of three days (March 27 to March 30, 2017) is shown (Figure 5-22a). The ridges of the forward-time FTLE field correspond to repelling Lagrangian coherent structures (i.e., material lines from which parcel trajectories separate the most). The white box marks the region for which the zoomed-in FTLE field is shown on the right. Winds and upper-ocean dynamics lead to rapid variability in the surface velocity field (not shown). When the velocity field is rapidly varying, the features in the FTLE field cannot be identified with the velocity field at one particular time instance. In Figure 5-22c,d we show forecast MI fields between candidate observations of salinity anywhere in the small domain (white box in Figure 5-22a), and the verification variable, which is a field defined over the whole small domain. The MI field between salinity and the scalar field of zonal velocity over the small domain indicates that the most informative salinity data locations are around $12.6^{\circ}N$, $58.2^{\circ}E$. The MI field between salinity and the FTLE field over the small domain (from which coherent structures can be estimated) indicates the most informative salinity data locations are around $12.5^{\circ}N$, $58.7^{\circ}E$. The informative locations in this field lie on the edge of a high-salinity intrusion (not shown here). We also note the differences in the locations of the most informative data in the two fields, confirming that observation

[2] `http://mseas.mit.edu/Sea_exercises/NASCar-OPS-17/`

data locations that are highly informative for one verification variable may not be so for another. In Figure 5-22e-g we show forecast MI fields between velocity and the forward-time FTLE field. The MI field between zonal velocity and forward-time FTLE field indicates that the most informative locations are around $12.5°N$, $58.8°E$. In contrast, when meridional velocity is measured, the most informative data locations lie near $11.6°N$, $59.2°E$. Also, there are more candidate observation locations that are highly informative about coherent structures when measuring either velocity component than when measuring salinity. Furthermore, if we were to observe both zonal and meridional velocity, the MI about the coherent structures is logically higher than when we measure only one of the components. This full velocity MI is, however, maximized when the observation locations are around $12.3°N$, $58.8°E$ and $11.7°N$, $59.6°E$.

Coherent Lagrangian Pathways from the Surface Ocean to Interior (CALYPSO)[3]: As a part of this ONR funded research project, we used MI to predict where and when to deploy drifters so as to best identify and explore subduction regions in the Alboran Sea from March 27 to April 11, 2019. For example, we answered where and which state variable to measure at the surface to maximize information about the temperature of a parcel starting at the location $35.8°N$, $3.128°W$, and $4m$ depth on April 8, 2019 12Z that is predicted to subduct by more than $100m$. In order to do this, we computed the MI between different state variables such as temperature, salinity, zonal and meridional velocities at $0m$ in the whole domain on April 8, 2019 12Z, with the temperature at the starting location of the parcel. Similarly, we also computed MI of these different state variables at the surface on April 8, 2019 12Z with the parcel's final location ($35.418°N$, $1.98°W$, and $106m$ depth) on April 12, 2019 12Z. Areas of high MI provide the candidate observation locations that would best inform our learning goals, as shown in figure 5-23.

[3] `http://mseas.mit.edu/Sea_exercises/CALYPSO/2019/`

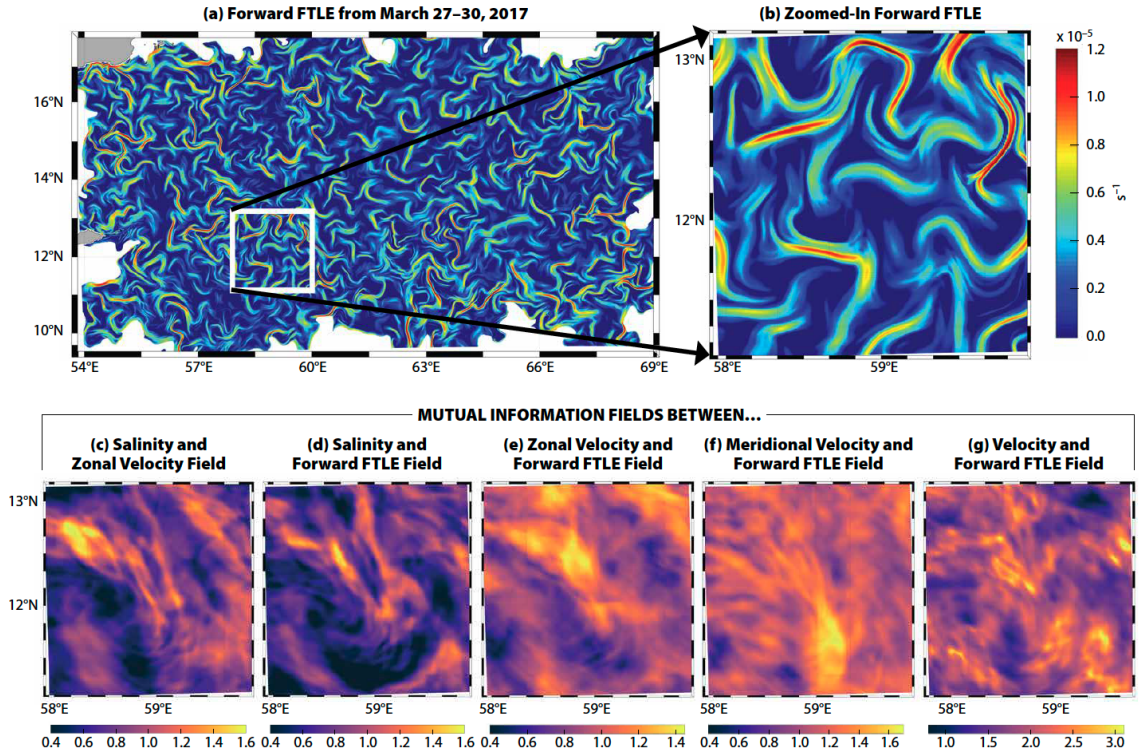


Figure 5-22: Adaptive sampling predictions for velocity or coherent structure fields. (a-b) Forecast realization of the forward-time finite-time Lyapunov exponent (FTLE) field (a) and of the same FTLE field but zoomed in a small domain (b), marked by the white box in (a). (c-g) Forecast mutual information fields within this small domain, between the observation variable at any location in the domain and the verification variable which is here always a field defined over that small domain. The five mutual information fields forecasts are between each of the following pairs of observation and verification variables: (c) salinity and zonal velocity field, (d) salinity and forward-time FTLE field, (e) zonal velocity and forward-time FTLE field, (f) meridional velocity and forward-time FTLE field, and (g) velocity (both components) and forward-time FTLE field. These mutual information fields forecast the most informative observation locations for estimating the verification variable over the small domain. Note that the color bars of panels (c-g) differ. This figure and caption exactly appeared in Lermusiaux *et al.*, 2017 [1].

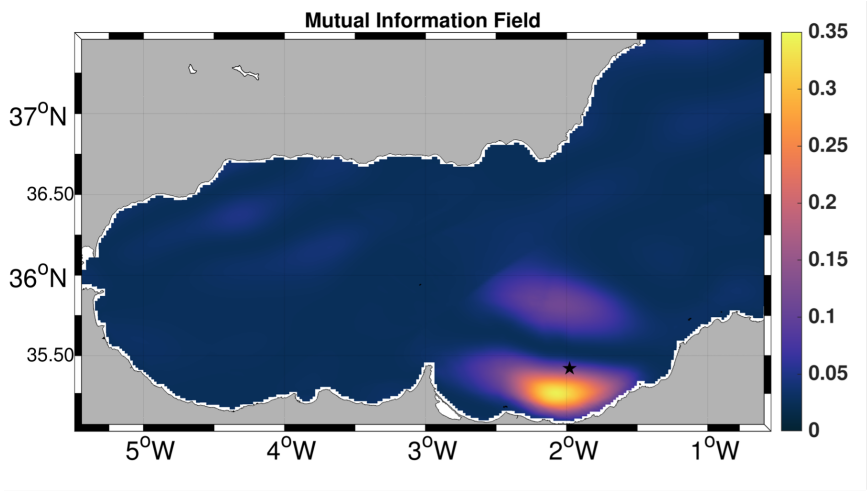


Figure 5-23: Mutual information between zonal velocity on the surface on April 8, 2019 12Z, with the temperature at the location $1.98°W$, $35.418°N$ and $106m$ depth on April 12, 2019 12Z.

5.5 Summary

In this chapter, we first derived the methodology to ensure realization-wise matching between a controlled approximation of the interior and boundary realizations respectively (strong imposition of boundary conditions), by splitting dynamically orthogonal (DO) modes into unforced and forced, and re-deriving the DO equations. Using an experiment based on the flow past a cylinder with stochastic initial and inlet Dirichlet boundary conditions, we compared the performance of strong vs. weak (interior satisfying the boundaries only up to 2^{nd} moment) imposition of stochastic boundary conditions. Next, we discussed the implementation of ghost cell immersed boundary method (GCIBM) to handle complex and smooth obstacle / boundary geometries accurately in the 2D finite-volume (FV) framework used in the experiments throughout this thesis. A deterministic Couette flow experiment was utilized to test the implementation of the GCIBM method. We also compare the performance and advantages of using linear vs. non-linear advection schemes with DO methodology in the case of uncertain flow field. We find central-difference scheme for advection and Heun's method for time-integration, along with applying the Shapiro filter to be able to eliminate the presence of spurious oscillations in the stochastic simula-

tions. To deal with cases where the size of the initial subspace becomes insufficient to capture the uncertainty evolution, we implemented the methodology for data-driven subspace augmentation and covariance inflation developed in Lin, 2020 [9]. The implemented method utilizes the discarded information contained in the projection of the observations lying outside the current subspace to make rank-1 updates. Using an experiment based on the three-component NPZ system with uncertainty in the initial conditions of the biological states, Ivlev grazing parameter, and the functional form of the zooplankton mortality term, we demonstrated an improvement in learning in the presence of data-driven subspace augmentation and covariance inflation. Finally, we showcased how mutual information can be efficiently computed and utilized for observation planning for high-dimensional multidisciplinary systems. We were able to derive scientifically sound knowledge to help make decisions about *what* (identifiability), *when* (predictability), and *where* (optimal locations) to observe in both idealized and realistic ocean simulations.

Chapter 6

Neural Closure Models for Dynamical Systems[1]

Most models only resolve spatio-temporal scales, processes, and field variables to a certain level of accuracy because of the high computational costs associated with high-fidelity simulations. Such truncation of scales, processes, or variables often limit the reliability and usefulness of simulations, especially for scientific, engineering, and societal applications where longer-term model predictions are needed to guide decisions. There are many ways to truncate high-fidelity models to low-fidelity models. Examples abound and three main classes of truncations are: evolving the original dynamical system in a reduced space, e.g., using reduced-order-models (ROMs) [149, 150]); coarsening the model resolution to the scales of interest [151, 152]; and reducing the complexity or number of state variables, components, and parameterizations [153, 154, 155]. In many applications, the neglected and unresolved terms along with their interactions with the resolved ones can become important over time, and a variety of modeling techniques have been developed to represent the missing terms. Techniques that express these missing terms as functions of modeled state variables and parameters are referred to as closure models. A main challenge is that no one closure approach to date is directly applicable to all four main classes of model

[1]This chapter is published as: Gupta, A., & Lermusiaux, P. F. (2021). Neural closure models for dynamical systems. *Proceedings of the Royal Society A*, 477(2252), 20201004.

truncations. Another is that closure models are only well-defined for either linear problems or simple cases. Finally, they can easily become ineffective in the face of nonlinearities.

Due to the explosion of use of a variety of machine learning methods for solving or simulating dynamical systems, a number of data-driven methods have been proposed for the closure problem. Most of them attempt to learn a neural network as the instantaneous map between the low-fidelity solution and the residual of the high- and low-fidelity solution, or their residual dynamics [156, 157, 2, 158, 159]. They often use recurrent networks such as long-short term memory networks (LSTMs), gated recurrent units (GRUs) etc., with justification based on the Mori-Zwanzig formulation [160, 161, 162] and embedding theorems by Whitney [163] and Takens [164]. These approaches do not however take into account accumulation of numerical time-stepping error in the presence of neural-networks during training. and uniformly-spaced high-fidelity data to be able to compute the time derivative of the state with high level of accuracy. Such requirement on the training data can be a luxury in a lot of scenarios. The requirement of very frequent snapshot data of the system is also true for methods which achieve model discovery using sparse-regression and provide interpretable learned models [24, 27, 159]. All of the above issues are addressed by using neural ordinary differential equations (nODEs; [33]) and some researchers recently used nODEs for closure modeling. Some directly learn the ODE system from high-fidelity simulation data without using the available low-fidelity models [165], which could lead to the requirement of bigger neural networks. Others combine nODEs with model discovery using sparse-regression [166] or only learn the values of parameters in existing closure models [167]. Nearly all existing studies primarily only attempt to address the closure for ROMs. Finally, the existing machine learned closure models are not yet used for long-term predictions, i.e. forecasting significantly outside of the time-period to which the training data belonged to.

In the present study, we propose a new neural delay differential equations (nDDEs) based framework to learn closure parameterizations for low-fidelity models using data from high-fidelity simulations and to increase the long-term predictive capabilities of

these models. Instead of using ODEs, we learn non-Markovian closure models using DDEs. We base the theoretical justification for using DDEs on the Mori-Zwanzig formulation [160, 161, 162] and the presence of inherent delays in many dynamical systems [168], especially biological systems [169, 170]. We refer to the new modeling approach as *neural closure models*. We demonstrate that our methodology drastically improves the predictive capability of low-fidelity models for the main classes of model truncations. Specifically, our neural closure models efficiently account for truncated modes in ROMs, capture the effects of subgrid-scale processes in coarse models, and augment the simplification of complex mathematical models. We also provide adjoint equation derivations and network architectures needed to efficiently implement nDDEs, for both discrete and distributed delays. In the case of distributed delays, we propose a novel architecture consisting of two coupled neural networks, which eliminates the need for using recurrent architectures for incorporating memory. We find that our nDDE closures substantially improve nODE closures and outperform classic dynamic closures such as the Smagorinsky subgrid-scale model. We explain the better performance of nDDE closures based on information theory and the amount of past-information being included. Our first two classes of simulation experiments utilize the advecting shock problem governed by the Burger's partial differential equation (PDE), with low-fidelity models derived either by proper-orthogonal-decomposition Galerkin projection [171] or by reducing the spatial grid resolution. Our third class of experiments considers marine biological models of varying complexities [3, 39, 172] and then their physical-biogeochemical extensions, with low-fidelity models obtained by aggregation of components and other simplifications of processes and parameterizations. Finally, we analyze computational complexity and explain the limited additional computational cost due to the presence of neural closure models.

6.1 Closure Problems

The need for closure modeling in dynamical systems arises for a variety of reasons. They often involve computational costs considerations, but also include the lack of

data to resolve complex real processes, the limited understanding of fundamental dynamics, and the inherent nonlinear growth of uncertainties due to model errors and predictability limits [e.g. 173, 174, 175, 176]. In this section, we examine three main classes of low-fidelity models that can require closure modeling.

6.1.1 Reduced Order Modeling

Let us consider a nonlinear dynamical system with state variable $u \in \mathbb{R}^N$ and the full-order-model (FOM) dynamics governed by,

$$\frac{du(t)}{dt} = Lu(t) + h(u(t)), \quad \text{with} \quad u(0) = u_0\,, \tag{6.1}$$

where $L \in \mathbb{R}^{N\times N}$ is the linear, and $h(\cdot) : \mathbb{R}^N \to \mathbb{R}^N$ the nonlinear, part of the system. We are mainly interested in dynamical systems whose solution could be effectively approximated on a manifold of lower dimension, $\mathcal{V} \in \mathbb{R}^m \subset \mathbb{R}^N$ [e.g. 13]. Ideally, the dimension of this manifold is much smaller than that of the system, i.e. $m << N$. For the classic Galerkin-based reduced-order modeling, a linear decomposition of the form,

$$u(t) \approx \bar{u} + Va \tag{6.2}$$

is used, where $\bar{u} \in \mathbb{R}^N$ is a reference value, the columns of $V = [v_1, ..., v_m] \in \mathbb{R}^{N\times m}$ a basis of the m-dimensional subspace $\mathcal{V}$, and $a \in \mathbb{R}^m$ the vector of coefficients corresponding to the reduced basis. A popular choice for this basis is the proper-orthogonal-decomposition (POD) that defines the subspace such that the manifold $\mathcal{V}$ preserves the variance of the system as much as possible when projected on V for a given m. The reference value ($\bar{u}$) is then commonly chosen as the mean of the system state, in order to prevent the first reduced coefficient from containing the majority of the energy of the system and to help stabilize the reduced system [177].

Now, substituting Eq. 6.2 into Eq. 6.1, and projecting the result on the orthonormal modes V, we obtain the following set of ordinary differential equations for the

coefficients a,

$$\frac{da}{dt} = V^T L V a + V^T h(\bar{u} + Va) + V^T L \bar{u}\,, \quad \text{with} \quad a(0) = V^T(u_0 - \bar{u})\,. \tag{6.3}$$

This m dimensional system, with $m << N$, is computationally much cheaper than the original FOM Eq. 6.2. This method of dimensionality reduction is commonly referred to as the POD Galerkin Projection (POD-GP) method. It can suffer from a number of issues. First, the truncated modes can play an important role in the dynamical behaviour of the system, and neglecting them can thus lead to a very different forecast [150]. Second, the error in the reduced state may be simply too large for truncation, i.e. the POD reduction is not efficient. Third, if steady POD are employed, they may quickly become irrelevant for the evolving system state [12, 13, 14]. To address these issues, several methods try to represent the effect of the truncated modes. The most common approaches introduce a nonlinear parameterization of the coefficients [e.g. 178] in Eq. 6.3, however, they are not yet generally applicable to all classes of closures.

The geometric interpretation of the goal of closure modeling for ROMs is sketched in Fig. 6-1. The FOM solution of our dynamical system lies outside the lower dimension manifold, $\mathcal{V}$. A ROM approximate solution, denoted by u^{ROM}, starts with the projection of the full-order initial condition onto the manifold, $VV^T u(0)$, but quickly diverges from the actual projection of the full-order solution onto the manifold ($VV^T u$), often leading to a significant source of error. A closure model in this case basically attempts to keep the updated "closed" solution, u^{ROM+C}, as close as possible to the actual projection of the FOM solution ($VV^T u$) which can be seen as the truth.

6.1.2 Subgrid-Scale Processes

A key decision while setting up any numerical simulation is the selection of spatio-temporal resolution, which is in general limited by the computing power available. Using a coarse resolution (low-fidelity) model may however lead to a number of undesired artifacts, such as missing critical scales and processes for longer-term predictions or numerical diffusion that causes unintended or unacceptable results [179, 180].

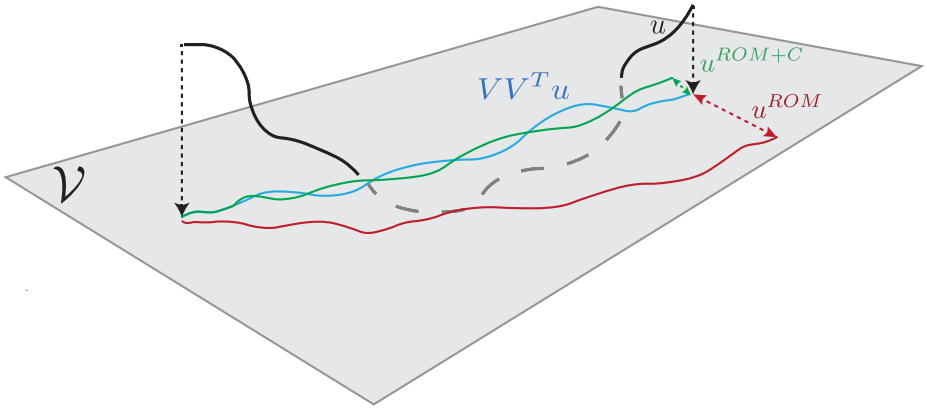


Figure 6-1: Geometric interpretation of the closure for reduced-order-models (ROMs). u (—): Solution to the full-order-model (FOM); VV^Tu (—): Projection of u on the subspace V; u^{ROM} (—): Solution to the proper-orthogonal-decomposition Galerkin-projection (POD-GP) ROM; and, u^{ROM+C} (—): Solution to POD-GP ROM with closure. Adapted from [2].

These artifacts become especially important in the case of ocean models. For example, present-day global observing systems and global model solutions only resolve open-ocean mesoscale processes ($\mathcal{O}(10-100km)$), but the submesocale (subgrid-scale) processes do have global consequences, in relation to the mechanisms of energy dissipation in the general circulation, vertical flux of material concentrations, and intermediate-scale horizontal dispersal of materials [181, 182, 183]. The neglected and unresolved scales along with their interactions with the resolved ones are then at the core of closure parameterizations. Most present oceanic models consist of a nonlinear system of PDEs, each of a nonlinear advection type, supplemented by other possible diagnostic nonlinear equations, and boundary conditions. There is however no unique way of defining such parameterizations and multiple approaches such as non-dimensional analyses, physical balance hypotheses, statistical correlation constraints, and other empirical methods are commonly employed to develop closure models. Similar statements can be made for atmospheric, Earth system, and climate models [184]. For all these applications, a general approach for subgrid-scale closures would thus be most useful.

6.1.3 Simplification of Complex Dynamical Systems

Due to incomplete understanding and limited measurements, it is common when modeling real dynamical systems in nature and engineering that the dynamics cannot be accurately explained just by using conservation laws and fundamental process equations. We refer to such systems as complex dynamical systems. The number of candidate models and equations can then be almost as large as the number of modelers. The resulting models also vary greatly in terms of their complexity. More complex models can capture key processes and feedbacks. Complexity is increased by adding more parameters and parameterizations to the existing components (state variables) of the dynamical model, but at some point, it quickly becomes inevitable to add and model new components to capture the underlying real processes accurately, hence further increasing model complexity [185, 39, 186]. This is common, for example, in marine ecosystem models, where simpler models only resolve the broad biogeochemical classes, while more complex models capture detailed sub-classes [155, 153]. Increasing the number of components however can come at great computational cost, can increase the overall uncertainty, and can lead to loss of accuracy or stability due to the nonlinearities. Also, the unknown parameters for models with more components are calibrated from available data and the optimization process and parameter estimation quickly become challenging with the increase in complexity, due to the simultaneous explosion in the number of unknown parameters [187]. Thus, instead of adding more unknown parameterizations or increasing the number of components, one might use adaptive models [188, 61] and in general the present neural closure models with time delays to incorporate the effects of missing processes in low-complexity models, enabling them to adapt and emulate the response from high-complexity models.

6.2 Theory and Methodology

In this section, we develop the theory and methodology for learning data-assisted closure models for dynamical systems. We first review the Mori-Zwanzig (MZ) formula-

tion [160, 161, 162] which derives the exact functional form of the effects of truncated dynamics for common reduced models. Unfortunately, apart for very simple linear dynamical systems, the use of this formulation is challenging without making unjustified approximations and simplifications. We then discuss the presence of delays in complex dynamical systems, and their impact on modeling [169]. Motivated by both the MZ formulation and the presence of delays, we finally derive the new nDDEs and neural closure models, including adjoint equations and network architectures, for both discrete and distributed delays.

6.2.1 Mori-Zwanzig Formulation and Delays in Complex Dynamical Systems

Without loss of generality, the full nonlinear dynamical system model is written as,

$$\frac{du_k(t)}{dt} = R_k(u(t), t)\,, \quad \text{with} \quad u_k(0) = u_{0k}\,, \quad k \in \mathfrak{F}\,. \tag{6.4}$$

The full state vector is $u = (\{u_k\}),\ k \in \mathfrak{F} = \mathfrak{R} \cup \mathfrak{U}$, where $\mathfrak{R}$ is the set corresponding to the resolved variables (e.g. coarse field or reduced variables), and $\mathfrak{U}$ the set corresponding to the unresolved variables (e.g. subgrid field or complement variables), which as a union, $\mathfrak{F}$, form the set for full space of variables. We also denote $u = \{\hat{u}, \tilde{u}\}$ where $\hat{u} = (\{u_k\}),\ k \in \mathfrak{R}$ and $\tilde{u} = (\{u_k\}),\ k \in \mathfrak{U}$. Similarly, $u_0 = \{\hat{u}_0, \tilde{u}_0\}$, with $\hat{u}_0 = (\{u_{0k}\}),\ \in \mathfrak{R}$ and $\tilde{u}_0 = (\{u_{0k}\}),\ k \in \mathfrak{U}$.

The Mori-Zwanzig (MZ) formulation allows rewriting the above nonlinear system of ODEs as,

$$\frac{\partial}{\partial t}u_k(u_0, t) = \underbrace{R_k(\hat{u}(u_0, t))}_{\text{Markovian}} + \underbrace{F_k(u_0, t)}_{\text{Noise}} + \underbrace{\int_0^t K_k(\hat{u}(u_0, t-s), s)ds}_{\text{Memory}}, \quad k \in \mathfrak{R}\,, \tag{6.5}$$

where R_k is as in the full model dynamics (Eq. D.1). Importantly, the above equation is an exact representation of Eq. D.1 for the resolved components. A derivation is provided in the *Supplementary Information* (Sec. D.1). Eq. D.5 provides useful

guidance for closure modeling. The first term in Eq. D.5 is the Markovian term dependent only on the values of the variables at the present time, while the closure consists of two terms: the noise term and a memory term that is non-Markovian. We can further simplify Eq. D.5 by applying the P projection and using the fact that the noise term lives in the null space of P for all times, which could be easily proved. For ROMs with initial conditions devoid of any unresolved dynamics, i.e. $\tilde{u}_0 = 0$ and thus $u_0 = \hat{u}_0$, we then retain the exact dynamics after the projection step, noticing in this case that $Pu_k(u_0, t) = u_k(\hat{u}_0, t), \forall k \in \mathfrak{R}$,

$$\frac{\partial}{\partial t} u_k(\hat{u}_0, t) = PR_k(\hat{u}(\hat{u}_0, t)) + P \int_0^t K_k(\hat{u}(\hat{u}_0, t-s), s) ds, \quad k \in \mathfrak{R}\,. \tag{6.6}$$

Hence, for such systems, the closure model would only consider the non-Markovian memory term. The above derivation of the MZ formulation has been adapted from [160, 2, 162].

The MZ formulation clearly shows that a non-Markovian closure term requiring time-lagged state information is theoretically needed to model the unresolved or missing dynamics. This theoretical basis directly applies to the first two classes of low-fidelity dynamical systems (Sec. 6.1), ROMs and coarse resolution models. For the third category, the simplification of complex dynamical systems, we emphasize biological and chemical systems. Many are modeled using ODEs, with one state variable per biological or chemical component. Such ODEs implicitly assume that information between state variables is exchanged instantaneously. In reality, however, there are often time-delays for several reasons. First, changes in populations or reactions have non-negligible time-scales [e.g. 189, 190, 168]. Such time-scales are introduced in more complex models by modeling intermediate state variables. Hence, the time response of lower-complexity models can be comparable to that of high-complexity models only by explicitly introducing delays [189, 191, 169, 170]. Second, many reactive systems are modeled assuming smooth concentration fields of state variables governed by PDEs with fluid flow advection and/or mixing, leading to advection-diffusion-reaction PDEs [192, 193]. In that case, simplified models still

require time-delays due to the neglected reactive or biogeochemical dynamics but now also due to truncated modes and/or subgrid-scale processes of numerical models. For all of these reasons, the need for memory based closure terms is clearly justified to represent complex dynamical systems.

There are some results for data-assisted / data-driven closure modeling based on the MZ formulation. Some schemes create a coupled system of stochastic differential equation using appropriate hidden-variables for approximate Markovization of the non-Markovian term [194, 195]. Others use a variational approach to derive nonlinear parameterizations approximating the Markovian term [196]. Schemes using machine learning to learn non-Markovian residual of the high- and low-fidelity dynamics limit themselves to specific functional forms for the residual term, simple Euler time-stepping scheme, and very frequent and uniformly-spaced training data [159, 2, 157]. They also lack the rigorous use of the theory for time-delay systems [197].

6.2.2 Neural Delay Differential Equations

The non-Markovian closure terms with time-lagged state information lead us to delay differential equations (DDEs) [198]. DDEs have been widely used in many fields such as biology [199, 200], pharmacokinetic-pharmacodynamics [201], chemistry [202], economics [203], transportation [204], control theory [205], climate dynamics [206, 207], etc. Next, we summarize the state of the art for learning and solving differential equations using neural-networks (NNs) and develop theory and schemes for neural DDEs including adjoint equations for backpropagation.

The interpretation of residual networks as time integration schemes and flow maps for dynamical systems has led to pioneering development of neural ordinary differential equations (nODEs) [33]. A nODE parameterizes an ODE using a neural-network and solves the initial value problem (IVP) given by,

$$\frac{du(t)}{dt} = f_{NN}(u(t), t; \theta)\,, \quad t \in (0, T]\,, \quad \text{with} \quad u(0) = u_o\,, \tag{6.7}$$

where f_{NN} is the prescribed neural-network and θ are the weights. Starting from the initial conditions, the nODE (Eq. 6.7) is integrated forward in time using any time-integration scheme, and then gradients are computed based on a loss function using the adjoint sensitivity method. The gradient computation boils down to solving a second ODE backwards in time. Using standard backpropagation for Eq. 6.7 has however several issues: it would be very memory expensive as one needs to store the state at every time step; its computational cost would increase when using higher-order time-integration or *implicit* schemes; and, it might become infeasible if the forward time-integration code does not support automatic differentiation. The adjoint method, however, provides a backpropagation for nODEs [33] that is memory efficient and flexible as it treats the time-integration scheme as a "black-box". In our case, we need to incorporate state-delays. Though extending the nODE framework to incorporate DDEs comes under the ambit of universal differential equations (UDEs) [34], deeper investigations are warranted. First, the UDEs are presently implemented using the Julia library DiffEqFlux.jl [208] which can perform automatic adjoint equation solves, but other popular open-source languages such as Python and R, and ML-Frameworks such as TensorFlow [209], PyTorch [210], etc., would require explicit derivation and coding of the corresponding adjoint equations. Second, we need to study two different types of DDEs, the discrete and distributed delays, which it turns out require different architectures. Next, we thus develop the theory and schemes for efficient implementation of neural delay differential equations (nDDEs) in any programming language.

Discrete Delays

The most popular form of delay differential equations (DDEs) is,

$$\begin{aligned} &\frac{du(t)}{dt} = f(u(t), u(t-\tau_1), ..., u(t-\tau_K), t), \quad t \in (0, T]\,, \\ &\text{with} \quad u(t) = h(t), \quad t \leq 0\,, \end{aligned} \tag{6.8}$$

where $\tau_1, ..., \tau_K$ are K number of discrete-delays (discrete DDEs). Instead of a single initial value as in the case of ODEs, DDEs require specification of a history function, $h(t)$. Due to the presence of a given fixed number delays, we can parameterize the above system by replacing the time-derivative function with potentially any type of NNs. For example, to use fully-connected NNs we would concatenate all the delayed states vertically to form the input vector, or concatenate them horizontally to form an input matrix for a convolutional NN. However, recurrent NN (RNN) architectures, such as simple-RNNs, LSTMs, GRUs, etc., are ideal and most efficient for our need due to the time-series nature of the delayed states. We can assume that the discrete delays are evenly spaced (this is not a hard requirement as we can easily extend schemes to irregularly spaced discrete-delays using ODE-RNNs [211], but for brevity we make this assumption) and use a RNN with weights θ. Hence, our new discrete-DDE system can be written as,

$$\begin{aligned} &\frac{du(t)}{dt} = f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta), \quad t \in (0, T]\,, \\ &\text{with } \; u(t) = h(t), \quad t \leq 0\,, \end{aligned} \tag{6.9}$$

where $f_{RNN}(\bullet; \theta)$ is the recurrent architecture. We refer to this parameterization of discrete DDEs as *discrete-nDDE*. The graphical representation of Eq. 6.9 in time-discretized form is depicted in Figure 6-2a. Let data be available at M times, $T_1 < ... < T_M \leq T$. We then optimize the total loss function given by, $\mathcal{L} = \int_0^T \sum_{i=1}^M l(u(t))\delta(t - T_i)dt$, where $l(\bullet)$ are scalar loss functions such as mean-squared-error (MSE), and $\delta(t)$ is the Kronecker delta function. To perform this optimization with any gradient descent algorithm, we need the gradient of the loss function w.r.t. the weights of the RNN, θ. Using the adjoint sensitivity method [212] to compute the required gradients, we start by writing the Lagrangian for the above system,

$$\begin{aligned} L =& \mathcal{L}(u(t)) + \int_0^T \lambda^T(t) \left(d_t u(t) - f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta)\right) dt \\ &+ \int_{-\tau_K}^0 \mu^T(t)(u(t) - h(t))dt\,, \end{aligned} \tag{6.10}$$

where $\lambda(t)$ and $\mu(t)$ are the Lagrangian variables. In order to find the gradients of L w.r.t. θ, we first solve the following adjoint equation (for brevity we denote, $\frac{\partial}{\partial(\bullet)} \equiv \partial_{(\bullet)}$ and $\frac{d}{d(\bullet)} \equiv d_{(\bullet)}$),

$$
\begin{aligned}
d_t\lambda^T(t) &= \sum_{i=1}^{M} \partial_{u(t)} l(u(t))\delta(t - T_i) \\
&- \lambda^T(t)\partial_{u(t)} f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta) \\
&- \sum_{i=1}^{K} \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}\left(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta\right) \ , \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad t \in [0, T) \ , \\
\lambda(t) &= 0, \qquad t \geq T \ .
\end{aligned}
\tag{6.11}
$$

Details of the derivation of the above adjoint Eq. 6.11 are in the accompanying *Supplementary Information*. Note that Eq. 6.11 needs to be solved backward in time, and one would require access to $u(t),\ 0 \leq t \leq T$. In the original nODE work [33], Eq. 6.9 is solved backward in time and augmented with the adjoint Eq. 6.11, so as to shrink the memory footprint by avoiding the need to save u at every time-step. Solving Eq. 6.9 backward can however lead to catastrophic numerical instabilities as is well known in data assimilation [213, 214]. Improvements have been proposed, such as the ANODE method [215], but they are not applicable in case of DDEs. In our present implementation, in order to access $u(t),\ 0 \leq t \leq T$, while solving the adjoint equation, we create and continuously update an interpolation function using the u obtained at every time-step as we solve Eq. 6.9 forward in time. To be more memory efficient, we can, for example, use the method of *checkpointing* [216], or the interpolated reverse dynamic method (IRDM) [217]. After solving for λ, we can compute the required gradients as,

$$
d_\theta L = -\int_0^T \lambda^T(t)\partial_\theta f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta)dt \ . \tag{6.12}
$$

Finally, using any gradient descent algorithm, we can find the optimal values of the weights θ.

Distributed Delays

In some applications, the delay is distributed over some past time-period [218],

$$\begin{aligned} &\frac{du(t)}{dt} = f\left(u(t), \int_{t-\tau_2}^{t-\tau_1} g(u(\tau),\tau)d\tau, t\right), \quad t \in (0,T]\,, \\ &\text{with} \quad u(t) = h(t), \quad t \leq 0\,. \end{aligned} \tag{6.13}$$

It should be noted that the discrete DDEs can be written as a special case of distributed DDEs using dirac-delta functions. We can approximate the two functions f and g using two different neural-networks, and re-write the above Eqs. 6.13 as our new coupled discrete DDEs,

$$\begin{aligned} &\frac{du(t)}{dt} = f_{NN}\left(u(t), y(t), t; \theta\right), \quad t \in (0,T] \\ &\frac{dy(t)}{dt} = g_{NN}(u(t-\tau_1), t-\tau_1; \phi) - g_{NN}(u(t-\tau_2), t-\tau_2; \phi), \quad t \in (0,T] \\ &\text{with} \quad u(t) = h(t), \quad \tau_2 \leq t \leq 0\,, \quad \text{and} \quad y(0) = \int_{-\tau_2}^{-\tau_1} g_{NN}(h(t), t; \phi)dt\,, \end{aligned} \tag{6.14}$$

where $f_{NN}(\bullet;\theta)$ and $g_{NN}(\bullet;\phi)$ are the two NNs parameterized by θ and ϕ respectively. We refer to this parameterization of distributed DDEs as *distributed-nDDE*. The graphical representation of the above system (Eqs. 6.14) in time-discretized form is depicted in Figure 6-2b. Interestingly in the case of distributed-delays, we obtain a novel architecture consisting of two coupled NNs, which enables us to incorporate memory without the use of any recurrent networks such as RNN, LSTMs, GRUs, etc. We can consider f_{NN} as the main network, and g_{NN} as the auxiliary network. Again, we define a scalar loss function given by $\mathcal{L} = \int_0^T \sum_{i=1}^M l(u(t))\delta(t-T_i)dt$ for the available data at M times, $T_1 < ... < T_M \leq T$. The Lagrangian for the above system

is,

$$
\begin{aligned}
L =& \mathcal{L}(u(t)) + \int_0^T \lambda^T(t)(d_t u(t) - f_{NN}(u(t), y(t), t; \theta))\, dt \\
&+ \int_0^T \mu^T(t)\left(d_t y(t) - g_{NN}(u(t-\tau_1), t-\tau_1; \phi) + g_{NN}(u(t-\tau_2), t-\tau_2; \phi)\right) dt \\
&+ \int_{-\tau_2}^0 \gamma^T(t)(u(t) - h(t))dt + \alpha^T \left(y(0) - \int_{-\tau_2}^{-\tau_1} g_{NN}(h(t), t; \phi)dt \right) ,
\end{aligned}
\tag{6.15}
$$

where $\lambda(t), \mu(t), \gamma(t)$, and α are the Lagrangian variables. In order to find the gradients of L w.r.t. the parameters of the two NNs, we first solve the following coupled adjoint equations backward in time,

$$
\begin{aligned}
d_t \lambda^T(t) =& \sum_{i=1}^{M} \partial_{u(t)} l(u(t)) \delta(t - T_i) - \lambda^T(t) \partial_{u(t)} f_{NN}(u(t), y(t), t; \theta) \\
&- \mu^T(t + \tau_1) \partial_{u(t)} g_{NN}(u(t), t; \phi) \\
&+ \mu^T(t + \tau_2) \partial_{u(t)} g_{NN}(u(t), t; \phi)\,, \ t \in [0, T) \\
d_t \mu^T(t) =& - \lambda^T(t) \partial_{y(t)} f_{NN}(u(t), y(t), t; \theta)\,, \qquad t \in [0, T) \\
\lambda^T(t) =& \ 0 \quad \text{and} \quad \mu^T(t) = 0, \qquad t \geq T\,.
\end{aligned}
\tag{6.16}
$$

Details of the derivation of the above adjoint Eq. 6.16 are in the *Supplementary Information*. For accessing u values while solving the adjoint equations, we use the same approach as for our discrete-nDDE (Sec. 6.2.2). After solving for λ and μ, we can compute the required gradients as,

$$
\begin{aligned}
d_\theta L =& - \int_0^T \lambda^T(t) \partial_\theta f_{NN}(u(t), y(t), t; \theta) dt\,, \\
d_\phi L =& - \int_0^T \mu^T(t)\left(\partial_\phi g_{NN}(u(t-\tau_1), t-\tau_1; \phi) - \partial_\phi g_{NN}(u(t-\tau_2), t-\tau_2; \phi)\right) dt \\
&- \mu^T(0) \int_{-\tau_2}^{-\tau_1} \partial_\phi g_{NN}(h(t), t; \phi) dt\,.
\end{aligned}
\tag{6.17}
$$

Finally, using any gradient descent algorithm, we can optimize the neural-networks f_{NN} and g_{NN}, and find the optimal values of the weights θ and ϕ.

6.2.3 Neural Closure Models

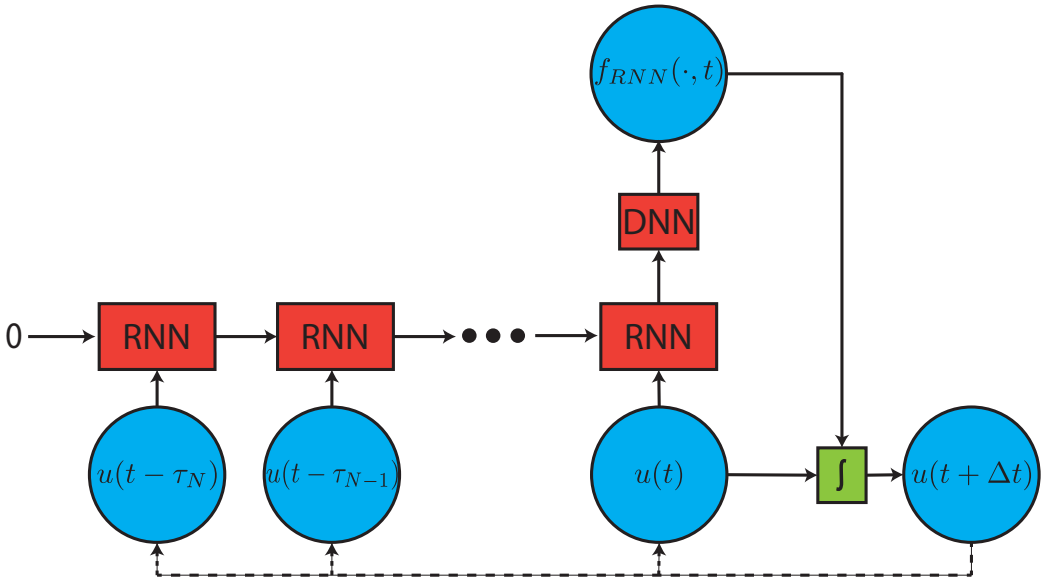
Now that we have the framework for representing delay differential equations using neural-networks, we can replace the non-Markovian memory term in Eq. D.6 using nDDEs to obtain a hybrid closure model which could be trained using data from high-fidelity simulations or real observations. The modified low-fidelity dynamical system with the nDDE closures, which approximates the high-fidelity model would be given by,

$$\begin{aligned} \frac{\partial \hat{u}(t)}{\partial t} &= \underbrace{PR(\hat{u}(t))}_{\text{Low-Fidelity}} + \underbrace{f_{RNN}(\hat{u}(t), \hat{u}(t-\tau_1), ..., \hat{u}(t-\tau_K), t; \theta)}_{\text{Neural Closure}} \\ \hat{u}(0) &= \hat{u}_0, \qquad t \leq 0 \end{aligned} \tag{6.18}$$

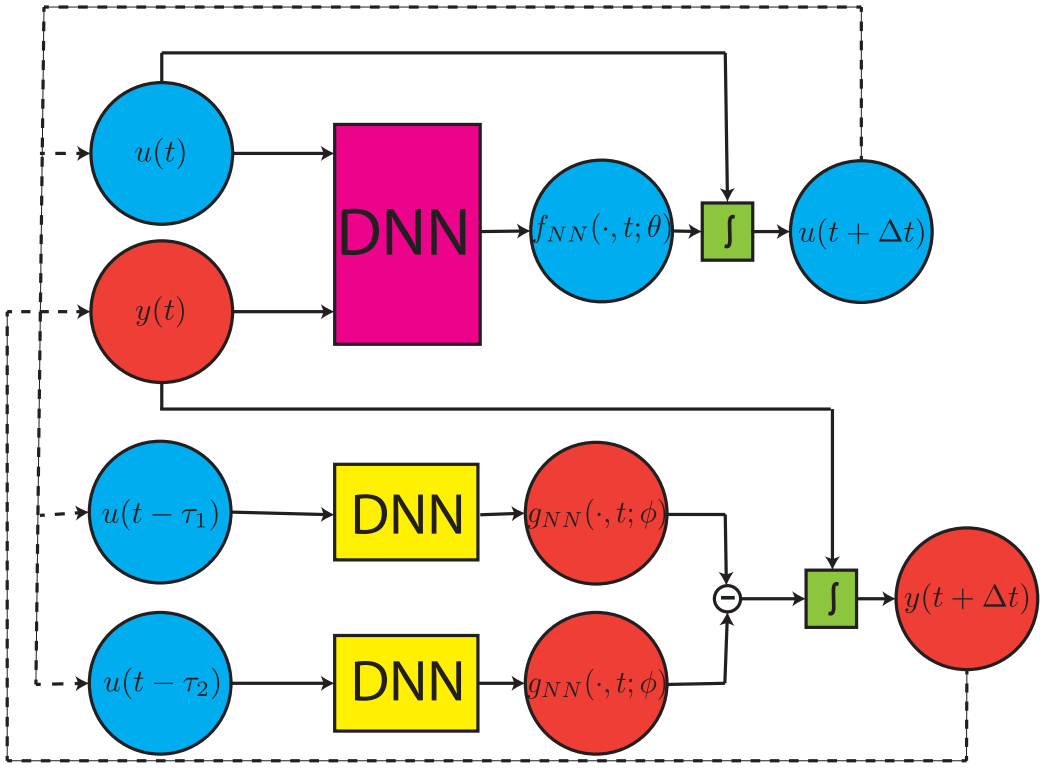
using discrete-delays, or by,

$$\begin{aligned} \frac{\partial \hat{u}(t)}{\partial t} &= \underbrace{PR(\hat{u}(t))}_{\text{Low-Fidelity}} + \underbrace{f_{NN}\left(\hat{u}(t), \int_{t-\tau_2}^{t-\tau_1} g_{NN}(\hat{u}(\tau), \tau; \phi) d\tau, t; \theta\right)}_{\text{Neural Closure}} \\ \hat{u}(0) &= \hat{u}_0, \qquad t \leq 0 \end{aligned} \tag{6.19}$$

using distributed-delays. The initial conditions at $t = 0$, $\hat{u}_0$, can be used for $t < 0$ as well, as an approximation. Apart from the neural-network architectures, the amount of delay to be used also becomes a hyperparameter to tune. These novel *neural closure models* provides extreme flexibility in designing the non-Markovian memory term in order to incorporate subject matter expert insights. At the same time, we can also learn the unknown parts of the Markovian low-fidelity model using nODEs if the need arises. Next, we will compare the performance and advantages of using no-delays (nODEs), discrete-delays (discrete-nDDEs), and distributed-delays (distributed-nDDEs) in closure terms for various low-fidelity dynamical systems.



(a) Discrete-nDDE



(b) Distributed-nDDE

Figure 6-2: Graphical representation of the time discretized neural delay differential equations (nDDEs). The blocks labeled *RNN* and *DNN* represent any recurrent or deep neural-network architectures respectively. The block labeled $\int$ symbolizes any time-integration scheme.

6.3 Application Results and Discussion

After presenting the main classes of low-fidelity dynamical models that require closure (Sec. 6.1), we derived a novel, versatile, and rigorous methodology for learning and modeling non-Markovian closure terms using nDDEs (Sec. 6.2). The resulting neural closure models have their underpinning in the Mori-Zwanzig formulation and the presence of inherent delays in models of complex dynamical systems such as biogeochemical systems. Now, we evaluate the performance and advantages of these new neural closure models over those of neural ODEs (Markovian).

We run experiments encompassing each of the classes of low-fidelity models (Sec. 6.1). For each experiment, we follow the same training protocol for nODEs (no-delays) and the two nDDEs, discrete-nDDE (discrete-delays) and distributed-nDDE (distributed-delays), closure models. The training data are regularly sampled from high-fidelity simulations in all experiments, but this is not a requirement. We use performance over the validation period (past the period for which high-fidelity data snapshots are used for training) to fine tune various training related hyperparameters. The final evaluation is based on much longer-term future prediction performance, well past these periods. As the field of scientific machine learning (SciML; [37]) is relatively new, the metrics for performance evaluation vary greatly. On the one hand, many learning studies randomly sample small time-sequences from a given period for which high-fidelity data are available, and then split them into training, validation, and test sets [165]. As the training, validation, and test sets belong to the same time-domain, hence, the learned networks are only evaluated for their interpolation performance and predicting the unseen data becomes easy for them. On the other hand, for the few studies where the training and test (prediction) periods do not overlap, the prediction period is often much shorter than the training period [219]. In the present work, we consider a more stringent evaluation. First, our validation period does not overlap the training period. Second, our future prediction period is equal to or much longer than the training and validation periods, and has no overlap with either. Hence, we strictly measure the out-of-sample/generalization performance of the learned network

for its extrapolation capabilities into the future. Of course, other evaluation metrics are possible and there is indeed a need for standardization of evaluation procedures in the SciML community. In the rest of the paper, for all the figure, table, and section references prefixed with "SI-", we direct the reader to the *Supplementary Information*.

6.3.1 Experiments 1: Advecting Shock - Reduced Order Model

For the first experiments, neural closure models learn the closure of proper-orthogonal-decomposition Galerkin projection (POD-GP) based reduced order model of the advecting shock problem. The full-order-model (FOM) for this problem is given by the Burger's equation,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial^2 x}\,, \tag{6.20}$$

where ν is the non-dimensional diffusion coefficient. The initial and boundary conditions are

$$u(x,0) = \frac{x}{1+\sqrt{\frac{1}{t_0}}\exp\left(Re\frac{x^2}{4}\right)}\,, \quad u(0,t) = 0, \quad \text{and} \quad u(L,t) = 0\,, \tag{6.21}$$

where $Re = 1/\nu$ and $t_0 = \exp(Re/8)$. Let the POD of the state variable $u(x,t)$ be given by, $u(x,t) = \bar{u}(x) + \sum_{i=1}^{m} u_i(x)a_i(t)$, we obtain the reduced-order equations as outlined in Section 6.1.1,

$$\begin{aligned}\frac{da_k}{dt} =& -\left\langle \bar{u}\frac{\partial \bar{u}}{\partial x}, u_k \right\rangle - a_i \left\langle u_i \frac{\partial \bar{u}}{\partial x}, u_k \right\rangle - a_j \left\langle \bar{u}\frac{\partial u_j}{\partial x}, u_k \right\rangle - a_i a_j \left\langle u_i \frac{\partial u_j}{\partial x}, u_k \right\rangle \\ &+ \left\langle \nu \frac{\partial^2 \bar{u}}{\partial x^2}, u_k \right\rangle + a_i \left\langle \nu \frac{\partial^2 u_i}{\partial x^2}, u_k \right\rangle\,, \\ \text{with} \quad & a_k(0) = \langle (u(x,0) - \bar{u}(x)), u_k(x) \rangle\,.\end{aligned} \tag{6.22}$$

We solve the FOM (Eqs. 6.20 and 6.21) for $Re = 1000$, $L = 1$, and maximum time $T = 4.0$. The singular value decomposition (SVD) of this solution form the POD modes for the ROM. We only keep the first three modes which capture 60.8% of energy, and evolve the corresponding coefficients using Eq. 6.22, thus requiring a closure. The high-fidelity or true coefficients are obtained solving the FOM (Eq. 6.20) with

initial conditions without the contribution from the unresolved modes, i.e. $u(x,0) = \bar{u}(x)+\sum_{i=1}^{3} u_i(x)a_i(0)$, and projecting the obtained solution onto the first three modes. For comparison, we also present the true coefficients in Fig. 6-3, which is what the ROMs with neural closure are trying to match. For this true data generation, we solve the FOM using an explicit Runge-Kutta (RK) time-integration of order (4)5 (*dopri5*; [134]) with adaptive time-stepping (storing data at time-steps of $\Delta t = 0.01$) and grid spacing of $\Delta x = 0.01$, using finite-difference schemes (upwind for advection and central difference for diffusion).

Our three test periods for the advecting shock ROM (Eq. 6.22) with three modes are as follows. For training our neural closure models, we only use the true coefficient values up to time $t = 2.0$. For validation (used only to tune hyperparameters), we use true coefficient values from $t = 2.0$ to $t = 4.0$. Finally for testing, we make a future prediction from $t = 4.0$ to final time $T = 6.0$. We compare the three different closures: nODE (no-delays), discrete-nDDE, and distributed-nDDE with architecture details presented in Table D.1. The architectures are not exactly the same for the three cases, but they are set-up to be of comparable expressive power. Mostly, we employ a bigger architecture for the no-delays case in order to help it compensate the lack of past information. We also ensure that the networks are neither under-parameterized nor over-parameterized. Along with the classical hyperparameters such as batch size, number of iterations per epoch, number of epochs, learning rate schedule, etc., we also have the delay values ($\tau_1, ..., \tau_K$ for discrete-nDDE; and $\tau_1,\ \tau_2$ for distributed-nDDE) as additional hyperparameters to tune. We chose to use six discrete delays for the discrete-nDDE in the present experiments. The values of other hyperparameters are given in Sec. D.3.2. For evaluation, at each epoch, we evolve the coefficient of the learned system ($\{a^{pred}(T_i) = \{a_k^{pred}(T_i)\}_{k=1}^{3}\}_{i=1}^{M}$) using the RK time-integration scheme mentioned earlier, and compare them with the true coefficients ($\{a^{true}(T_i)\}_{i=1}^{M}$) using the time-averaged L_2 error, $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\left(\sqrt{\sum_{k=1}^{3}|a_k^{pred}(T_i) - a_k^{true}(T_i)|^2}\right)$, which is also our loss function for training. The error for the time period $t = 0$ to 2.0 forms the training loss, the error for $t = 2.0$ to 4.0 the validation loss, and the error for $t = 4.0$ to 6.0 the prediction loss.

The performance of the three neural closure models after 200 epochs (the stochastic gradient descent nearly converges, see Fig. D-1a) is evaluated by comparison with the true coefficients and with the POD-GP coefficients spanning training, validation, and future prediction periods. Results are shown in Figure 6-3. The details of the architectures employed are in Table D.1. We find that using no-delays (nODE), discrete-delays (discrete-nDDE), and distributed-delays (distributed-nDDE) perform equally well for the training period, exactly matching the true coefficients. As soon as one enters the validation period, all the neural closure models starts to slightly diverge, with the nODE diverging the most by the end of prediction period. Importantly, both nDDE closures maintain a great improvement over just using the POD-GP model, and showcase a better performance than the nODE closure, even though the latter had a deeper architecture with significantly more trainable parameters. We also find that the performance of the distributed-nDDE closure is a little better than that of the discrete-nDDE closure for the prediction period. In a similar set of experiments, Maulik et. al., 2020 (section 3.1, "Advecting Shock", [165]) used nODE and LSTM to learn the time evolution of the first three high-fidelity (true) coefficients without utilizing the known physics/low-fidelity model. As a result, they required bigger architectures and more time samples for training data than we do. This confirms our benefits of learning only the unknown closure model. Due to the highly nonlinear nature of neural networks, analytical stability analyses are not direct. Nonetheless, we provide empirical stability results by reporting the evolution of the root-mean-square-error (RMSE) (Fig. 6-3). We find that both discrete-nDDE and distributed-nDDE closures, due to the existence of delays, may have a stronger dissipative character and thus show much better stability at later times than the POD-GP and the nODE closure.

One might expect the distributed-delay (distributed-nDDE) to always perform better than the discrete-nDDE closure because of the presence of the integral of the state variable over a delay period instead of the state variable at specific points in the past. The former thus seemingly contains more information, but there is in fact no guarantee for this being true in all cases. We can derive an intuition for this from

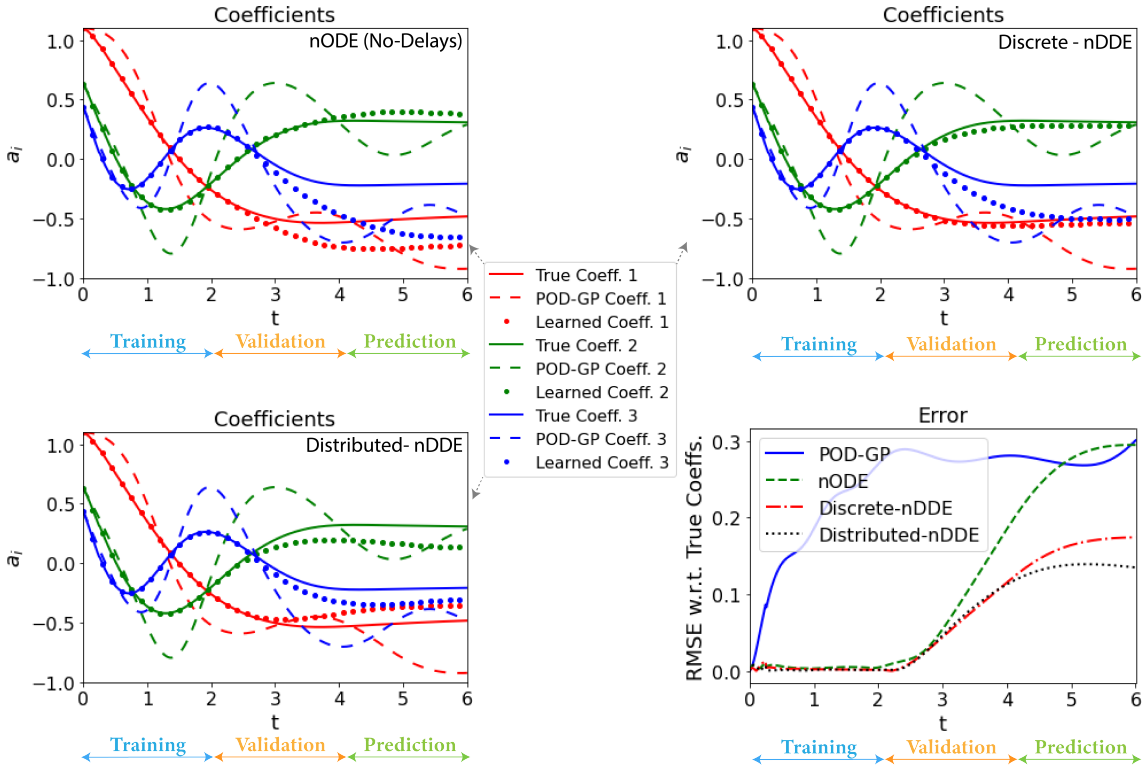


Figure 6-3: Comparison of the true coefficients (*solid*) with the coefficients from the POD-GP ROM (*dashed-dot*) and from the POD-GP ROMs augmented with the three different learned neural closure models at the end of training (*dashed*). For each neural closure, the training period is from $t = 0$ to 2.0, the validation period from $t = 2.0$ to 4.0, and the future prediction period from $t = 4.0$ to 6.0. *Top-left:* neural ODEs with no-delays (nODE); *Top-right:* neural DDEs with discrete-delays (Discrete-nDDE); *Bottom-left:* neural DDEs with distributed-delays (Distributed-nDDE). *Bottom-right:* Evolution of root-mean-squared-error ($\text{RMSE}(t) = \sqrt{\frac{1}{3}\sum_{k=1}^{3} |a_k^{pred}(t) - a_k^{true}(t)|^2}$) of coefficients from the four different ROMs. These results correspond to the architectures detailed in Table D.1.

information theory. According to the data processing inequality [142], let X and Y be two random variables, then,

$$I(g(X);Y) \leq I(X;Y)\,, \tag{6.23}$$

where I is the mutual information and g is any function which post-processes X. Now, if X is composed of K random variables, $X = \{X_1, ..., X_K\}$, and $g(X) = X_1 + ... + X_K$, then,

$$I(X_1 + ... + X_K;Y) \leq I(\{X_1, ..., X_K\};Y)\,. \tag{6.24}$$

If we consider the effect of the integral of the state variable over the delay period in the case of distributed-nDDE as a data processing step, this might actually be decreasing the information content as compared to the discrete-nDDE closure. We use "might", even though Eq. 6.24 is a strong bound, because in the present experiments we only use six delay values for the discrete-nDDE, while the integral in the distributed-nDDE is computed using many past state values, and also the architectures are different. Hence, a direct comparison using the data processing inequality (Eq. 6.24) is not possible, but it provides us with a plausible explanation.

In addition to the results just illustrated, we completed many other experiments-1 to assess the sensitivity of our framework to various hyperparameters. In all cases, the time-period corresponding to the training data should be at least equal to one characteristic time-scale of the dynamics, otherwise the prediction performance deteriorated, as shown in the Fig. D-2a and discussed in Sec. D.3.3. Adding the neural closure to the low-fidelity model improved its matching with the high-fidelity data in nearly all cases. Its performance deteriorated with increasing the length of the time-sequences used to form the batches, and also with increasing the batch-size (the number of iterations per epoch is a dependent hyperparameter as mentioned earlier). This also led to an increase in training time. Depth of the networks affected the performance significantly, with shallower networks performing poorly than deeper networks as expected, however, the incremental gain in performance starts to taper off after certain depths (see Fig. D-2b and Sec. D.3.3). Using an exponentially decaying

learning schedule over a constant learning rate tremendously improved learning performance and reduced the number of epochs needed. Further, training times slightly increased when using more delay times in the case of discrete-nDDE. In general, we found that the training time for discrete-nDDEs was similar to that for distributed-nDDEs. Such behaviors by machine learning methods are difficult to anticipate in advance but they should be mentioned.

Overall, in the experiments-1, we find that using memory-based neural closure models as we derived from the Mori-Zwanzig formulation is advantageous over just a Markovian closure. Using the new nDDEs as closure models helps maintain generalizability of the learned models for longer time-periods, and significantly reduces the longer-term prediction error of the ROM.

6.3.2 Experiments 2: Advecting Shock - Subgrid-Scale Processes

In the second experiments, we again use the advecting shock problem governed by the Burger's equation (Eq. 6.20), but we now reduce the computational cost of the FOM by coarsening the spatial resolution, again leading to the need of a closure model (Sec. 6.1.2). For the high-fidelity/high-resolution solution, we employ a fine grid with the number of grid point in the x direction $N_x = 100$, while for the low-fidelity/low-resolution solution, we employ a 4 times coarser grid with $N_x = 25$. A comparison of high- and low- resolution solutions solved using exactly the same numerical schemes and data stored at every time-step of $\Delta t = 0.01$ is provided in Fig. 6-4. We observe that by decreasing the resolution, we introduce numerical diffusion and error in the location of the shock peak at later times. The goal of the neural closure models in these experiments is thus to augment the low-resolution model such that it matches the sub-sampled/interpolated high-resolution solution at the coarse (low-resolution) grid points.

For training our neural closure models for the low-resolution discretization with $N_x = 25$, we use the same training regiment as in Experiments-1 (Sec. 6.3.1), with

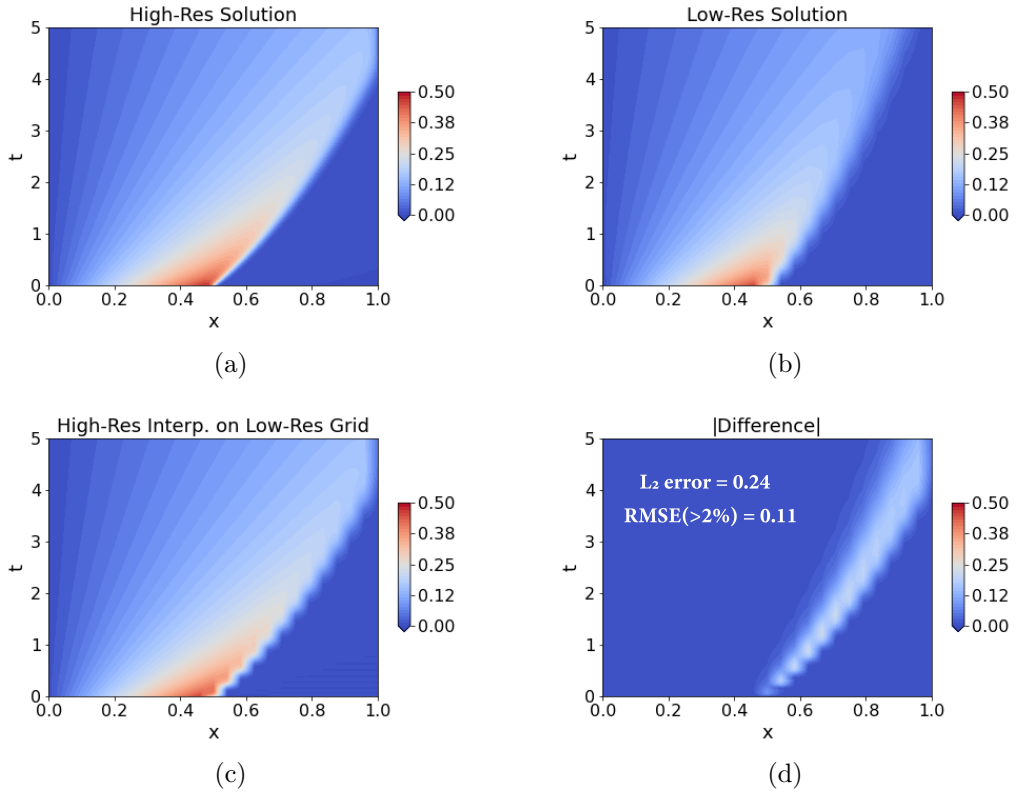


Figure 6-4: Comparison of solutions of Burger's equation (Eq. 6.20) for different grid resolutions. *(a):* Solution for a high-resolution grid with number of grid points, $N_x = 100$; *(b):* Solution for a low-resolution grid with $N_x = 25$; *(c):* High-resolution solution interpolated onto the low-resolution grid. *(d):* Absolute difference between fields in panels (b) and (c). We also provide a pair of time-averaged errors, specifically: L_2 error; and RMSE considering only the grid points where the error is at least 2% of the maximum velocity value, denoted by *RMSE(>2%)*.

architectures details presented in Table D.1. In order to exploit the fact that each grid point only affects its immediate neighbors over a single time-step, we use 1-D convolutional layers for these experiments. For the nODE, we again employ a deeper architecture with more trainable parameters, and for the discrete-nDDE, six discrete delay values are again used. The values of the other hyperparameters are in Sec. D.3.2. the validation period from $t = 1.25$ to 2.5, and the future prediction period from $t = 2.5$ to 5.0. We have chosen a prediction period of the combined length of training and validation periods. For time-integration, we use the *Vode* scheme [220] with adaptive time-stepping. The true data are generated by interpolating the high-resolution solution onto the low-resolution grid

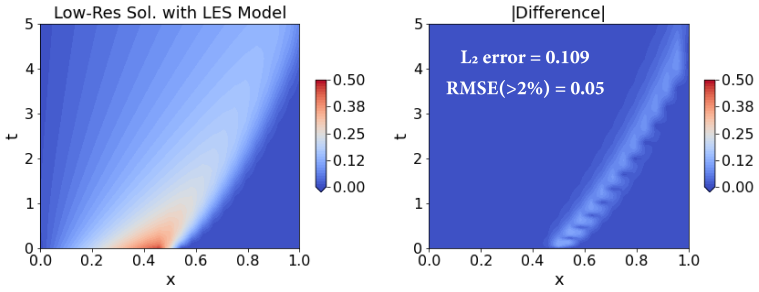
$(\{\{u^{true}(x_k,T_i)\}_{k=1}^{N_x=25}\}_{i=1}^{M})$, as shown in Fig. 6-4c, and we use the time-averaged L_2 error, $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\left(\sqrt{\sum_{k=1}^{N_x=25}|u^{pred}(x_k,T_i)-u^{true}(x_k,T_i)|^2}\right)$, as the loss function.

The performance of the three neural closure models after 250 epochs (the stochastic gradient descent nearly converges, see Fig. D-1b) is evaluated by taking the absolute difference with the high-resolution solution interpolated onto the low-resolution grid (Fig. 6-4c) spanning training, validation, and prediction periods. We further benchmark our performance against the popular Smagorinsky model [221] used for subgrid-scale turbulence closure in large eddy simulation (LES). For the Burger's Eq. 6.20, it introduces a dynamic turbulent eddy viscosity (ν_e) leading to,

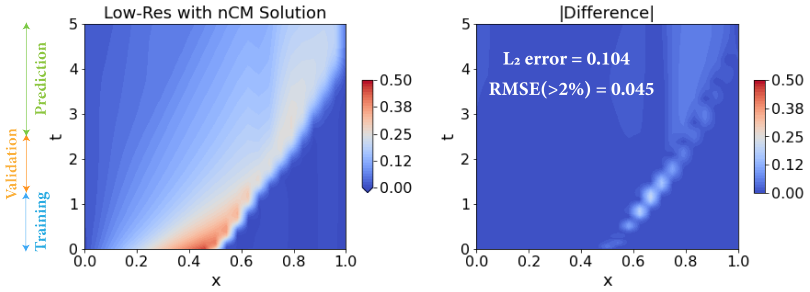
$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} + \frac{\partial}{\partial x}\left(\nu_e\frac{\partial u}{\partial x}\right), \tag{6.25}$$

where $\nu_e = (C_s\Delta x)^2\left|\frac{\partial u}{\partial x}\right|$ and C_s is the Smagorinsky constant. Results are shown in Fig. 6-5. The details of the architectures employed are in Table D.1. As shown by the error fields of the baseline (Fig. 6-4d) and closure models (Fig. 6-5), and by the corresponding pairs of averaged error numbers (see Figs.), all closures improve the baseline. However, the nODE and Smagorinsky closures only lead to a 55-60% decrease in error, while the nDDE closures achieve a 80-90% decrease. Despite the deeper architecture for the nODE, both the discrete-nDDE and distributed-nDDE (with smaller architectures) again achieve smaller errors, for the whole period of $t = 0$ to 5.0. This means that they have lower numerical diffusion, thus capturing the targeted subgrid-scale process. As opposed to the findings of experiments-1 (Fig. 6-3), in the present experiments-2, the discrete-nDDE performs slightly better than the distributed-nDDE in the prediction period.

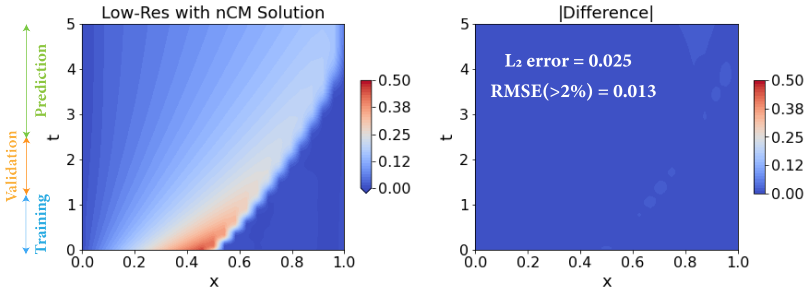
We now study the effect of changing the amount of past information incorporated in the closure model on the time-averaged L_2 error. For this, we fix $\tau_1 = 0$ for the distributed-nDDE closure, and vary the values of only τ_2, keeping the architecture the same (Table D.1). First, for the time-averaged training loss (not shown), we found no discernible trend and differences were mostly due the stochastic gradient descent. Second, for the validation period, Fig. 6-6a shows the statistical summary



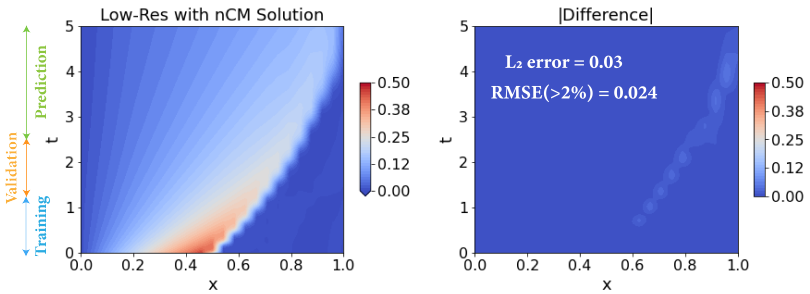
(a) Smagorinsky LES model



(b) Neural closure model with no-delays (nODE)



(c) Neural closure model with discrete-delays (Discrete-nDDE)



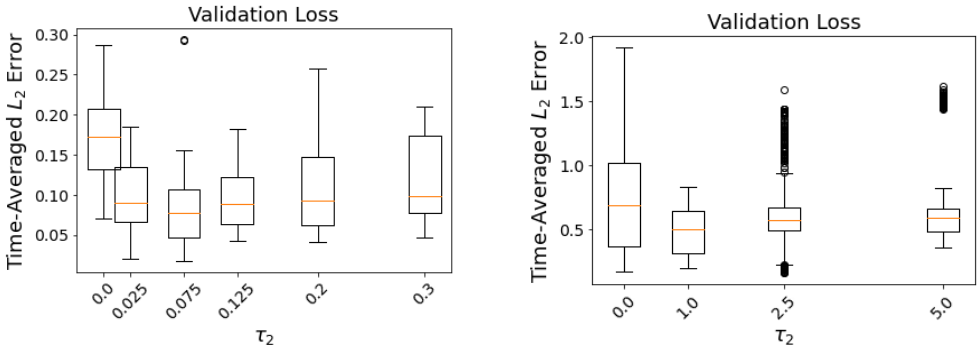
(d) Neural closure model with distributed-delays (Distributed-nDDE)

Figure 6-5: Solutions of the Burger's PDE on the low-resolution grid with different closure models (*left-column*), and their absolute differences (*right-column*) with the high-resolution solution interpolated onto the low-resolution grid (Fig. 6-4c). For the trained neural closure models, the training period is from $t = 0$ to 1.25, the validation period from $t = 1.25$ to 2.5, and the prediction period from $t = 2.5$ to 5.0. For each closure, we also provide the pair of time-averaged errors (see Fig. 6-4 for description). *(a):* Smagorinsky LES model with $C_s = 1.0$; *(b)*, *(c)*, *(d)*: different neural closure models. These results correspond to the architectures detailed in Table D.1.

of the validation losses (time-averaged L_2 error) between the last epochs 200 to 250, for different delay-period lengths. In order to ensure statistical soundness of the results, $10-15$ repeats of the training were done, and the results aggregated for each delay-period length. Results indicate that the validation loss first decreases and then increases as we incorporate more-and-more past information, starting from a very small delay period. For a specified architecture, neither too little nor too much past information is helpful: there is likely an optimal amount of information to incorporate. The initial improvement in the performance of the closure models as the delay period is increased is due to the increase in information content about the recent past. However, a particular network architecture of finite size will have a limit on capturing the increasing information content effectively due to its limited expressive power, thus leading to a decrease in performance when too much information is provided. An estimate of the range of delay period lengths to consider can be obtained from properties of the given dynamical system such as the main time-scales, e.g. advection and diffusion times-scales in the present system, and main decorrelation times of state variables. Overall, from Fig. 6-6a, we can notice the optimal delay period length to be around 0.075. Similar trends between performance of neural closure models and delay period lengths were also found in Experiments-1 (not shown). Some published studies attempt to derive analytical expressions for the optimal memory length, making many approximations in the process [222, 159]. A final option is to learn the delay amount as a part of the training process itself, however, this requires modified adjoint equations.

We conducted again a series of experiments-2 to assess the sensitivity to the various other hyperparameters, and found similar trends (not shown here) as in the experiments-1. Finally, we noticed that using the *dopri5* [134] time-integration scheme severely impaired the learning ability in the experiments-2.

Overall, these results demonstrate the superiority of using our new memory-based closure models in capturing subgrid-scale processes.



(a) Experiments - 2

(b) Experiments - 3a

Figure 6-6: Variation of distributed-nDDE closure validation loss (time-averaged L_2 error) averaged over the last 50 training epoch for Experiments-2 & 3a. All the experiments have $\tau_1 = 0$, and different τ_2 (horizontal-axis). Note that $\tau_2 = 0$ corresponds to the nODE closure. We use boxplots to provide statistical summaries for multiple training repeats done for each experiment. The box and its whiskers provide a five number summary: minimum, first quartile (Q1), median (orange solid line), third quartile (Q3), and maximum, along with outliers (black circles) if any.

6.3.3 Experiments 3a: 0-D Marine biological Models

For our third experiments, we use neural closure models to incorporate the effects of missing processes and state variables in lower-complexity biological models, thus targeting the third class of closure modeling (Sec. 6.1.3). Marine biological models are based on ODEs that describe the food-web interactions in the ecosystem. They can vary greatly in terms of complexity [39]. The marine biological models used in our experiments are adapted from Newberger et. al., 2003 ([3]). They were used to simulate the ecosystem in the Oregon coastal upwelling zone. They provide hierarchical embedded models compatible with each other. We employ the three-component NPZ model (nutrients (N), phytoplankton (P), and zooplankton (Z)), and the five-component NNPZD model (ammonia (NH_4), nitrate (NO_3), P, Z, and detritus (D)) in a zero-dimensional setting (0-D; only temporal variation). The low complexity

NPZ model is given by,

$$\begin{aligned}
\frac{dN}{dt} &= -G\frac{PN}{N+K_u} + \Xi P + \Gamma Z + R_m\gamma Z(1-\exp^{-\Lambda P}) \\
\frac{dP}{dt} &= G\frac{PN}{N+K_u} - \Xi P - R_m Z(1-\exp^{-\Lambda P}) \\
\frac{dZ}{dt} &= R_m(1-\gamma)Z(1-\exp^{-\Lambda P}) - \Gamma Z \\
\text{with} \quad N(0) &= T_{bio}, \quad P(0) = 0, \quad \text{and} \quad Z(0) = 0\,,
\end{aligned} \tag{6.26}$$

with G representing the optical model,

$$G = V_m\frac{\alpha I}{(V_m^2+\alpha^2 I^2)^{1/2}}, \quad \text{and} \quad I(z) = I_0\exp(k_w z)\,, \tag{6.27}$$

where z is depth and $I(z)$ models the availability of sunlight for photo-chemical reactions. The parameters in Eqs. 6.26 and 6.27 are: k_w, light attenuation by sea water; α, initial slope of the $P-I$ curve; I_0, surface photosynthetically available radiation; V_m, phytoplankton maximum uptake rate; K_u, half-saturation for phytoplankton uptake of nutrients; Ξ, phytoplankton specific mortality rate; R_m, zooplankton maximum grazing rate; Λ, Ivlev constant; γ, fraction of zooplankton grazing egested; Γ, zooplankton specific excretion/mortality rate; and T_{bio}, total biomass concentration. In the NPZ model (Eq. 6.26), the nutrient uptake by phytoplankton is governed by a Michaelis-Menten formulation, which amounts to a linear uptake relationship at low nutrient concentrations that saturates to a constant at high concentrations. The grazing of phytoplankton by zooplankton follows a similar behavior: their growth rate becomes independent of P in case of abundance, but proportional to available P when resources are scarce, hence zooplankton grazing is modeled by an Ivlev function. The death rates of both P and Z are linear, and a portion of zooplankton grazing in the form of excretion goes directly to nutrients.

In the higher complexity NNPZD model, the nutrients are divided into ammonia and nitrates, which are the two most important forms of nitrogen in the ocean. With the intermediate of decomposed organic matter, detritus, the NNPZD model captures new processes such as: phytoplankton cells preferentially taking up ammonia over ni-

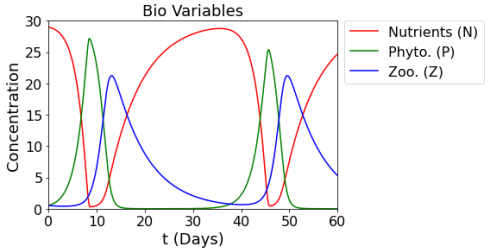
trate because the presence of ammonia inhibits the activity of the enzyme nitrate reductase essential for the uptake kinetics; the pool of ammonium coming from remineralization of detritus; and part of this ammonium pool getting oxidized to become a source of nitrate called the process of nitrification, etc. Overall, the NNPZD model is given by,

$$
\begin{aligned}
\frac{dNO_3}{dt} &= \Omega NH_4 - G\left[\frac{NO_3}{NO_3+K_u}\exp^{-\Psi NH_4}\right]P \\
\frac{dNH_4}{dt} &= -\Omega NH_4 + \Phi D + \Gamma Z - G\left[\frac{NH_4}{NH_4+K_u}\right]P \\
\frac{dP}{dt} &= G\left[\frac{NO_3}{NO_3+K_u}\exp^{-\Psi NH_4} + \frac{NH_4}{NH_4+K_u}\right]P - \Xi P - R_m Z(1-\exp^{-\Lambda P}) \\
\frac{dZ}{dt} &= R_m(1-\gamma)Z(1-\exp^{-\Lambda P}) - \Gamma Z \\
\frac{dD}{dt} &= R_m\gamma Z(1-\exp^{-\Lambda P}) + \Xi P - \Phi D \\
\text{with} \quad & NO_3(0) = T_{bio}/2, \quad NH_4(0) = T_{bio}/2, \\
& P(0) = 0, \quad Z(0) = 0, \ \text{ and } \ D(0) = 0\,,
\end{aligned}
\tag{6.28}
$$

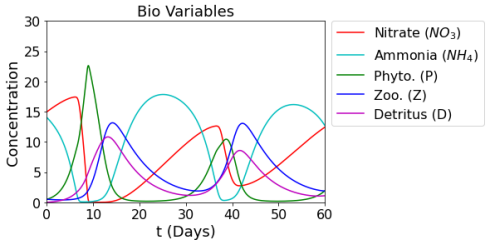
where the new parameters are: Ψ, NH_4 inhibition parameter; Φ, detritus decomposition rate; and Ω, NH_4 oxidation rate.

Solutions of the above two models are presented in Fig. 6-7. Different values of the parameters and initial conditions set these models in different dynamical regimes. From the responses in time, the present solutions in experiments-3a are in a stable nonlinear limit-cycle regime. The N class in the NPZ model is a broader class encompassing NO_3, NH_4, and D from the NNPZD model. Since the NNPZD model resolves many more processes, the concentrations of $NO_3 + NH_4 + D$, P, and Z differ significantly from the N, P, and Z of the NPZ model. The goal of the neural closure models in these experiments is thus to augment the low-complexity NPZ model such that it matches the aggregated states of the high-complexity NNPZD model.

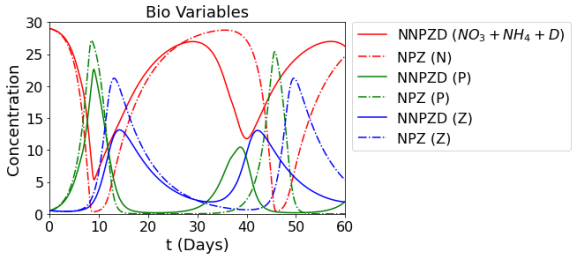
For training our neural closure models for the NPZ model, we use the same training regiment as in Experiments-1 & 2 (Secs. 6.3.1 & 6.3.2), with architectures de-



(a) NPZ Model



(b) NNPZD Model



(c) Comparison of NPZ and NNPZD Models

Figure 6-7: Solutions of the marine biological models used in Experiments-3a (concentrations vs. time in *days*). Parameter values used are (adopted from [3]): $k_w = 0.067\ m^{-1}$, $\alpha = 0.025\ (W\ m^{-2}\ d)^{-1}$, $V_m = 1.5\ d^{-1}$, $I_0 = 158.075\ W\ m^{-2}$, $K_u = 1\ mmol\ N\ m^{-3}$, $\Psi = 1.46\ (mmol\ N\ m^{-3})^{-1}$, $\Xi = 0.1\ d^{-1}$, $R_m = 1.52\ d^{-1}$, $\Lambda = 0.06\ (mmol\ N\ m^{-3})^{-1}$, $\gamma = 0.3$, $\Gamma = 0.145\ d^{-1}$, $\Phi = 0.175\ d^{-1}$, $\Omega = 0.041\ d^{-1}$, $z = -25\ m$, and $T_{bio} = 30\ mmol\ N\ m^{-3}$. *(a):* Nutrient-Phytoplankton-Zooplankton (NPZ) model (Eq. 6.26); *(b):* Nitrate-Ammonia-Phytoplankton-Zooplankton-Detritus (NNPZD) model (Eq. 6.28); *(c):* Comparison between $NO_3 + NH_4 + D$, P, and Z from the NNPZD model (*solid*) with N, P and Z from the NPZ model (*dashed-dot*).

tails presented in Table D.2. For the nODE, we again employ a bigger architecture, and for the discrete-nDDE, six discrete delay values are again used. The values of other hyperparameters are given in Sec. D.3.2. The training period ranges from $t = 0$ to 30 *days*, validation period from $t = 30$ to 60 *days*; and the prediction period from $t = 60$ to 330 *days*. We have chosen a prediction period nine times longer than the training period. For biological ODE models, there exists invariant knowledge about the system, such as biological state variables cannot be negative, and the sum of all the states remains constant with time (this can be verified by summing the ODEs of NPZ or NNPZD models). We enforce the constraints as follows. The positivity is enforced as a penalization term in the loss function. The constant total biomass constraint is embedded in the architectures of neural closures by intro-

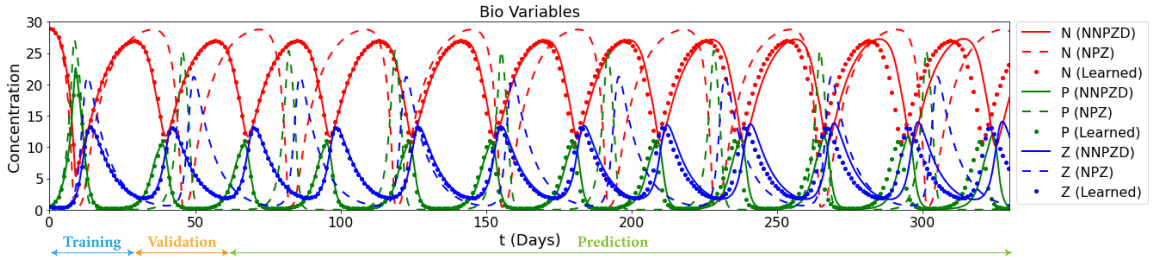
ducing a new custom layer named, *BioConstrainLayer*. This layer is applied at the end, and expects an input of size 1. The output of this layer is formed by splitting the input into three with the proportions, β, -1, and $1-\beta$; where β is a trainable parameter. This ensures that summing the right hand side of the augmented NPZ model does not leave any new residual due to the neural closure terms. The stiffness of such biological ODE models also poses a challenge in maintaining these desired properties [223]. The flexibility of our framework however allows the use of appropriate time-stepping schemes, such as A-stable implicit schemes, etc., to overcome stiffness. The true data are generated by aggregating the variables of the NNPZD model ($N \equiv NO_3 + NH_4 + D$, P, and Z; $\{\{B^{true}(T_i)\}_{B\in\{N,P,Z\}}\}_{i=1}^{M}$). Finally, we use the *dopri5* [134] scheme with adaptive time-stepping and simulation data were stored at every $\Delta t = 0.05$ days for all our time-integration requirements, along with a L_2 error loss function, $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\left(\sqrt{\sum_{B\in\{N,P,Z\}}|B^{pred}(T_i) - B^{true}(T_i)|^2}\right)$.

The performance of the three neural closure models augmenting the NPZ is evaluated after 350 epochs of training (the stochastic gradient descent nearly converges, as evident from the Fig. D-1c) by comparison with the aggregated biology variables from the high-complexity NNPZD model (Eq. 6.28) spanning training, validation, and prediction periods. Results are presented in Figure 6-8. The details of the architectures employed are in Table D.2. When compared with the aggregated NNPZD variables (true variables), we find again that despite the bigger architecture of the nODE, it starts to develop significant errors around $t = 180$ *days* and quickly gets out-of-phase thereafter (Fig. 6-8a). The discrete-nDDE and distributed-nDDE, both with smaller architectures, are however able to match the true variables for nearly the whole period of $t = 0$ to 330 *days* (Fig. 6-8b), with only distributed-nDDE starting to getting out-of-phase after $t = 270$ *days* (Fig. 6-8c) at the end of the long prediction period. These results are corroborated by the time evolution of the RMSE and average cross-correlation for the three variables over the prediction period (Fig. 6-8d). From the progression of the time-averaged L_2 loss (here, the error between the variables from the closure-model-augmented NPZ system, and the true variables), the nODE performs either equally well or better than both discrete-nDDE and distributed-nDDE

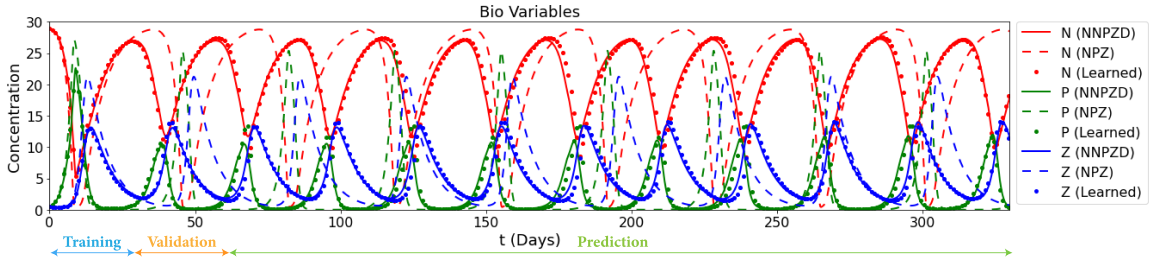
during training and validations periods (Fig. D-1c), however, it is not able to maintain long-term accuracy. We also notice very large spikes in the first half of the training regime, which are due to weights of the neural-networks taking values that lead to negative biology variables. As training progresses, we however don't observe this behavior anymore because the trainable weights starts to converge towards biologically feasible regimes. In conclusion, using a memory based closure for a low-complexity model can efficiently help emulate the high-complexity model dynamics.

As for experiments-2, we conducted a series of experiments-3a, to study the effect of changing the amount of past information incorporated in the neural closure models. In Fig. 6-6b, we show the variation of the average validation loss (time-averaged L_2 error) between the last epochs 300 to 350, for different delay-period lengths ($\tau_1 = 0$, and τ_2 varying in case of distributed-nDDE). In order to ensure statistical soundness of the results, $10 - 12$ repeats of the training were done, and the results were aggregated for each delay-period length (excluding the runs which diverged). We again find an optimal memory length for a specified architecture, however, with more and more runs failing to converge for longer delay period lengths. For the present system, estimates of delay period lengths to consider can be obtained from the time-scales of biological behaviors and adjustments, and from the decorrelation times of the biological state variables. Taking into account the limited effectiveness of a network architecture of finite size for capturing increasing information content, from Fig. 6-6b, we find an optimal delay period length to be around 1 *day*. We also conducted a series of experiments-3a to study the sensitivity to the various hyperparameters, and found similar trends (not shown here) as in experiments-1 & 2. For good performance, we further found that using a small enough time-step was critical as well as limiting the number of internal steps in the *dopri5* [134] time-integration scheme, while penalizing negative values in the loss function did not make much of a difference. Whenever multiple terms are present in the loss function enforcing different inherent properties of the system, they should be normalized (e.g. using non-dimensional variables) and given appropriate relative weights.

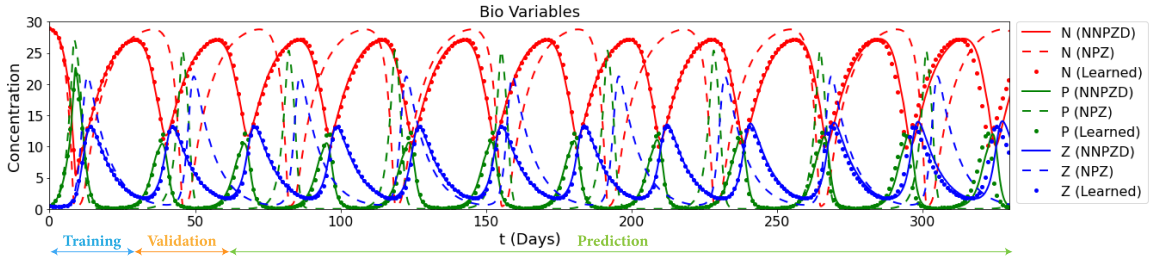
In general, the ecosystem ODEs are coupled with regional or global ocean mod-



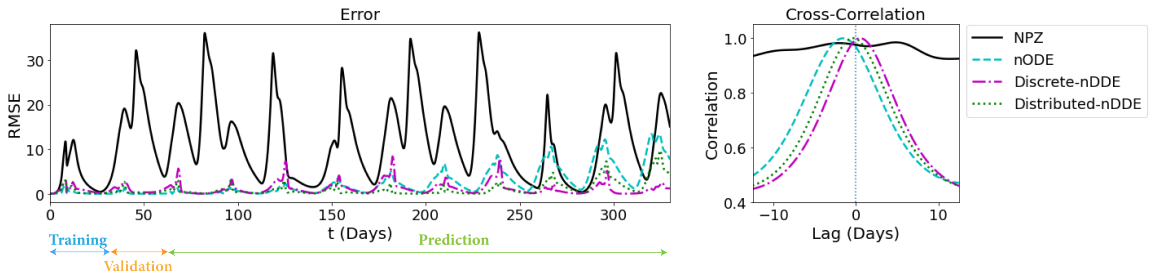
(a) Neural closure model with no-delays (nODE)



(b) Neural closure model with discrete-delays (Discrete-nDDE)



(c) Neural closure model with distributed-delays (Distributed-nDDE)



(d) Performance comparison of different neural closure models

Figure 6-8: Comparison of the biological variables from the learned NPZ model augmented with the three neural closure models (*dashed*), aggregated variables from the NNPZD model (ground truth; *solid*), and variables from the NPZ model (*dashed-dot*) at the end of training. For each neural closure, the training period is from $t = 0$ to 30 days, the validation period is from $t = 30$ to 60 days, while prediction period is from $t = 60$ to 330 days. *(a)*, *(b)*, *(c)*: different neural closure models; *(d)*: the *left* plot shows the evolution of root-mean-squared-error (RMSE), and the *right* plot shows the average cross-correlation (only for the prediction period) w.r.t. the ground truth. These results correspond to the architectures detailed in Table D.2.

eling systems, leading to advection-diffusion-reaction PDEs [62]. If highly complex ecosystem models are employed, a very large number of PDE state variables need to be solved for, rendering the computations very expensive. A large number of unknown parameter values as well as uncertain initial conditions then also need to be estimated, requiring specific methods [e.g. 84]. The available biogeochemical observations are not always sufficient for calibrating these many unknown parameters and for estimating initial conditions of high-complexity models. If the corresponding errors are large, this can lead to integrating models in the wrong dynamical regimes [e.g. 224]. Finally, in some applications, one is only interested in the aggregated state variables as the output, but cannot use low-complexity models because their dynamics are too inaccurate for the goals of the applications. Using neural closure models as shown here, one can increase the accuracy of the low-complexity models to match the response of high-complexity models (possibly up to models such as ERSEM [46]) without adding the computational burden of modeling all the intermediate biological states and processes, while reducing the effects of other uncertainties listed above. Results of our neural closures in 1-D PDEs is showcased next.

6.3.4 Experiments 3b: 1-D Marine Biogeochemical Models

For our final set of experiments, we extend the ODE models used in Experiments-3a (Sec. 6.3.3) to contain a vertical dimension (thus, 1-D) and vertical eddy mixing parameterized by the operator, $\partial/\partial z\,(K_z(z, M)\partial/\partial z(\bullet))$, where K_z is a dynamic eddy diffusion coefficient. A mixed layer of varying depth ($M = M(t)$) is used as a physical input to the ecosystem models. Thus, each biological state variable $B(z, t)$ is governed by the following non-autonomous PDE,

$$\frac{\partial B}{\partial t} = S^B + \frac{\partial}{\partial z}\left(K_z(z, M(t))\frac{\partial B}{\partial z}\right) , \tag{6.29}$$

where S^B are the corresponding biology source terms, which also makes it stiff. The dynamic depth-dependent diffusion parameter K_z is given by,

$$K_z(z, M(t)) = K_{z_b} + \frac{(K_{z_0} - K_{z_b})(\arctan(-\gamma(M(t) - z)) - \arctan(-\gamma(M(t) - D)))}{\arctan(-\gamma M(t)) - \arctan(-\gamma(M(t) - D))}, \tag{6.30}$$

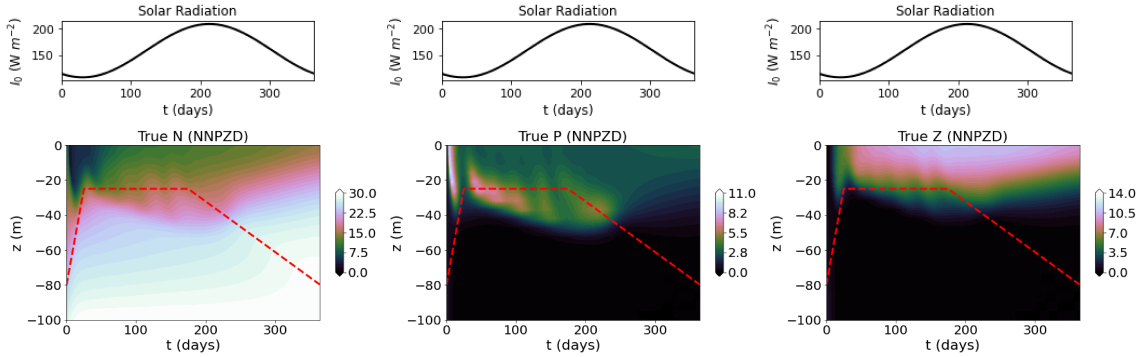
where K_{z_b} and K_{z_0} are the diffusion at the bottom and surface respectively, γ is the thermocline sharpness, and D is the total depth. The 1-D model and parameterizations are adapted from Eknes and Evensen, 2002 [4], and Newberger et. al., 2003 [3]. They simulate the seasonal variability in upwelling, sunlight, and biomass vertical profiles. The dynamic mixed layer depth, surface photosynthetically-available radiation $I_0(t)$, and biomass fields $B(z, t)$ are shown in Figure 6-9a. The radiation $I_0(t)$ and total biomass concentration, $T_{bio}(z, t)$, affect S^B and the initial conditions.

For these Experiments-3b, we consider 20 grid points in the vertical and use the *dopri5* [134] scheme with adaptive time-stepping. Data is stored at every time-step of $\Delta t = 0.1$ *days* for all our time-integration requirements. Solutions of aggregated states of the high-complexity 1-D NNPZD model (true data) and their absolute difference with the corresponding low-complexity 1-D NPZ model states are provided in Figs. 6-9a & 6-9b, respectively. For training our neural closure models for the 1-D NPZ model, we use the same training regiment as in Experiments-1, 2, & 3a (Secs. 6.3.1, 6.3.2, & 6.3.3). We note that in the 1-D NPZ model, the local mixing across depths occurs only due to the eddy diffusion term, and not to the biology source terms. Thus, we employ 1-D convolutional layers with receptive fields of size 1. We again use the custom layer, *BioConstrainLayer* (Sec. 6.3.3), to ensure that the sum of the biology source terms of the augmented 1-D NPZ model does not leave any new residual due to the neural closure terms. Along with this, we define a new custom layer, called *AddExtraChannels*, to add additional channels to the input of this layer. We add one for the depths at different grid points, and the other for the corresponding values of available sunlight for photo-chemical reactions ($I(z, t)$). The architectures details for the three closure models used are presented in Table D.2. For the nODE, we again employ a bigger architecture, and for the discrete-nDDE,

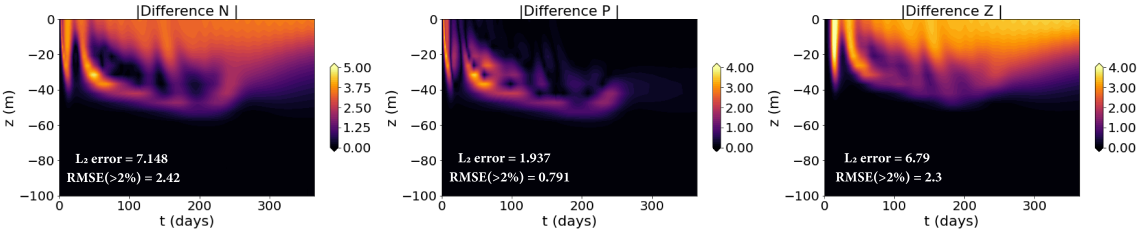
only four discrete delay values are used. The values of other hyperparameters are given in Sec. D.3.2. The training period ranges from $t = 0$ to 30 *days* and validation period from $t = 30$ to 60 *days*, both within the first season. The prediction period, however, ranges from $t = 60$ to 364 *days*: it is more than 10 times longer than the training period and involves the four seasons. Together, the three periods span a full year. For loss function, we combine the L_2 errors, considering all the biological states computed for individual depths, and then averaged over all the depths and times, $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\left(\frac{1}{N_z}\sum_{k=1}^{N_z=20}\left(\sqrt{\sum_{B\in\{N,P,Z\}}|B^{pred}(z_k,T_i) - B^{true}(z_k,T_i)|^2}\right)\right)$.

The performance of the three neural closure models augmenting the 1-D NPZ model is evaluated after 200 epochs of training (the stochastic gradient descent nearly converges, see Fig. D-1d). The truth fields are the aggregated biology variables from the high-complexity 1-D NNPZD model (Eqs. 6.28 & 6.29) spanning training, validation, and prediction periods. Results are presented in Figure 6-9. We find again that despite the bigger architecture for the nODE case, it develops spurious oscillations around $t = 250$ *days*. The discrete-nDDE and distributed-nDDE, both with smaller architectures, however match well with the true variables for nearly the full year of simulation, about 10 months of which is future prediction. The distributed-nDDE performs slightly better than its counterpart. In Fig. 6-9, we also provide averaged error numbers for the baseline (Fig. 6-9b) and the different closure models, all of which improve the baseline. As in Experiments-3a, we again notice large spikes in the starting of the training regime, for the same reason as given earlier, and similar trends for hyperparameter sensitivity. We also found that the Experiments-3b were affected by the choice of loss function. For example, using L_2 error computed for each biological state vector (containing values for all the depths) and then averaging over the number of biological states and times deteriorated the quality of learning.

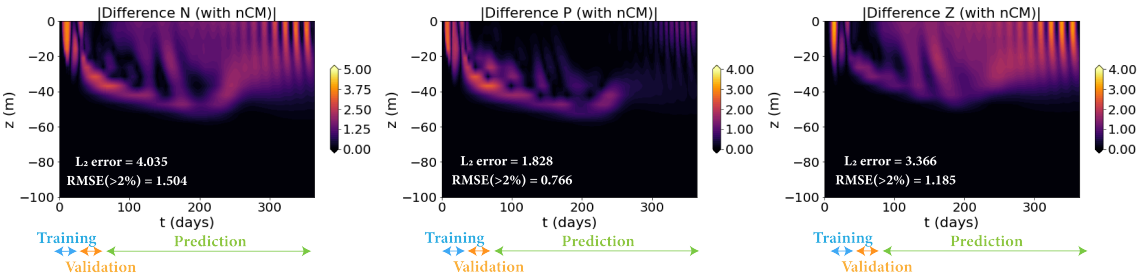
Despite the presence of complex physical processes and relatively large dimensions as compared to the previous experiments, the nDDEs closures were found to effectively match the high-complexity model and maintain long-term accuracy.



(a) Aggregated NNPZD model states (ground truth)

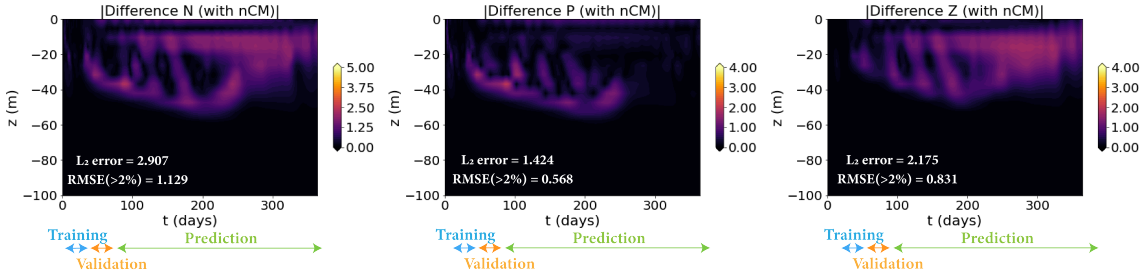


(b) NPZ model (without neural closure model)

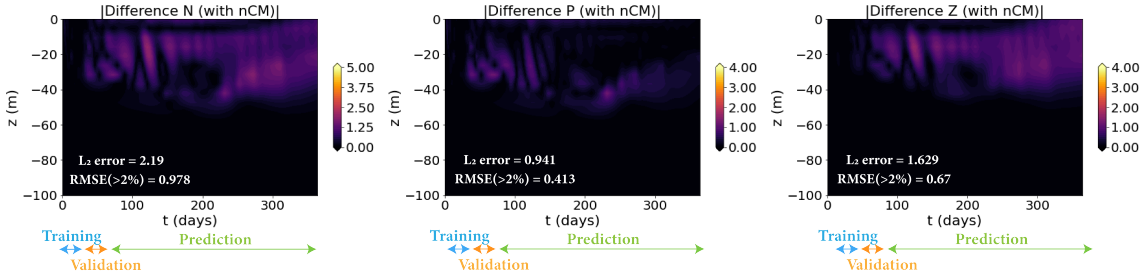


(c) NPZ model augmented with no-delay neural closure (nODE)

Figure 6-9: Comparison of the 1-D physical-biogeochemical PDE models used in Experiments-3b with and without closure models. Along with the parameter values mentioned in Figure 6-7, we consider: a sinusoidal variation in $I_o(t)$; linear vertical variation in total biomass $T_{bio}(z)$ from 10 $mmol\ N\ m^{-3}$ at the surface to 30 $mmol\ N\ m^{-3}$ at $z = 100\ m$; $K_{z_b} = 0.0864\ (m^2/day)$; $K_{z_0} = 8.64\ (m^2/day)$; $\gamma = 0.1\ m^{-1}$; and $D = -100\ m$, all adapted from [3, 4]. For the neural closure models, the training period is from $t = 0$ to 30 days, the validation period from $t = 30$ to 60 days, and the long future prediction period from $t = 60$ to 364 days. *(a): Top* plots show the yearly variation of solar radiation and the *bottom* plots the aggregated states from the NNPZD model (*ground truth*) overlaid with the dynamic mixed layer depth in *dashed red* lines. In the subsequent plots *(b)*, *(c)*, *(d)*, and *(e)*, we show the absolute difference of the different neural closure cases with the ground truth. For each case, we also provide the pair of time-averaged errors (see Fig. 6-4 for description). These results correspond to the architectures given in Table D.2. *(Cont.)*



(d) NPZ model augmented with discrete-delay neural closure (Discrete-nDDE)



(e) NPZ model augmented with distributed-delay neural closure (Distributed-nDDE)

Figure 6-9: Comparison of the 1-D physical-biogeochemical PDE models used in Experiments-3b with and without closure models. Along with the parameter values mentioned in Figure 6-7, we consider: a sinusoidal variation in $I_o(t)$; linear vertical variation in total biomass $T_{bio}(z)$ from 10 $mmol\ N\ m^{-3}$ at the surface to 30 $mmol\ N\ m^{-3}$ at $z = 100\ m$; $K_{z_b} = 0.0864\ (m^2/day)$; $K_{z_0} = 8.64\ (m^2/day)$; $\gamma = 0.1\ m^{-1}$; and $D = -100\ m$, all adapted from [3, 4]. For the neural closure models, the training period is from $t = 0$ to 30 days, the validation period from $t = 30$ to 60 days, and the long future prediction period from $t = 60$ to 364 days. *(a): Top* plots show the yearly variation of solar radiation and the *bottom* plots the aggregated states from the NNPZD model (*ground truth*) overlaid with the dynamic mixed layer depth in *dashed red* lines. In the subsequent plots *(b)*, *(c)*, *(d)*, and *(e)*, we show the absolute difference of the different neural closure cases with the ground truth. For each case, we also provide the pair of time-averaged errors (see Fig. 6-4 for description). These results correspond to the architectures given in Table D.2.

6.3.5 Computational Complexity

It is crucial to analyze complexity and in particular the cost of adding a neural closure model to a low-fidelity model. In this section, we analyze the computational complexity in terms of *flop* (floating point operations) count for evaluating the right-hand-side (RHS) of the low-fidelity models, and for the forward-pass of the neural closure models [225]. We will also comment on the training costs. The Burger's PDE considered for Experiments-1&2 (Secs. 6.3.1 & 6.3.2) has a nonlinear advection term. Hence, for the POD-GP ROM and the FOM, the upper *flops* is of the order of the square of number of resolved modes and of the spatial grid resolution, respectively. In general, for reaction terms and biogeochemical systems, the number of nonlinear parameterizations present are of the order of the number of components in the model. Hence, even for Experiments-3a (Sec. 6.3.3), the upper *flops* is of the order of the square of the number of biological components. For Experiments-3b (Sec. 6.3.4), the upper *flops* is also affected by the diffusion terms. Let the number of state variables in the low-fidelity models be $N \in \mathbb{N}$, thus the leading order computational complexity would be $\mathcal{O}(cN^2)$, where $c \in \mathbb{R}^+$ is some constant dependent on the numerical schemes used for spatial discretization, the exact functional form of the RHS, etc.

Now, when neural closure models are added to the low-fidelity models, the time integration requires a forward pass through the neural-network. This cost varies with the neural architecture and model type, here either a fully-connected or convolutional, and discrete-nDDE or distributed-nDDE, respectively. As observed in our experiments, using delays in the closure model enables us to use shallower networks, with a depth independent of the number of state variables (N). We also found that the width of the networks was similar to, or smaller than, N. In case of distributed-nDDEs, we observed that the width of the auxiliary network (g_{NN}) could be on an average nearly half the size of the main network (f_{NN}). Let the size of the hidden state for the RNN in discrete-nDDEs be $N_h \in \mathbb{N}$, and the number of neurons in the hidden layers of the main and auxiliary networks in the case of distributed-nDDEs be N_h and $N_h/2$, respectively, with $N_h \lesssim N$. It could be easily shown that the leading

order complexity for a single iteration of RNN would be $\mathcal{O}(N_h^2 + N_h N)$, which is due to the hidden and input state vectors being multiplied by the weight matrices, while the application of activation function would be $\mathcal{O}(N_h)$ only. As the number of discrete-delays in discrete-nDDEs are independent of N and $\mathcal{O}(1)$, it does not affect the complexity of the RNN. The complexity of the first hidden layer and/or the output layer of the deep neural-networks used in discrete-nDDEs and distributed-nDDEs (main network, f_{NN}) will be $\mathcal{O}(N_h N)$ and $\mathcal{O}(N_h(N + N_h/2))$, respectively, while for the remaining hidden layers, it will be $\mathcal{O}(N_h^2)$. Focusing on the integral of the auxiliary network (g_{NN}) over the delay period in distributed-nDDEs, if implemented efficiently, at every time-step, we only need to compute the integral twice over periods of size Δt, each adjacent to the ends of the present delay-period. We can add and subtract these integrals over Δt periods to compute the overall integral in a rolling window sense. Hence, the contribution to the computational cost by the auxiliary network would be $\mathcal{O}(N_h N/2)$ (for first hidden layer) and $\mathcal{O}(N_h^2/4)$ (subsequent hidden layers). Considering $D \in \mathbb{N}$ as the depth for all of the networks considered, the complexity for the forward pass through the discrete-nDDE closure is $\mathcal{O}(DN_h^2 + DN_h N)$, while for distributed-nDDE, it is $\mathcal{O}((3/2)N_h N + (7/4)DN_h^2)$. These costs were computed considering fully-connected layers, however, will only be cheaper in case of convolutional layers. Thus, the additional computational cost due to the presence of neural closure models is of similar or lower complexity than the existing low-fidelity model.

Estimating the computational cost/complexity of training in *flops* is not common because apart from time-integrating the forward model and adjoint equations, there are many other operations such as here: automatic differentiation through the neural networks; creation and use of interpolation functions; the integral to compute the final derivatives; the gradient descent step, etc. The overall cost also depends on the number of epochs needed for convergence. The present training cost is of course non-negligible, as with any supervised learning algorithm. However, in applications where one needs to repeatedly solve a low-fidelity model, investing in a one-time cost of training a neural closure model can later lead to accuracy close to that of the high-fidelity model with only a small increase in the computational cost of the low-fidelity

model.

6.4 Summary

We developed a novel, versatile, rigorous, and unified methodology to learn closure parameterizations for low-fidelity models using data from high-fidelity simulations. The Mori-Zwanzig formulation [160, 161, 162] and the presence of inherent delays in complex dynamical systems [226], especially biological systems [169, 170], justify the need for non-Markovian closure parameterizations. To learn such non-Markovian closures, our new *neural closure models* extend neural ordinary differential equations (nODEs; [33]) to neural delay differential equations (nDDEs). Our nDDEs do not require access to the high-fidelity model or frequent enough and uniformly-spaced high-fidelity data to compute the time derivative of the state with high accuracy. Further, it enables the accounting of errors in the time-evolution of the states in the presence of neural networks during training. We derive the adjoint equations and network architectures needed to efficiently implement the nDDEs, for both discrete and distributed delays, agnostic to the specifics of the time-integration scheme, and capable of handling stiff systems. For distributed-delays, we propose a novel architecture consisting of two coupled deep neural networks, which enables us to incorporate memory without the use of any recurrent architectures.

Through simulation experiments, we showed that our methodology drastically improves the long-term predictive capability of low-fidelity models for the main classes of model truncations. Specifically, our neural closure models efficiently account for truncated modes in reduced-order-models (ROMs), capture the effects of subgrid-scale processes in coarse models, and augment the simplification of complex biological and non-autonomous physical-biogeochemical models. Our first two classes of simulation experiments utilize the advecting shock problem governed by the Burger's PDE, with its low-fidelity models derived either by proper-orthogonal-decomposition Galerkin projection or by reducing the spatial grid resolution. Our third class of experiments considers marine biological ODEs of varying complexities and their physical-

biogeochemical PDE extensions with non-autonomous dynamic parameterizations. The low-fidelity models are obtained by aggregation of components and other simplifications of processes and parameterizations. In each of these classes, results consistently show that using non-Markovian over Markovian closures improves the accuracy of the learned system while also requiring smaller network architectures. Our use of the known-physics/low-fidelity model also helps to reduce the required size of the network architecture and the number of time samples for the training data. We also outperform classic dynamic closures such as the Smagorinsky subgrid-scale model. These results are obtained using stringent evaluations: we compare the performance of the learned system for the training period (during which high-fidelity data snapshots are used for training) and validation period (during which hyperparameter tuning occurs) as often done, but we also compare it for much longer-term future prediction periods with no overlap with the preceding two. We even consider a prediction period reaching 10 times the length of the training/validation period, thus successfully demonstrating the extrapolation capabilities of nDDE closures.

In our experiments, we find that just using a few numbers of discrete delays might perform equally well or better than using a distributed delay which involves an integral of the state variable over a delay period. We provide a plausible explanation of this counter-intuitive observation using the data processing inequality from information theory. We also show that there exists an optimal amount of past information to incorporate for a specified architecture and the relevant time-scales present in the dynamical system, thus indicating that neither too little nor too much past information is helpful. Finally, a computational complexity analysis using *flop* (floating point operation) count proves that the additional computational cost due to the presence of our neural closure models is of similar or lower complexity than the existing low-fidelity model.

The present work provides a unified framework to learn non-Markovian closure parameterization using delay differential equations and neural networks. It enables the use of the often elusive Mori-Zwanzig formulation [160, 161, 162] in its full glory without unjustified approximations and simplifications. Our nDDE closures are not

just limited to the shown experiments, but could be widely extended to other fields such as control theory, robotics, pharmacokinetic-pharmacodynamics, chemistry, economics, biological regulatory systems, etc.

Addendum

As an extension to the above work, in appendix E, we develop the methodology to seamlessly estimate the optimal delay-period length for distributed-nDDEs, instead of brute-force tuning as a hyperparameter. We propose to learn the optimal delay-period length from data along with the other trainable neural-network weights. We evaluate the performance of our methodology using a series of experiments consisting of a two-variable system with known delay, and the advecting shock problem governed by the Burger's equation (section 6.3.2).

Also, figure 6-10, made by combining elements from figures 6-2 & 6-9 to represent our *neural closure models* framework, was selected for the cover of *Proceedings of the Royal Society A*, August 2021 edition.

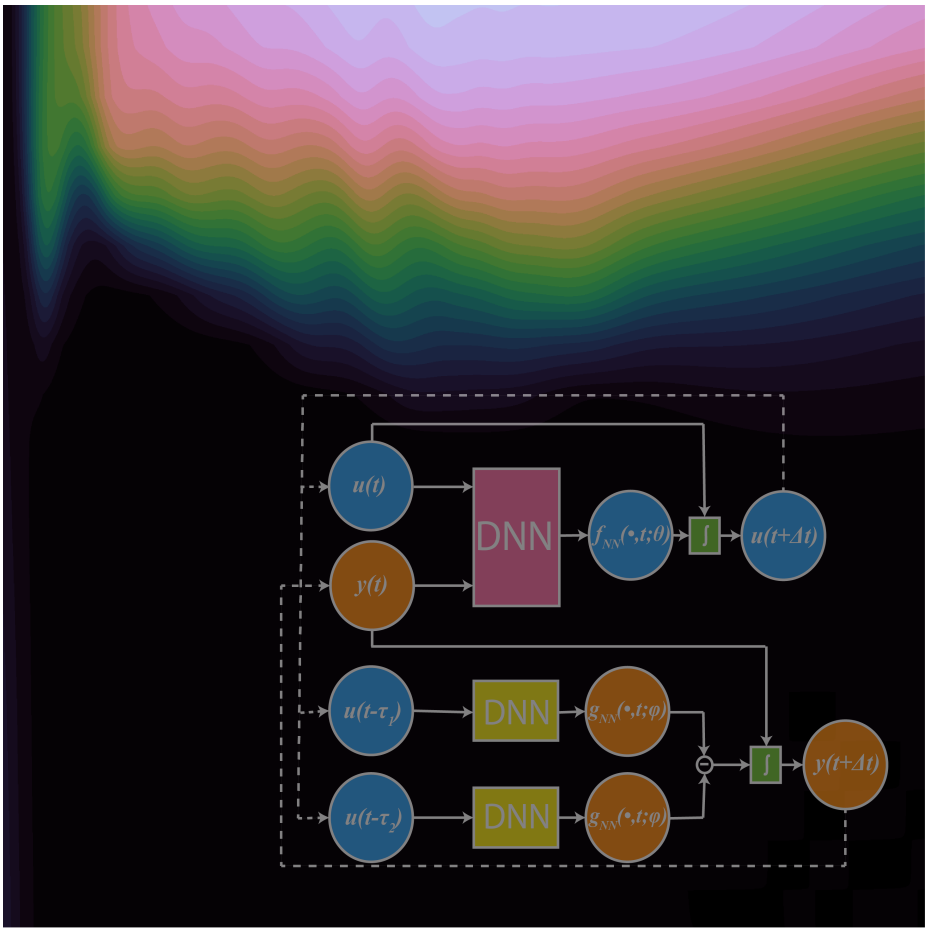


Figure 6-10: The background is a spatio-temporal zooplankton field, simulated using a complex nonlinear 5-component 1-D physical-biogeochemical model. Seasonal variability is forced through the surface photosynthetically-available radiation and mixed layer depth, each of which vary in time. The 5-component model is one of the dynamical systems used to illustrate our novel neural closure modeling. Overlaid on the zooplankton field is the graphical representation of the time-discretized distributed neural delay differential equation (Distributed-nDDE). The blocks labeled DNN and the integral symbol represent any deep neural-network architecture and time-integration scheme. Appeared on the cover of *Proceedings of the Royal Society A*, August 2021 edition.

Chapter 7

Generalized Neural Closure Models with Interpretability

Complex dynamical systems are used for predictions in many domains. However, due to computational cost constraints, models are often truncated, coarsened, or aggregated. As the neglected and unresolved terms become important, the utility of model predictions diminishes. There is a great deal of research on methods to model the missing dynamics and, with the advent of machine learning, there has been a renewed interest to learn the missing dynamics in a data-driven fashion [159, 156, 158, 157, 2, 227, 228]. Such techniques that express the missing dynamics as functions of modeled state variables and parameters are referred to as closure models.

The need for closure modeling in dynamical systems arises for a variety of reasons, ranging from computational cost considerations, preference of simpler models over complex ones due to overparameterization, or just lack of scientific understanding of processes involved in the system of interest. The simpler or the known model is often called a low-fidelity model, while the complex counterpart or the real-world data is called high-fidelity model/data. Encompassing various scenarios, low-fidelity models could be categorized into three categories: 1. Reduced order models, in which the original high-dimensional dynamical system is projected and solved in a reduced space. While it is computationally cheaper to solve the low-dimensional system, these models quickly start to accumulate errors due to the missing interactions with

the truncated dimensions [15, 14, 12]; 2. Coarse resolution models, in which we only resolve the scales of interest. In these cases, the neglected and unresolved scales, along with their interactions with the resolved ones, are important and lead to unintended and unacceptable effects at global scales [180, 179, 183, 181, 182]; 3. Simple models, which are used due to incomplete understanding of processes and interactions, leading to a gross and incorrect approximation of the real-world phenomena [39, 186, 185].

In our recently published work [229], which also acts as a precursor to the present study, we developed a novel neural delay differential equations (nDDEs) based framework to learn closure parameterizations for low-fidelity models using data from high-fidelity simulations to increase the long-term predictive capabilities of these models, called *neural closure models*. The need for using time-delays in closure parameterizations is deep rooted in the presence of inherent delays in real-world systems [169, 170], and theoretical justification from the Mori-Zwanzig formulation [161, 162, 160, 163]. Using nDDEs for closure modeling has a number of advantages such as allowing for the use of smaller architectures and accounting for the accumulation of numerical time-stepping error in the presence of neural-networks (NNs) during training. Additionally, nDDEs are agnostic to the time-integration scheme, they handle unevenly-spaced training data, and have good performance over prediction periods much longer than the training or validation periods. Moreover, when it comes to the melding of computational physics and machine learning (scientific machine learning; SciML [37]), there are other desirable properties as well, such as, generalization over different computational grid resolutions, initial conditions, boundary conditions, domain geometries, physical or problem specific parameters, and interpretability. There are a number of approaches in the field of SciML which attempt to address these properties of interests. However, on average, they are only able to address a subset of these, especially for partial differential equations (PDEs) based dynamical systems. This is often the case because NNs are typically used with the discretized ordinary differential equation (ODE) form of the corresponding PDEs, which makes it inherently difficult to generalize to changes in boundary conditions, domain geometry, and computational grid. Recently, a few studies have taken steps at addressing this drawback. Sirignano

et al. [227] augments the underlying PDE with a neural network, however they only learn a Markovian closure. The inputs to the neural network include the state, its spatial derivatives, and a fixed number of neighbouring grid points. They also provide an accompanying discrete adjoint PDE for efficient training. Saha *et al.* [228] uses a radial basis functions based collocation method to allow for mesh-free embedding of NNs, however, it also only learns a Markovian closure, does not take into account the accumulation of time-integration error, and lacks interpretability.

In the present study, we propose an unified approach based on neural partial delay differential equations (nPDDEs) which augments low-fidelity models in their PDE forms with both Markovian and non-Markovian closures parameterized with NNs. The input to each of the closure terms could potentially consist of the modeled states, their spatial derivatives, problem specific parameters, etc. This is followed by discretization using the desired numerical schemes. The melding of the low-fidelity model and NNs in the continuous spatio-temporal space automatically allows for generalizability to computational grid resolution, boundary conditions, initial conditions, and also provides interpretability. We refer to our new framework as *generalized* neural closure models (gnCM), and it is extendable to any popular numerical method used in computational physics. Further, we also provide adjoint PDE derivations in the continuous form, thus allowing one to implement across differentiable and non-differentiable computational physics codes, and also different machine learning frameworks. Through a series of experiments, we demonstrate the flexibility of our framework to learn closures either in an interpretable fashion, a black-box fashion, or both simultaneously, using the prior scientific knowledge about the problem at hand. We also demonstrate the generalizability of our learned closures to changes in physical parameters relevant to the problem, grid resolution, initial conditions and boundary conditions. Our first class of simulation experiments use the advecting shock problem governed by the KdV-Burgers and the classic Burgers PDE. Our learned closure model finds missing terms, rediscovers the leading discretization error, and a correction to the non-linear advection term. We find that training on data corresponding to just a few combinations of grid resolution and Reynolds number is sufficient to ensure

that the learned closures are generalizable and outperforms the popular Smagorinsky subgrid-scale closure model. Our second class of experiments is based on ocean acidification models, where we learn the functional form of certain ambiguous biological processes and compensate for lack of complexity in simpler low-fidelity models. Finally, we comment on the computational advantages of our new gnCM framework.

7.1 Theory and Methodology

The functional form of the closure models representing the missing dynamics is derived by the Mori-Zwanzig formulation [161, 162, 160, 163], which proves it to be dependent on the time-lagged state dynamics. Along with this formulation, many chemical or biological systems are modeled assuming smooth concentration fields of state variables governed by PDEs with fluid flow advection and/or mixing, leading to advection-diffusion-reaction PDEs. Such PDE systems implicitly assume that information between state variables is exchanged instantaneously at any spatial location. In reality, however, there are often time delays for several reasons. First, changes in populations or reactions have non-negligible time scales. Such time delays are captured in more complex models by modeling intermediate state variables. In the case of lower complexity models, the time response can be made to approximate that of high-complexity models by explicitly introducing delays [169, 170]. Second, time delays arise due to the truncated modes and/or missing subgrid-scale processes. For all of these reasons, the need for memory-based closure terms is clearly justified to represent complex dynamical systems. The above arguments in favor of memory-based closure terms are thoroughly discussed in Gupta and Lermusiaux, 2021 [229].

The need for non-Markovian closure terms to augment low-fidelity models was motivated in the preceding paragraph. Furthermore, it is often the case that the low-fidelity model is additionally outright missing Markovian terms due to truncation or ambiguity in functional form of some of the terms. As a result, we can assume that the full closure model is actually a combination of Markovian and non-Markovian terms, where each could potentially be modeled using NNs. To help with interpretability

of the learned weights of the NNs of the closure models, we further assume that the closure terms will be dependent on the state variables, their spatial derivatives, and combinations of these belonging to a function library. The non-Markovian term is considered to have a finite time-delay (τ) associated with it. Given a continuous state vector comprising of N_s different states, $u(x,t) : \mathbb{R} \times [0,T] \rightarrow \mathbb{R}^{N_s}$, let us consider a dynamical system belonging to domain Ω of the following form,

$$\begin{aligned}\frac{\partial u(x,t)}{\partial t} = & \underbrace{\mathcal{L}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., x, t; \nu\right)}_{Low-Fidelity\ /\ Known\ Model} \\ & + \underbrace{\mathcal{F}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., x, t; \phi\right)}_{Markovian\ Closure\ Term} \\ & + \underbrace{\int_{t-\tau}^{t} \mathcal{G}_{NN}\left(u(x,s), \frac{\partial u(x,s)}{\partial x}, \frac{\partial^2 u(x,s)}{\partial x^2}, ..., x, s; \theta\right) ds}_{Non-Markovian\ Closure\ Term}, \quad x \in \Omega,\ t \geq 0, \\ u(x,t) = h(x,t),\ t \leq 0 \quad & \text{and} \quad \mathcal{B}(u(x,t)) = g(x,t),\ x \in \partial\Omega,\ t \geq 0\,,\end{aligned} \tag{7.1}$$

where $\mathcal{L}$, $\mathcal{F}_{NN}$, and $\mathcal{G}_{NN}$ are non-linear functions parameterized with ν, ϕ, and θ, respectively. ν are problem specific parameters associated with the physical/biological/chemical phenomenon of interest, while ϕ and θ are the NN weights. The operator $\mathcal{B}$ represents appropriate boundary conditions such as Dirichlet, Neumann, etc. which are needed to solve the system uniquely. Furthermore, we note that we have assumed a one-dimensional (1D) domain, however, the method is easily extendable to 2D and 3D domains.

7.1.1 Neural Partial Delay Differential Equations

The goal of the present study is to add both Markovian and non-Markovian terms to the low-fidelity models in their PDE forms. This results in what are known as partial delay differential equations (PDDEs; DDEs are a subclass of PDDEs, in the same fashion as ODEs are for PDEs). They are widely used in ecology, control theory,

biology and climate dynamics, to name a few application areas, and are especially useful in situations where both spatial and temporal evolution matter [230].

In this section we derive the theory and schemes for PDDEs parameterized using NNs and learned from data, called *neural* partial delay differential equations (nPDDEs). Without loss of generality, and for brevity, we limit ourselves to nPDDEs with a Markovian term and a non-Markovian term with distributed delays. The low-fidelity model could be considered to be absorbed in the Markovian closure term, and the presence of discrete delays is a special case of distributed delays. Hence, our nPDDE is of the form,

$$\begin{aligned}\frac{\partial u(x,t)}{\partial t} &= \mathcal{F}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., \frac{\partial^d u(x,t)}{\partial x^d}, x, t; \phi\right) \\ &\quad + \int_{t-\tau}^{t} \mathcal{G}_{NN}\left(u(x,s), \frac{\partial u(x,s)}{\partial x}, \frac{\partial^2 u(x,s)}{\partial x^2}, ..., \frac{\partial^d u(x,s)}{\partial x^d}, x, s; \theta\right) ds\,, \\ &\qquad\qquad x \in \Omega,\ t \geq 0, \\ u(x,t) = h(x,t),\ t \leq 0 \quad &\text{and} \quad \mathcal{B}(u(x,t)) = g(x,t) \quad x \in \partial\Omega,\ t \geq 0\,.\end{aligned} \tag{7.2}$$

As compared to PDEs, PDDEs require the specification of a history function ($h(x,t)$) for the initial conditions. $\mathcal{F}_{NN}(\bullet;\phi)$ and $\mathcal{G}_{NN}(\bullet;\theta)$ are two NNs parameterized by ϕ and θ, respectively. For generality, we assume the NNs to be functions of an arbitrary number of spatial derivatives, with the highest order defined by $d \in \mathbb{Z}^+$. We can rewrite the above equation 7.2 as an equivalent system of coupled PDDEs

with discrete delays,

$$\begin{aligned}
\frac{\partial u(x,t)}{\partial t} &= \mathcal{F}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., \frac{\partial^d u(x,t)}{\partial x^d}, x, t; \phi\right) + y(x,t)\ , \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x \in \Omega,\ t \geq 0\,, \\
\frac{\partial y(x,t)}{\partial t} &= \mathcal{G}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., \frac{\partial^d u(x,t)}{\partial x^d}, x, t; \theta\right) \\
&\quad - \mathcal{G}_{NN}\left(u(x,t-\tau), \frac{\partial u(x,t-\tau)}{\partial x}, \frac{\partial^2 u(x,t-\tau)}{\partial x^2}, ..., \frac{\partial^d u(x,t-\tau)}{\partial x^d}, x, t-\tau; \theta\right)\ , \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x \in \Omega,\ t \geq 0\,, \\
u(x,t) &= h(x,t),\ t \leq 0 \quad \text{and} \quad \mathcal{B}(u(x,t)) = g(x,t),\ x \in \partial\Omega,\ t \geq 0\,, \\
y(x,0) &= \int_{-\tau}^{0} \mathcal{G}_{NN}\left(h(x,s), \frac{\partial h(x,s)}{\partial x}, \frac{\partial^2 h(x,s)}{\partial x^2}, ..., \frac{\partial^d h(x,s)}{\partial x^d} x, s; \theta\right) ds\,.
\end{aligned} \tag{7.3}$$

Let us assume that high-fidelity data is available at M discrete times, $T_1 < ... < T_M \leq T$, and at $N(T_i)$ spatial locations $(x_k^{T_i} \in \Omega, \forall k \in 1, ..., N(T_i))$ for each of the times. Thus, we define the scalar loss function as, $L = \frac{1}{M}\sum_{i=1}^{M} \frac{1}{N(T_i)} \sum_{k=1}^{N(T_i)} l(u(x_k^{T_i}, T_i)) \equiv \int_0^T \frac{1}{M}\sum_{i=1}^{M} \int_\Omega \frac{1}{N(T_i)} \sum_{k=1}^{N(T_i)} l(u(x,t))\delta(x - x_k^{T_i})\delta(t - T_i) dx dt \equiv \int_0^T \frac{1}{M}\sum_{i=1}^{M} \frac{1}{|\Omega|} \int_\Omega \hat{l}(u(x,t))\delta(t - T_i) dx dt$, where $l(\bullet)$ are scalar loss functions such as mean-absolute-error (MAE), and $\delta(\bullet)$ is the Kronecker delta function. In order to derive the adjoint PDEs, we start with the Lagrangian corresponding to the above system,

$$\begin{aligned}
\mathbb{L} &= L(u(x,t)) + \int_0^T \int_\Omega \lambda^T(x,t) \left(\partial_t u(x,t) - \mathcal{F}_{NN}(\bullet, t; \phi) - y(x,t)\right) dx dt \\
&+ \int_0^T \int_\Omega \mu^T(x,t) \left(\partial_t y(x,t) - \mathcal{G}_{NN}(\bullet, t; \theta) + \mathcal{G}_{NN}(\bullet, t-\tau; \theta)\right) dx dt \\
&+ \int_\Omega \alpha^T(x) \left(y(x,0) - \int_{-\tau}^{0} \mathcal{G}_{NN}(h(x,t), \partial_x h(x,t), \partial_{x^2} h(x,t), ..., \partial_{x^d} h(x,t), x, t; \theta) dt\right) dx\,,
\end{aligned} \tag{7.4}$$

where $\lambda(x,t)$, $\mu(x,t)$ and $\alpha(x)$ are the Lagrangian variables. To find the gradients of $\mathbb{L}$ w.r.t. ϕ and θ, we first solve the following adjoint PDEs (for brevity we denote,

$\partial/\partial(\bullet) \equiv \partial_{(\bullet)}$, and $d/d(\bullet) \equiv d_{(\bullet)}$),

$$
\begin{aligned}
0 = & \frac{1}{M}\frac{1}{|\Omega|}\sum_{k=1}^{M} \partial_{u(x,t)}\hat{l}(u(x,t))\delta(t-T_k) \\
& - \partial_t \lambda^T(x,t) - \lambda^T(x,t)\partial_{u(x,t)}\mathcal{F}_{NN}(\bullet,t) + \sum_{i=1}^{d}(-1)^{i+1}\partial_{x^i}\left(\lambda^T(x,t)\partial_{\partial_{x^i}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right) \\
& - \mu^T(x,t)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta) + \sum_{i=1}^{d}(-1)^{i+1}\partial_{x^i}\left(\mu^T(x,t)\partial_{\partial_{x^i}u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right) \\
& + \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta) - \sum_{i=1}^{d}(-1)^{i+1}\partial_{x^i}\left(\mu^T(x,t+\tau)\partial_{\partial_{x^i}u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right) \ , \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x \in \Omega \ , \ t \in [0,T) \, , \\
0 = & -\lambda^T(x,t) - \partial_t \mu^T(x,t) \ , \quad x \in \Omega \ , \ t \in [0,T) \, ,
\end{aligned}
\tag{7.5}
$$

with initial conditions, $\lambda(x,t) = \mu(x,t) = 0, \ t \geq T$. The boundary conditions are derived based on those of the forward PDDE and they satisfy,

$$
\begin{aligned}
0 = & \sum_{i=0}^{d}\sum_{j=0}^{d-i-1}(-1)^{j+1}\partial_{x^j}\left(\lambda^T(x,t)\partial_{\partial_{x^{j+i+1}}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right) d_\theta \partial_{x^i}u(x,t) \\
& + \sum_{i=0}^{d}\sum_{j=0}^{d-i-1}(-1)^{j+1}\partial_{x^j}\left(\mu^T(x,t)\partial_{\partial_{x^{j+i+1}}u(x,t)}\mathcal{G}_{NN}(\bullet,t)\right) d_\theta \partial_{x^i}u(x,t) \\
& - \sum_{i=0}^{d}\sum_{j=0}^{d-i-1}(-1)^{j+1}\partial_{x^j}\left(\mu^T(x,t+\tau)\partial_{\partial_{x^{j+i+1}}u(x,t)}\mathcal{G}_{NN}(\bullet,t)\right) d_\theta \partial_{x^i}u(x,t) \ , \\
& \qquad\qquad\qquad\qquad\qquad\qquad x \in \partial\Omega, \ t \in [t,T) \, .
\end{aligned}
\tag{7.6}
$$

Details of the derivation of the above adjoint PDEs is in section F.1. Note, that the adjoint PDEs need to be solved backward in time, and one would require access to $u(x,t), \forall x \in \Omega, \ 0 \leq t \leq T$. In our current implementation, we create and continuously update an interpolation function using the u obtained at every time-step as we solve the forward model (equation 7.2). To be more memory efficient, we can use the method of *checkpointing* [216]. After solving for the Lagrangian variables,

$\lambda(x,t)$ and $\mu(x,t)$, we can compute the required gradients as,

$$
\begin{aligned}
d_\theta \mathcal{L} &= -\int_0^T \int_\Omega \mu^T(x,t) \partial_\theta \mathcal{G}_{NN}(\bullet, t; \theta) dx dt + \int_0^T \int_\Omega \mu^T(x,t) \partial_\theta \mathcal{G}_{NN}(\bullet, t-\tau; \theta) dx dt \\
&\quad - \int_\Omega \mu^T(x,0) \int_{-\tau}^0 \partial_\theta \mathcal{G}_{NN}(h(x,t), \partial_x h(x,t), \partial_{xx} h(x,t), x, t; \theta) dt dx \,, \\
d_\phi \mathcal{L} &= -\int_0^T \int_\Omega \lambda^T(x,t) \partial_\phi \mathcal{F}_{NN}(\bullet, t; \phi) dx dt \,.
\end{aligned}
\tag{7.7}
$$

Finally, using any stochastic gradient descent algorithm, we can find the optimal values of the weights ϕ and θ.

Furthermore, for interpretability – especially for the Markovian closure term – we can use a simple NN architecture with no hidden layers and linear activation. However, non-linearity can be introduced by having input features which are combinations of the states and their derivatives belonging to a function library. This will be equivalent to a linear combination of the input features. Along with this, a L_1 regularization on the NN weights, and pruning below a threshold could help promote sparsity. Although this approach may seem similar to SINDy [24, 25, 27], it is very different because it accounts for accumulation of time-integration errors during training and does not require training data to be rich enough to allow for the computation of temporal and spatial derivatives.

The forward model (equation 7.1 or 7.2) and the adjoint PDEs (equation 7.5) are discretized and integrated using numerical schemes [5], such as, finite differences, collocation methods, etc. Our approach, where we augment the NN based closures first followed by numerical discretization, ensures that the burden of generalization over boundary conditions, domain geometry, and computational grid resolution, along with computing the relevant spatial derivatives is handled by the numerical schemes, and not by the learned NNs. This also automatically makes the learning only dependent on local features and affine equivariant, similar to numerical schemes.

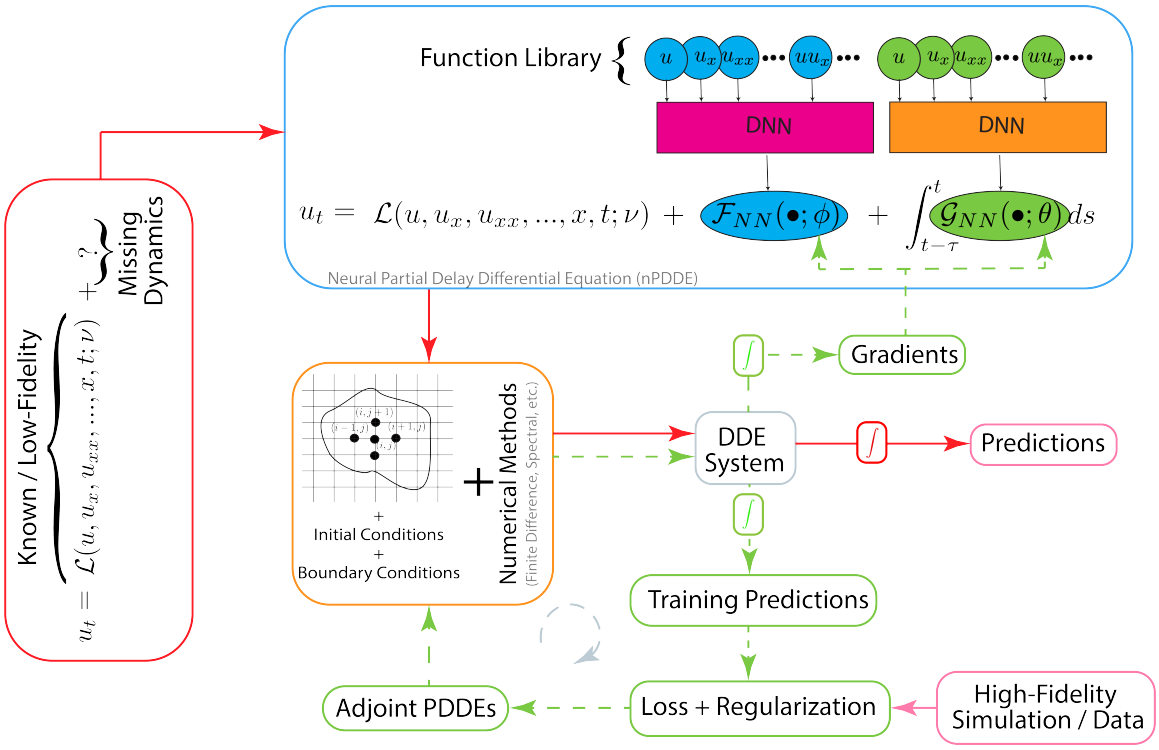


Figure 7-1: Overview of the *generalized* neural closure models (gnCM) framework. The blocks labeled *DNN* represent any deep neural-network architectures. The block labeled $\int$ symbolizes any time-integration scheme. DDE stands for delay differential equation.

7.2 Application Results and Discussion

We now evaluate the performance and advantages of our new closure modeling framework (gnCM) in terms of generalizabity over grid resolutions, boundary and initial conditions, and problem specific parameters. Our experiments will encompass various scenarios which require closure, and we will also attempt to interpret the learned closures.

Our training and evaluation protocol is similar to that established in Gupta and Lermusiaux, 2021 [229]. In all our experiments, the training data is regularly sampled in both space and time from the high-fidelity simulations, however, this is not a requirement. We use performance over the validation period (past the period for which high-fidelity data snapshots are used for training) to fine-tune various training related hyperparameters. The final evaluation is based on continuous evolution through the training and validation periods, followed with longer-term future predictions. We also compare the learned closure with the known true model, whenever available. In the rest of the paper, for all the figure, table, and section references prefixed with "SI-", we direct the reader to the *Supplementary Information*.

7.2.1 Experiments 1a: Advecting Shock - Model Ambiguity

Models for advecting shock are important to study various physical phenomena such as wind-driven surface waves. The Korteweg de Vries (KdV)-Burgers equation is often used in the study of the weak effects of dispersion, dissipation, and non-linearity in wave propagation [231]. In the first set of experiments, we consider a 1D domain where we only have the prior knowledge about the existence of the advection term,

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x}\,, \tag{7.8}$$

which acts as the low-fidelity model. The other dominant effects are unknown and need to be discovered. We assume these unknown effects to be mainly Markovian in nature and that they can be modeled using a linear combination from a library of

functions comprising of terms up to 3rd order spatial derivatives and arising in the generalized KdV-Burgers equation: $\left\{\frac{\partial^2 u}{\partial x^2}, \frac{\partial^3 u}{\partial x^3}, u\frac{\partial u}{\partial x}, u^2\frac{\partial u}{\partial x}\right\}$.

The high-fidelity model (truth) consists of two solitary waves colliding with each other, governed by the equation,

$$\frac{\partial u}{\partial t} = -6u\frac{\partial u}{\partial x} - \frac{\partial^3 u}{\partial x^3}\,, \tag{7.9}$$

with initial and boundary conditions given by,

$$\begin{aligned} u(x,0) &= 2\eta_1^2 \mathrm{sech}[\eta_1(x-x_1)] + 2\eta_2^2 \mathrm{sech}[\eta_2(x-x_2)], \\ u(-L,t) = 0,\ &\left.\frac{\partial u(x,t)}{\partial x}\right|_{x=L} = 0,\ \text{and}\ \left.\frac{\partial^2 u(x,t)}{\partial x^2}\right|_{x=L} = 0\,, \end{aligned} \tag{7.10}$$

where x_1 is the location, $2\eta_1^2$ is the amplitude, and $1/\eta_1$ is the width of the first soliton wave, whereas x_2 is the location, $2\eta_2^2$ is the amplitude, and $1/\eta_2$ is the width of the second soliton wave, initially. The analytical solution of the above system is given by,

$$u(x,t) = \frac{8(\eta_1^2-\eta_2^2)(\eta_1^2\cosh\theta_2 + \eta_2^2\sinh\theta_1)}{((\eta_1-\eta_2)\cosh(\theta_1+\theta_2) + (\eta_1+\eta_2)\cosh(\theta_1-\theta_2))^2}\,, \tag{7.11}$$

where $\eta_1 \geq \eta_2$, and θ_1 and θ_2 are given by,

$$\begin{aligned} \theta_1 &= \eta_1(x - x_1 - 4\eta_1^2 t)\ , \\ \theta_2 &= \eta_1(x - x_2 - 4\eta_2^2 t)\ . \end{aligned} \tag{7.12}$$

We choose $L = 10$, maximum time $T = 1.5$, $\eta_1 = 1.2$, $\eta_2 = 0.8$, $x_1 = -6.0$ and $x_2 = -2.0$. All the numerical solutions are obtained by using finite difference schemes. For the advection term, 2^{nd} order accurate upwind [232] was used, while all other terms and derivatives were discretized with 4^{th} order accurate central-difference. Furthermore, the *Vode* scheme [220] with adaptive time-stepping was used. Further, we employ a fine grid with $N_x = 200$ number of grid points in the $x-$direction in order to keep the discretization errors low. The comparison between the numerical solution of the low-fidelity model (equation 7.8) with the analytical solution of the high-fidelity

model (equations 7.9, 7.10 & 7.11) is provided in the figure 7-2. Both the models start from the same initial condition, however, their evolved solutions are drastically different from each other. In the case of the high-fidelity model, it can be observed that the two solitons interact elastically, i.e., their amplitudes and physical forms are unchanged before and after the interaction, however, they do experience a phase shift in their positions. On the contrary, in the case of the low-fidelity model, the two solitons do not even come close to interacting with each other.

For the gnCM, we only consider the Markovian term with a simple neural network with no hidden layer and only linear activation in the output layer, in-effect equivalent to a linear combination of the inputs. The training data consists of the analytical solution (equation 7.11) sampled at time intervals of 0.01 until time $t = 1.0$, with a validation period from $1.0 \leq t \leq 1.25$. In all the experiments, we use both $\mathcal{L}_1$ and $\mathcal{L}_2$ regularizations for the weights of the neural network, and prune them if their value drops below a certain threshold (only if weightage of $\mathcal{L}_1$ regularization is non-zero), in order to promote sparsity. The set of tuned hyperparameters used to generate the results presented next, are provided in the supplementary information, section F.2.2. Given the analytical solution, $\{u^{true}(x, T_i),\ -L \leq x \leq L\}_{i=1}^{M}$, the loss function is based on time and space averaged mean-absolute-error (MAE), $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\int_{-L}^{L}\frac{1}{2L}|u^{pred}(x, T_i) - u^{true}(x, T_i)|dx$. We perform 6 repeats of the experiment with exactly the same set of hyperparameters, and the learned model with mean and standard deviation of the weights is as follows,

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - (4.9680 \pm 0.0008)u\frac{\partial u}{\partial x} - (1.0105 \pm 0.0002)\frac{\partial^3 u}{\partial x^3}\,. \tag{7.13}$$

The true coefficients corresponding to the learned $u\frac{\partial u}{\partial x}$ and $\frac{\partial^3 u}{\partial x^3}$ terms are -5.0 and -1.0, respectively. The learned closure is able to recover the true model, and the slight discrepancy in the learned coefficients is to compensate for the very small discretization error. To illustrate this, we compare the root-mean-square-error (RMSE), $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\sqrt{\sum_{j=1}^{N_x}\frac{1}{N_x}(u^{pred}(x_j, T_i) - u^{true}(x_j, T_i))^2}$, of the learned closure and the true model solved using the same numerical schemes. The RMSE (mean and standard

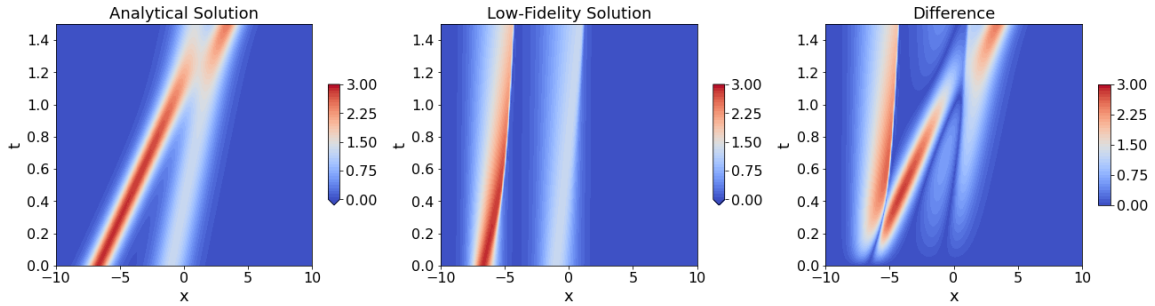


Figure 7-2: Comparison of the numerical solution of the KdV-Burgers equation with only the advection term (equation 7.8; low-fidelity model; *middle plot*), with the analytical solution corresponding to the equation with stronger advection and 3^{rd} order derivative term (equations 7.9, 7.10 & 7.11; high-fidelity model; *left plot*). The low-fidelity model is solved on a grid with $N_x = 200$ grid points, and the absolute difference between the two solutions is provided in the *right plot*.

deviation) obtained for the learned closure and the true model solved numerically is 0.0063 ± 0.0014 and 0.0251, respectively. Thus, on average, the learned closure leads to a smaller RMSE.

The learning was sensitive to the batch-time, and higher values were especially detrimental to convergence. This behavior is in general observed when the error between the low- and high-fidelity models is large. Using a smaller batch-size and regularization weightages lead to slightly different values of the learned coefficients. This is especially noted for the $u^2 \frac{\partial u}{\partial x}$ term, whose weight tends towards a non-zero value with a very small magnitude. In the current set of experiments, the learning framework is able to recover the known true model and, due to this, we do not additionally focus on demonstrating generalization over initial conditions, boundary conditions, and grid resolution.

7.2.2 Experiments 1b: Advecting Shock - Subgrid-scale Processes

In the second set of experiments we consider the classic form of the Burgers equation as the governing model,

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} + \nu\frac{\partial^2 u}{\partial x^2}\,, \quad 0 \leq x \leq L,\ t \in (0, T]\,, \tag{7.14}$$

where ν is the diffusion coefficient. The initial and boundary conditions are

$$u(x,0) = \frac{x}{1+\sqrt{\frac{1}{t_0}}\exp\left(Re\frac{x^2}{4}\right)}\,, \quad u(0,t) = 0, \quad \text{and} \quad \left.\frac{\partial u(x,t)}{\partial x}\right|_{x=L} = 0\,, \tag{7.15}$$

where Reynolds number, $Re = 1/\nu$ and $t_0 = \exp(Re/8)$. The analytical solution of the Burgers equation with the above mentioned initial and boundary conditions is given by,

$$u(x,t) = \frac{x/(t+1)}{1+\sqrt{\frac{t+1}{t_0}}\exp\left(Re\frac{x^2}{4t+4}\right)}\,. \tag{7.16}$$

However, when the discrete version of the above equation is solved numerically, the numerical solution incurs errors from three sources; 1. *Projection error*, which accounts for the fact that the exact solution is approximated using a finite number of degrees of freedom. This error cannot be avoided; 2. *Discretization error*, which accounts for the fact that partial derivatives which appear in the continuous problem are approximated on the computational grid using Finite Difference, Finite Volume, Finite Element (or other similar) schemes; 3. *Resolution error*, which accounts for the fact that the absence of some scales of the exact solution result in the evaluation of the non-linear flux function to be inexact, even if the discretization error is driven to zero [233].

To numerically solve the Burgers equation (7.14), we use a finite difference scheme. Specifically, we use $1st$ order accurate upwind for the advection term, 2^{nd} order

accurate central-difference for the diffusion term, and the *Vode* scheme for adaptive time-stepping. Thus, the leading order discretization error term is given by, $-\frac{\Delta x}{2}u\frac{\partial^2 u}{\partial x^2} + \mathcal{O}(\Delta x^2)$, where Δx is the uniform grid-spacing. The terms in $\mathcal{O}(\Delta x^2)$ contains spatial derivatives of order 3 and above. First, we only consider a Markovian closure, as a linear combination of a library of four terms, $\{\ \Delta x\left(\frac{\partial u}{\partial x}\right)^2, \Delta x^3\left(\frac{\partial^2 u}{\partial x^2}\right)^2, \Delta x^2\left(\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2}\right), \Delta x\left(u\frac{\partial^2 u}{\partial x^2}\right)\ \}$, out of which three are up to second degree combinations of $\frac{\partial u}{\partial x}$ and $\frac{\partial^2 u}{\partial x^2}$, and the fourth is the leading order discretization error term itself. Each of the terms is multiplied with appropriate powers of Δx, such that, the closure terms are dimensionally consistent with the other already existing terms in the Burgers equation. 4^{th} order accurate central and upwind finite-difference schemes [232] were used to compute the spatial derivatives in the Markovian closure, in order to eliminate additional sources of discretization error for our analysis. The training data consists of the analytical solution up until $T = 4.0$ solved in a domain of length $L = 1.25$ and saved at every 0.01 time-intervals, for three different combinations of N_x (number of grid points in $x-$direction) and Re. The chosen (N_x, Re) pairs, $\{(100, 50), (150, 750),\ \text{and}\ (200, 1250)\}$, are such that, the $-\frac{\Delta x}{2}u\frac{\partial^2 u}{\partial x^2}$ term is really the leading source of error. In every epoch, we parse through the training data of each of these pairs, selected in random order by sampling without replacement. We tune the hyperparameters based on performance in the training period ($0.0 \leq t \leq 4.0$) and the validation period ($4.0 \leq t \leq 6.0$), and these are provided in section F.2.2. The Markovian closure model is a simple neural network with no hidden layers and only linear activation in the output layer, in-effect equivalent to a linear combination of the inputs. Given the analytical solution, $\{u^{true}(x, T_i),\ 0 \leq x \leq L\}_{i=1}^{M}$, the loss function is once again the time and space averaged mean-absolute-error (MAE), $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\int_0^L \frac{1}{L}|u^{pred}(x, T_i) - u^{true}(x, T_i)|dx$. We perform 8 repeats of the same experiment with the tuned hyperparameters, and the learned model with mean and

standard deviation of the coefficients is as follows,

$$\begin{aligned}\mathcal{F}_{NN}\left(\Delta x\left(\frac{\partial u}{\partial x}\right)^2, \Delta x^3\left(\frac{\partial^2 u}{\partial x^2}\right)^2, \Delta x^2\left(\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2}\right), \Delta x\left(u\frac{\partial^2 u}{\partial x^2}\right); \phi\right) \\ = (0.133 \pm 0.017)\Delta x\left(\frac{\partial u}{\partial x}\right)^2 + (0.009 \pm 0.023)\Delta x^3\left(\frac{\partial^2 u}{\partial x^2}\right)^2 \\ - (0.323 \pm 0.022)\Delta x\left(u\frac{\partial^2 u}{\partial x^2}\right).\end{aligned} \tag{7.17}$$

We first compare the performance of the learned closure w.r.t. using the true leading discretization error term ($-\frac{\Delta x}{2}u\frac{\partial^2 u}{\partial x^2}$) as the closure itself. For both the cases, we evolve the Burgers equation with the respective closure terms up until $T = 8.0$ (beyond training and validation time-periods), for (N_x, Re) pairs in the 2D domain spanned by $50 \leq N_x \leq 200$ and $50 \leq Re \leq 1500$. In figure 7-4 we provide the $RMSE(> 2\%)$ error (see figure 7-3 for description). In the case of using the true leading discretization error term as the closure, it can be noted that increasing Re and lowering N_x values leads to instabilities in the solution which causes it to explode. On the contrary, in the learned closure case, even though it was not shown any training data in the high Re and low N_x regime, it is still able to lead to a stable solution, and, on average, performs better than its counterpart in the other regions of the (N_x, Re) domain. In order to interpret the learned closure, we rewrite it by substituting, $\frac{\partial}{\partial x}\left(u\frac{\partial u}{\partial x}\right) = \left(\frac{\partial u}{\partial x}\right)^2 + \left(u\frac{\partial^2 u}{\partial x^2}\right)$ in equation 7.17,

$$\begin{aligned}\mathcal{F}_{NN}\left(\Delta x\left(\frac{\partial u}{\partial x}\right)^2, \Delta x^3\left(\frac{\partial^2 u}{\partial x^2}\right)^2, \Delta x^2\left(\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2}\right), \Delta x\left(u\frac{\partial^2 u}{\partial x^2}\right); \phi\right) \\ = (0.133 \pm 0.017)\Delta x\frac{\partial}{\partial x}\left(u\frac{\partial u}{\partial x}\right) + (0.009 \pm 0.023)\Delta x^3\left(\frac{\partial^2 u}{\partial x^2}\right)^2 \\ - (0.456 \pm 0.012)\Delta x\left(u\frac{\partial^2 u}{\partial x^2}\right).\end{aligned} \tag{7.18}$$

Thus, the learned closure contains the $\Delta x\left(u\frac{\partial^2 u}{\partial x^2}\right)$ term with a coefficient of correct sign but slightly smaller value – in absolute value – in comparison to that of the true leading discretization error term. Along with that, the other significant term, $\Delta x\frac{\partial}{\partial x}\left(u\frac{\partial u}{\partial x}\right)$, corresponds to a first order Taylor series correction to the non-linear

advection term, and could potentially help with mitigating the resolution error highlighted earlier.

Next, keeping the same Markovian closure term formulation as earlier, we additionally introduce the non-Markovian closure term with inputs, $\{u, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \nu, \Delta x\}$, discretized using 4^{th} order finite-difference schemes, and the neural network (NN) architecture given in table F.1. The output of the NN is multiplied with $|u|$ to ensure that the contribution of the non-Markovian closure term is zero in the right-hand parts of the domain where the shock is yet to reach. As the non-Markovian closure term is non-linear, we do not explicitly make the inputs dimensionally consistent with other terms in the Burgers equation. The overall training and evaluation setup is kept the same as earlier, however, this time four pairs of (N_x, Re) were used such that all four combinations of high and low N_x and Re are contained in the training data. The chosen pairs were, $\{(50, 750), (200, 750), (50, 1250), (200, 1250)\}$. The tuned set of hyperparameters are provided in section F.2.2. The time-delay, $\tau = 0.075$, is based on the optimal-time delay established for the Burgers equation experiments in Gupta and Lermusiaux, 2021 [229]. We perform 7 repeats of the experiment with exactly the same set of tuned hyperparameters. The learned coefficients for the Markovian term are different than those in equation 7.17 due to the presence of the non-Markovian term, however, once again, the most weightage is given to the $\Delta x \left(\frac{\partial u}{\partial x}\right)^2$ and $\Delta x \left(u \frac{\partial^2 u}{\partial x^2}\right)$ terms. Upon inspection, the weights of the input layer of the NN in the non-Markovian term being multiplied with ν were consistently found to be particularly small ($\mathcal{O}(10^{-4})$), indicating that the learned closure is independent of ν. For one of the experiment runs, the performance for (N_x, Re) pairs in the 2D domain spanned by $50 \leq N_x \leq 200$ and $50 \leq Re \leq 1500$ is provided in figure 7-4, and compared with the popular Smagorinsky model used for subgrid-scale turbulence closure in large eddy simulations (LES). To the Burgers equation (7.14), this model introduces a dynamic turbulent eddy viscosity (ν_e) resulting in,

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} + \nu\frac{\partial^2 u}{\partial x^2} + \frac{\partial}{\partial x}\left(\nu_e \frac{\partial u}{\partial x}\right) , \tag{7.19}$$

where $\nu_e = (C_s \Delta x)^2 \left|\frac{\partial u}{\partial x}\right|$ and C_s is the Smagorinsky constant. As the rectangle formed by the training (N_x, Re) pairs is only a subset of the rectangle in which we evaluate the learned closure, we are testing both interpolation and extrapolation performance w.r.t. changing the physical parameter governing the model and grid resolution. It can clearly be noted that the learned neural closure model outperforms the Smagorinsky model. As claimed earlier, we expect the learned closure to be also generalizable over different boundary conditions. We tested this by modifying the boundary conditions. The analytical solution (equation 7.16) used in training corresponded to Neumann boundary conditions on the right edge of the domain. This was changed to a zero Dirichlet boundary condition. Furthermore, the length of the domain was decreased to $L = 1$, and $N_x = 50$ number of equally-spaced grid-points were used in our low-fidelity model with $Re = 1000$. Since no closed form analytical solution exists for the Dirichlet boundary conditions case, we solve the system with $N_x = 1000$ grid-points and use that as the true solution for comparing the performance of our learned closure. In figure 7-5, we can notice that the learned closure is able to keep the errors remarkably low throughout the time period encompassing training, testing, and prediction.

In general, the quality of learning was less sensitive to the batch-time hyperparameter, however, higher values led to more interpretable closures. Using lower-order finite-difference schemes for the closure inputs did not compromise on the performance of the learned closures, however it did lead to a decrease in interpretability. Sensitivity to other hyperparameters was similar to that observed in Experiments-1a.

7.2.3 Experiments 2a: Ocean Acidification - Model Ambiguity

Next, we will use our framework to determine the functional form of certain processes in ocean acidification (OA) models. These models help to study essential aspects of carbonate chemistry and biological production cycles, and their interplay with global warming. For this set of experiments, we will use a model similar to the Hadley Centre Ocean Carbon Cycle (HadOCC) model [123], where the biological part will consist of a modified version of four components (nutrients (N), phytoplankton (P),

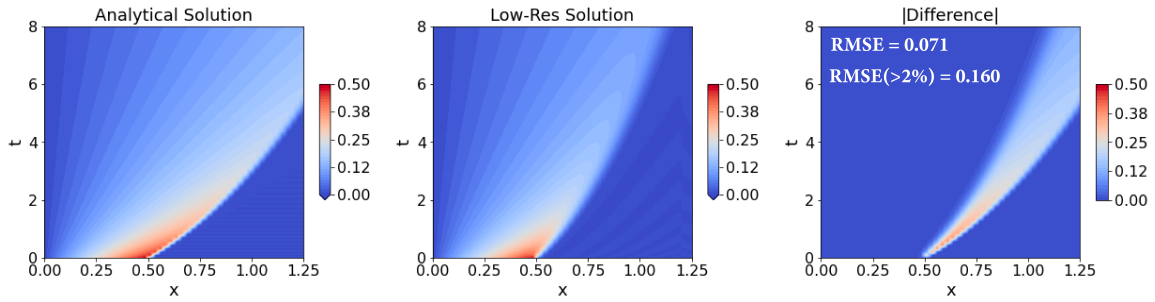


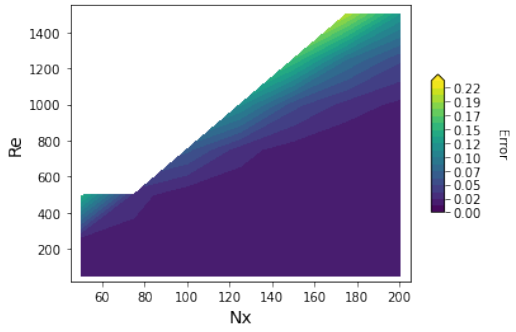
Figure 7-3: Comparison of the numerical solution of the Burgers equation (with $Re = 1000$) on a low-resolution grid (equations 7.14 & 7.15; low-fidelity model; *middle plot*), with its corresponding analytical solution (equation 7.16; high-fidelity model; *left plot*). The low-fidelity model is solved on a grid with $N_x = 50$ grid points, and the absolute difference between the two solutions is provided in the *right plot*. We also provide a pair of time-averaged errors, specifically: root-mean-squared-error (RMSE); and RMSE considering only the grid points where the error is at least 2% of the maximum velocity value, denoted by RMSE($> 2\%$).

zooplankton (Z), and detritus (D)) developed by Tian *et al.* [77] for the Gulf of Maine, along with dissolved inorganic carbon (DIC), total alkalinity (TA) for the carbonate part. The NPZD model is given by,

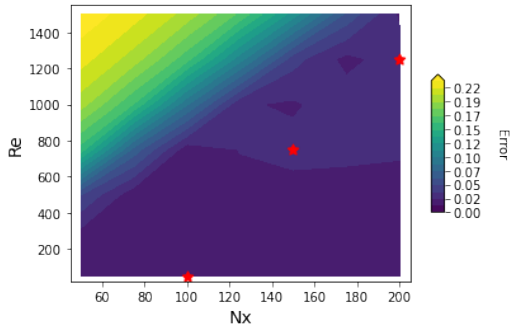
$$\begin{aligned}
\frac{dN}{dt} &= -U_P + \lambda G_Z + \varepsilon D \ , \\
\frac{dP}{dt} &= U_P - G_Z - m_P P \ , \\
\frac{dZ}{dt} &= \gamma G_z - M_Z(Z) \ , \\
\frac{dD}{dt} &= (1-\gamma-\lambda) G_Z + m_P P + M_Z(Z) - \varepsilon D \ ,
\end{aligned} \tag{7.20}$$

with U_P representing the phytoplankton growth, regulated by nitrogen limitation based on Michaelis-Menten kinetics ($f(N)$), and photosynthetically active radiation ($f(I)$), G_Z the zooplankton grazing, $M_Z(Z)$ the zooplankton mortality, given by,

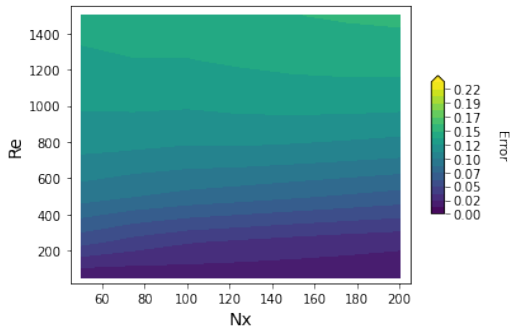
$$\begin{aligned}
U_P &= \mu_{max} f(N) f(I) P, \quad f(N) = \frac{N}{N + K_N}, \\
f(I) &= (1 - \exp(\alpha I/\mu_{max})) \exp(-\beta I/\mu_{max}) \\
I(z) &= I_0 \exp(-k_W z), \quad G_Z = \frac{g_{max} Z P^2}{P^2 + K_P^2} \ .
\end{aligned} \tag{7.21}$$



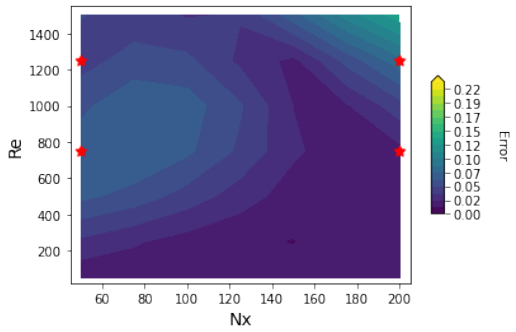
(a) Leading discretization error term as closure



(b) gnCM with only Markovian closure term

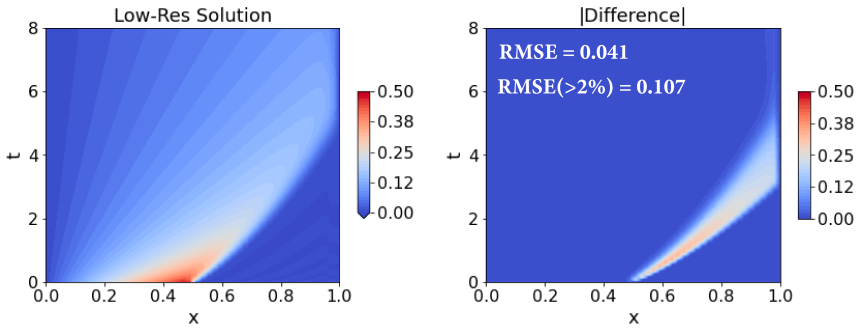


(c) Smagorinsky closure

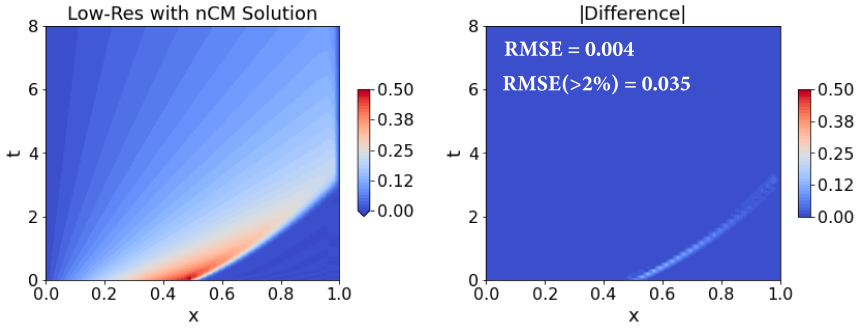


(d) gnCM

Figure 7-4: Performance of Burgers equation (equations 7.14 & 7.15) with different closure models evaluated for various (N_x, Re) pairs in the 2D domain spanned by $50 \leq N_x \leq 200$ and $50 \leq Re \leq 1500$. The error provided is the $RMSE(> 2\%)$ (see figure 7-3 for description) computed w.r.t. the corresponding analytical solutions (equation 7.16) for $0.0 \leq t \leq 8.0$ in a domain of length $L = 1.25$. *(a):* Leading discretization error term, $-\frac{\Delta x}{2} u \frac{\partial^2 u}{\partial x^2}$, as closure. The *white* region in the top-left denotes an unconverged numerical solution; *(b):* Learned generalized neural closure model (gnCM) with only the Markovian term; *(c):* Smagorinsky LES model with $C_s = 1.0$; *(d):* Learned gnCM with both Markovian and non-Markovian closure terms. The *red* $\star$'s mark the (N_x, Re) pairs used as training data.



(a) No closure



(b) gnCM

Figure 7-5: Solution of the Burgers equation with and without the learned generalized neural closure model (gnCM) for $Re = 1000$, a low-resolution grid ($N_x = 50$), and zero Dirichlet boundary condition on the right edge. For each case, we also provide the pair of time-averaged errors (see figure 7-3 for description).

In the above equations, the concentration of biological variables is in $mmol\ N\ m^{-3}$ (measured in nitrogen), z is depth, and the other parameters are given as: μ_{max} is the maximum growth rate of phytoplankton; K_N is the half-saturation constant; α and β are the light-growth slope and inhibition coefficient; I_0 is photosynthetically active radiation (PAR) at the sea surface and k_W is the attenuation coefficient of water; g_{max} is the zooplankton maximum grazing rate and K_P the half-saturation constant for zooplankton grazing; γ is the assimilation coefficient; m_z is the zooplankton mortality coefficient; m_p is the phytoplankton mortality coefficient; λ is the active respiration zooplankton expressed as a fraction of grazing; ε is the remineralization rate of detritus. The carbon in the system is coupled with the nitrogen by fixed carbon-nitrogen ratios, C_P, C_Z, and C_D,

$$\begin{aligned} \frac{d(DIC)}{dt} &= -C_P\frac{dP}{dt} - C_Z\frac{dZ}{dt} - C_D\frac{dD}{dt} - \gamma_c C_P U_P\ , \\ \frac{d(TA)}{dt} &= -\frac{1}{\rho_w}\frac{dN}{dt} - \frac{2\gamma_c C_P U_P}{\rho_w}\ , \end{aligned} \tag{7.22}$$

and neither DIC or TA has any effect on the biology because phytoplankton growth is not carbon limited. The last term in the DIC equation represents the precipitation of calcium carbonate to form shells and other hard body parts, which subsequently sink below the euphotic zone, also known as "hard flux". This flux is modeled to be proportional (and additional) to the uptake of carbon for primary production. Also, the chemistry dictates the decrease in total alkalinity by two molar equivalents for each mole of carbonate precipitated. In general, since TA is measured in $mmol\ kg^{-1}$ (or $\mu mol\ kg^{-1}$), we divide the right-hand-side (RHS) of the TA equation by the density of the sea-water (ρ_w). Moreover, the units of DIC concentration are $mmol\ m^{-3}$.

The above biological and carbonate models are often coupled with physical models to introduce both spatial and temporal components. For our experiments, we use a 1-D diffusion-reaction PDE with vertical eddy mixing parameterized by the operator $\partial/\partial z\,(K_z(z,M)\partial/\partial z(\bullet))$, where K_z is a dynamic eddy diffusion coefficient. A mixed layer of varying depth ($M = M(t)$) is used as a physical input to the OA models. Thus, each biological and carbonate state variable $B(z,t)$ is governed by the following

non-autonomous PDE,

$$\frac{\partial B}{\partial t} = S^B + \frac{\partial}{\partial z}\left(K_z(z, M(t))\frac{\partial B}{\partial z}\right) , \tag{7.23}$$

$$K_z(z, M(t)) = K_{z_b} + \frac{(K_{z_0} - K_{z_b})(\arctan(-\gamma_t(M(t) - z)) - \arctan(-\gamma_t(M(t) - D_z)))}{\arctan(-\gamma_t M(t)) - \arctan(-\gamma_t(M(t) - D_z))} , \tag{7.24}$$

where K_{z_b} and K_{z_0} are the diffusion at the bottom and surface, respectively, γ_t is the thermocline sharpness, and D_z is the total depth. The 1-D model and parameterizations are adapted from Eknes and Evensen, 2002 [4], and Newberger et. al., 2003 [3]. They simulate the seasonal variability in upwelling, sunlight, and biomass vertical profiles. The dynamic mixed layer depth, surface photosynthetically-available radiation $I_0(t)$, and biomass fields $B(z,t)$ are shown in figure 7-6. The radiation $I_0(t)$ and total biomass concentration, $T_{bio}(z,t)$, affects S^B and the initial conditions.

Often, for each biological process, there are as many functional forms as the number of biologists out there [40]. A set of parameter values and functional forms which might work in a particular part of the ocean may not work anywhere else. As an additional source of complexity, there may be seasonal variability in these functional forms. In the current set of experiments, we assume that we have only prior knowledge about the existence of a linear zooplankton mortality term ($M_Z(Z) = \frac{m_Z}{2}Z$), which forms our low-fidelity model. We further assume that the true zooplankton mortality consists of an additional quadratic dependence, $M_Z(Z) = \frac{m_Z}{2}(Z + Z^2)$, forming our high-fidelity model. Our Markovian closure term will belong to a linear combination of a library of popular mortality functions [40], $\{Z, Z^2, \frac{Z^2}{1+Z}, \exp Z\}$. Initializing the N state with the depth varying total biomass concentration and zero concentrations for the P, Z, and D states, we first do a one month spin-off of just the NPZD model without the diffusion term and a constant sea-surface solar radiation in order to determine the stable equilibrium of the biological states. These equilibrium states form the initial conditions for the respective states in the NPZD-OA model, and to initialize DIC we multiply the equilibrium state for N with the nitrogen-to-carbon ratio which is considered nearly equal to the value of C_P. TA is often assumed to

have a dependence on salinity and biological processes [126]. The contribution from salinity (S in PSU) is modeled using a linear relationship optimized for the Gulf of Maine, $TA = \begin{cases} (198.10 + 61.75S)/1000\ , & S < 32.34 \\ (744.41 + 44.86S)/1000\ , & S \geq 32.34 \end{cases}$ (Dr. Patrick J. Haley Jr., *pers. comm.*), while the biological impact is given by equation 7.22. We assume a stationary salinity profile described using a sigmoid function $S(z) = A + \frac{K-A}{(C+Q\exp(-Bz))^{1/\nu}}$ with $A = 31.4\ PSU$, $K = 32.8\ PSU$, $C = 1.0$, $Q = 0.5$, $B = 0.25$, and $\nu = 2.0$. Thus, we can initialize TA based on salinity and evolve it using equation 7.22 coupled with equation 7.23. In figure 7-6, we provide a year long simulation for the NPZD-OA model with linear and quadratic Z mortality terms, and easily notice the low Z concentration and enhanced P bloom in the later case. Values of the model parameters are provided in section F.2. We use a 2^{nd} order central difference scheme for the spatial discretization ($N_z = 20$), and *dopri5* [134] time integration scheme with adaptive time-stepping.

For the neural closure model – we only consider the Markovian term – we use once again a simple neural network with no hidden layers and linear activation in the output layer, which is in-effect equivalent to a linear combination of the inputs. The training data consists of the true/high-fidelity model solution sampled at time intervals of 0.1 day until time $t = 30\ days$, $\{\{B^{true}(z, T_i)\}_{B\in\{N,P,Z,D,DIC,TA\}}\}_{i=1}^{M}$. Using weight constraints for the output layer, we enforce biomass conservation in the N, P, Z, and D equations and couple with DIC and TA equations in the same fashion as that in the known system (equations 7.20 and 7.22). Architectural details are provided in the table F.1, and the tuned set of training hyperparameters are included in section F.2.2. We use a MAE based loss function, $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\int_0^D \frac{1}{D}\sqrt{\sum_{B\in\{N,P,Z,D,DIC,TA\}}\frac{1}{\sigma_B}|B^{pred}(z,T_i) - B^{true}(z,T_i)|}dz$. Here, σ_B's are hyperparameters to scale the importance of different state variables based on their magnitudes. After multiple hyperparameter tuning experiments, values of $\sigma_N = 1,\ \sigma_P = 0.25,\ \sigma_Z = 1,\ \sigma_D = 1,\ \sigma_{DIC} = 2,\ \sigma_{TA} = 0.1$, were found to aid in learning. In 7 repeats of the experiment with exactly the same hyperparameters, the learned models consisted of no contribution of the closure to the N, P, and TA equations,

while for the Z, D, and DIC equations the contributions were found – with mean and standard deviation – to be $(-0.02996 \pm 0.00014)Z^2$, $(0.03001 \pm 0.00013)Z^2$, and $(-0.05603 \pm 0.00136)Z^2$, respectively. For reference, the true contribution of the zooplankton quadratic mortality term to the Z, D, and DIC equations are given as $-0.02998Z^2$, $0.02998Z^2$, and $-0.05621Z^2$, respectively.

Multiple experiments were done to study the effects of hyperparameters, such as batch-time, batch-size, regularization factors, etc., and the convergence to the true model was the most severely compromised when increasing batch-time and changing the loss-scaling for individual state variables.

7.2.4 Experiments 2b: Ocean Acidification - Model Simplification

For the last set of experiments we consider the low complexity model to be the three-component NPZ model,

$$\begin{aligned} \frac{dN}{dt} &= -U_P + (1-\gamma)G_Z + m_P P + \frac{m_Z}{2} Z \ , \\ \frac{dP}{dt} &= U_P - G_Z - m_P P \ , \\ \frac{dZ}{dt} &= \gamma G_z - \frac{m_Z}{2} Z \ , \end{aligned} \tag{7.25}$$

coupled with the carbonate system using fixed carbon-nitrogen ratios, C_P, and C_Z,

$$\begin{aligned} \frac{d(DIC)}{dt} &= -C_P \frac{dP}{dt} - C_Z \frac{dZ}{dt} - \gamma_c C_P U_P \ , \\ \frac{d(TA)}{dt} &= -\frac{1}{\rho_w}\frac{dN}{dt} - \frac{2\gamma_c C_P U_P}{\rho_w} \ , \end{aligned} \tag{7.26}$$

and finally with the 1-D diffusion-reaction PDE given by equation 7.23. The high-fidelity model is the same as that used in Experiments-2a (section 7.2.3), where we also model the intermediate state of detritus, thus capturing additional processes such as remineralization and quadratic zooplankton mortality ($M_Z(Z) = \frac{m_Z}{2}(Z + Z^2)$). Since the NPZD-OA model resolves more processes, the concentrations of $N + D$ (aggre-

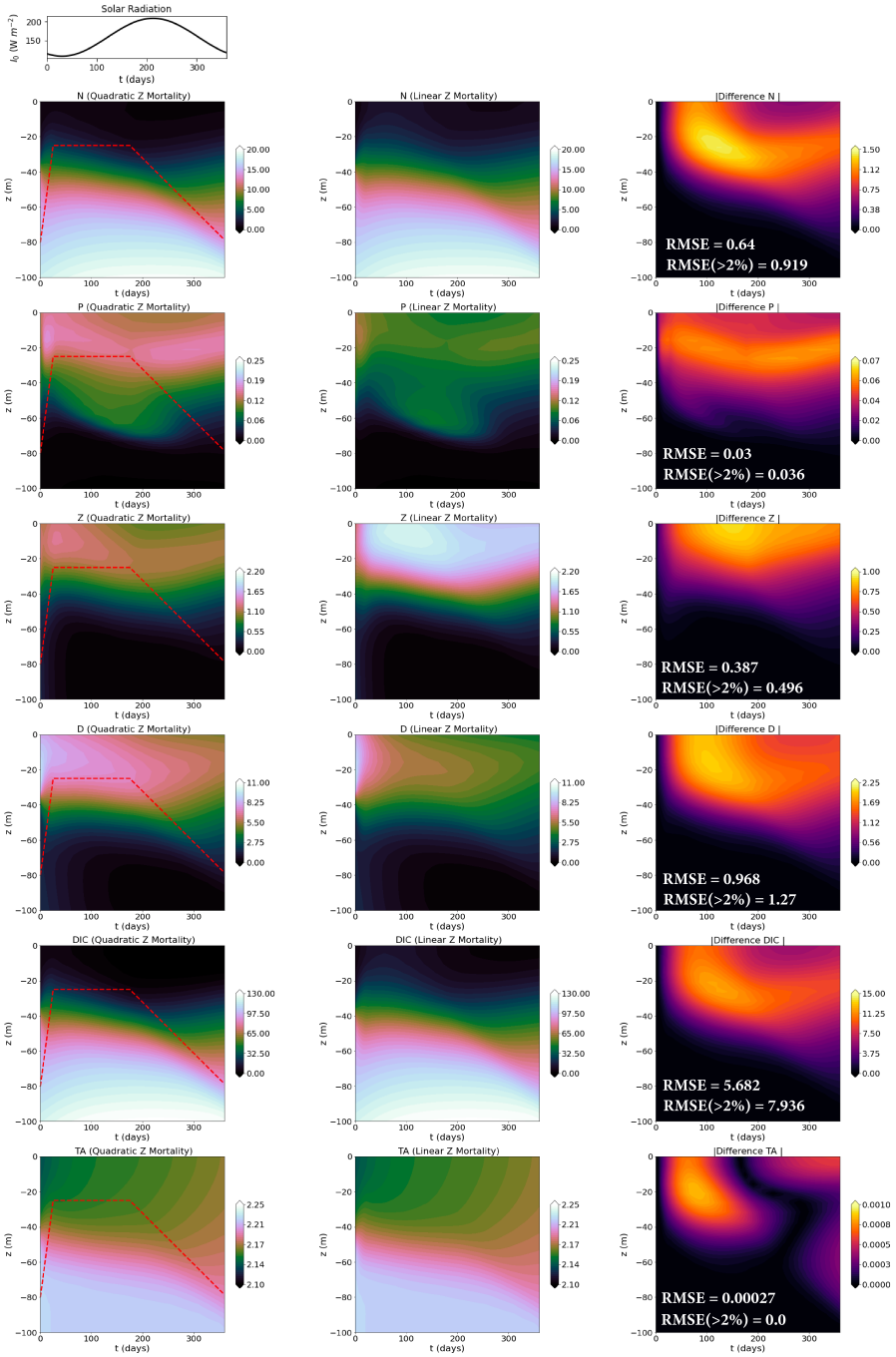


Figure 7-6: Solutions (concentrations vs. time in days; N, P, Z, D in $mmol\ N\ m^{-3}$, DIC in $mmol\ m^{-3}$, and TA in $mmol\ kg^{-1}$) of the ocean acidification model used in Experiments-2a, corresponding to different functional forms for the zooplankton mortality term. *Left-column:* The top plot shows the yearly variation of solar radiation and the subsequent plots depict the states from the NPZD-OA model with $M_Z(Z) = \frac{m_Z}{2}(Z + Z^2)$ (ground truth), overlaid with the dynamic mixed layer depth in dashed red lines; *Middle-column:* States from the NPZD-OA model with $M_Z(Z) = \frac{m_Z}{2}Z$ (low-fidelity); *Right-column:* Absolute difference between the corresponding states in the left- and middle- column. For each case, we also provide the pair of time-averaged errors (see figure 7-3 for description).

gated state), P, Z, DIC, and TA differ significantly from the N, P, Z, DIC, and TA of the NPZ-OA model, as shown in figure 7-7. The goal of the current experiments is to simultaneously learn the functional form of the zooplankton mortality terms using the Markovian closure term, and account for the missing intermediate state of detritus through the non-Markovian closure term. Once again, our Markovian closure consists of a linear combination of a library of popular mortality functions [40], $\{Z, Z^2, \frac{Z^2}{1+Z}, \exp Z\}$. Additionally, we use a deep neural network for the non-Markovian closure term, with $N(z,t)$, $P(z,t)$, $Z(z,t)$, and $I(z,t)$ as the input. The inclusion of photosynthetically active radiation, $I(z,t)$, makes the non-Markovian closure term non-autonomous. The architecture for the neural network used in the non-Markovian closure term is provided in table F.1. We do not include the states $DIC(z,t)$ and $TA(z,t)$ among the inputs in order to preserve one-way coupling between the biological and carbonate system. Along with this, biomass conservation and coupling of the carbonate system by converting to nitrogen (same fashion as in equations 7.25 and 7.26) is maintained in the non-Markovian closure terms by manipulating the channels of the output layer. On the other hand, in the Markovian layer, these constraints are imposed by constraining the weights of the output layer. To help with learning, we further impose the condition that the contribution of the Markovian closure term to the P equation is exactly equal to zero. See table F.1 for implementational details of these constraints.

The training data consists of solving the NPZD-OA model with $M(Z) = \frac{m_Z}{2}(Z + Z^2)$, and the solution sampled at time intervals of 0.1 day until time $t = 60$ *days*, $\{\{B^{true}(z,T_i)\}_{B\in\{N+D,P,Z,DIC,TA\}}\}_{i=1}^{M}$. Performance of the learned model in the validation interval of 60 *days* $\leq t \leq$ 120 *days* is used to tune the hyperparameters, provided in section F.2.2. We again use a MAE based loss function, $\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\int_0^D \frac{1}{D}\sqrt{\sum_{B\in\{N,P,Z,DIC,TA\}}\frac{1}{\sigma_B}|B^{pred}(z,T_i) - B^{true}(z,T_i)|}dz$, with $\sigma_N = 1$, $\sigma_P = 0.25$, $\sigma_Z = 1$, $\sigma_{DIC} = 2$, $\sigma_{TA} = 0.1$ (similar to those used in experiments-2a). A time delay of $\tau = 2.5$ *days* was used for the non-Markovian closure term based on the optimal delay value study performed in Gupta and Lermusiaux, 2021 [229]. In 9 repeats of the experiment with exactly the same set of hyperparameters, the

mean and standard deviation of the learned contribution of the Markovian closure term to the Z equation is given by, $(-0.03000 \pm 0.00067)Z^2$. For reference, the true contribution of the quadratic mortality term to the Z equation is $-0.02998Z^2$. Due to the weight constraints, the contribution of the Markovian closure term to other equations is exactly zero. We evaluate the performance of the learned neural closure model for long predictions, spanning over 1 year (365 *days*). The comparison with true/high-fidelity data for one of the experiments is provided in figure 7-7. Overall, the learned closure keeps the errors low throughout the 1 year time-period, apart from a slight increase observed for the OA states after ~ 200 *days*.

Multiple experiments were done to study the effects of hyperparameters, such as batch-time, batch-size, regularization factors, etc., and their effects were similar to that observed in previous experiments. The peculiar thing which was noticed in the current experiments resulted from using larger neural network architectures for the non-Markovian term, which led to the learned coefficients for the Markovian term having very high variability on repeats of the experiments with the same set of hyperparameters. This is probably because of the increased expressive power of the non-Markovian term, which over-shadows the significance of the learned Markovian term.

7.2.5 Computational Advantage

In Gupta and Lermusiaux, 2021 [229], through a flop-count analysis, we proved that the additional computational cost due to the presence of neural closure models is of similar or lower complexity than the existing low-fidelity model. However, in our current generalized framework, we have additional computational advantages. First, the size of the neural network architecture is completely independent of the number of discretized state variables, and only dictated by the number of local features to be used as inputs to the gnCM terms. Second, as the same neural networks gets applied locally at every grid points, it naturally is possible to use batches of the size of the number of grid points. It has been reported that larger batch sizes could lead to performance speed-ups in forward pass through neural networks during the

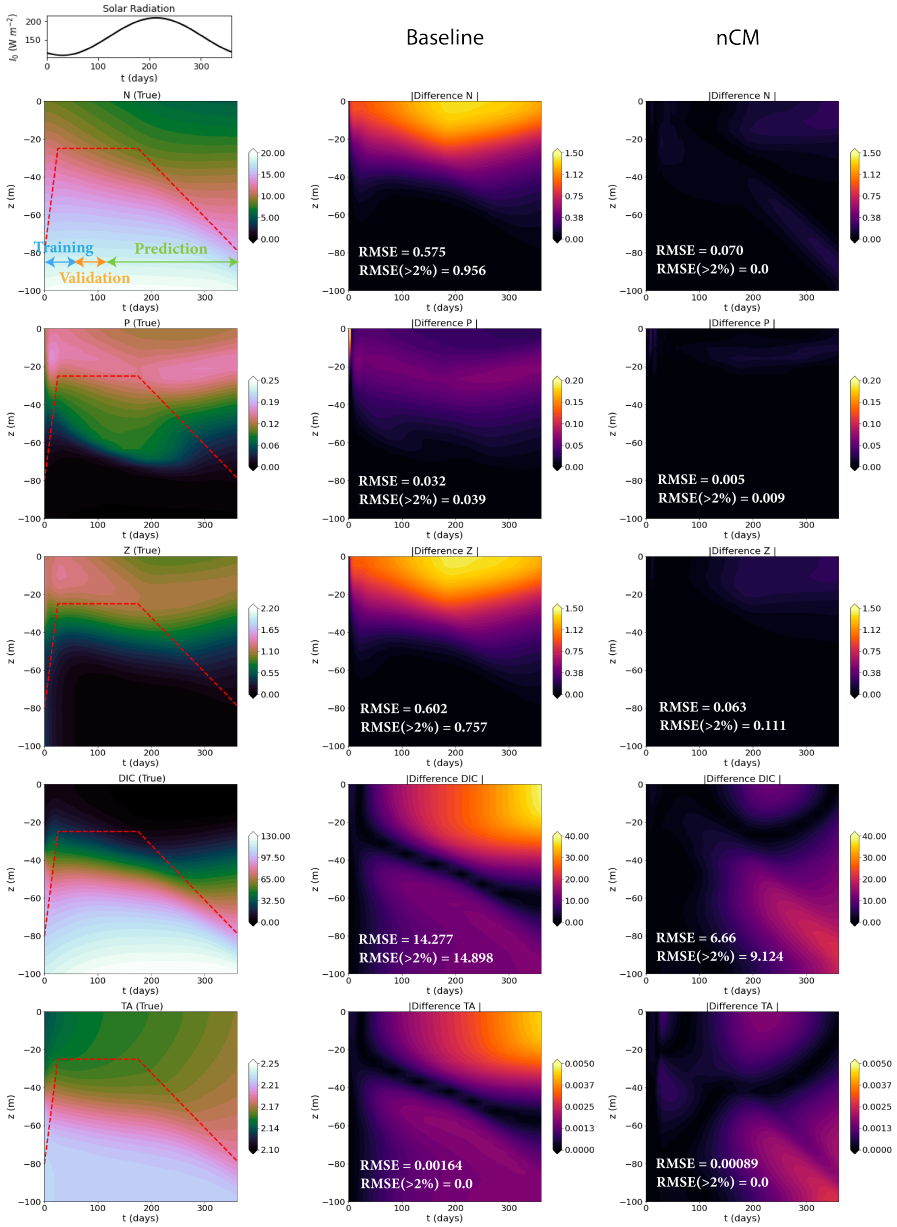


Figure 7-7: Comparison of the ocean acidification models used in Experiments-2b with and without closure models. The parameter values and concentration units are same as those provided in figure 7-6. For the generalized neural closure model (gnCM), the training period is from t = 0 to 60 *days*, the validation period from t = 60 to 120 *days*, and the future prediction period from t = 120 to 364 *days*. *Left-column:* The top plot shows the yearly variation of solar radiation and the subsequent plots depict the aggregated states from the NPZD-OA model with $M_Z(Z) = \frac{m_Z}{2}(Z + Z^2)$ (ground truth), overlaid with the dynamic mixed layer depth in dashed red lines; *Middle-column:* Absolute difference between the corresponding states in the left-column and those from the NPZ-OA model with $M_Z(Z) = \frac{m_Z}{2}Z$ (low-fidelity); *Right-column:* Absolute difference between the corresponding states from the low-fidelity model augmented with the learned gnCM and the ground truth. For each case, we also provide the pair of time-averaged errors (see figure 7-3 for description).

inference stage [234]. Estimating the leading flop-count order for training is non-trivial due to the presence of a number of operations ranging from time-integration of the forward model and adjoint PDEs; automatic differentiation through the neural networks; creation and use of interpolation functions; the integral to compute the final derivatives; the gradient descent step, etc. All these operations lead to training costs which are non-negligible. However, the generalizability of our learned neural closure over boundary conditions, initial conditions, domain, problem-specific parameters, etc. makes it easy to justify the one-time training cost.

7.3 Summary

In the present study, we propose a novel extention to neural closure models [229] which makes them readily generalizable over computational grid resolution, boundary conditions, initial conditions, and also provide interpretability. Our developed generalized neural closure models (gnCMs) are based on neural partial delay differential equations (nPDDEs) which augments low-fidelity models in their PDE forms with both Markovian and non-Markovian closure parameterized with neural networks (NNs). The melding in the continuous spatio-temporal space is then followed with numerical discretization. This ensures that the burden of generalization, along with computing the relevant spatial derivatives is carried by the numerical schemes, and not by the learned NNs. The space-time continuous form of the gnCMs also makes it very easy to interpret the learned closures. For efficient training, we also provide adjoint PDE derivations in the continuous form, thus enabling implementation across differentiable and non-differentiable computational physics codes, different machine learning frameworks, and allowing the method to be agnostic to the numerical methods being used. It further removes any requirements on the availability of regularly spaced training data in both space and time, and also accounts for errors in the time-evolution of the states in the presence of neural networks during training.

Through a series of experiments, we demonstrate the interpretability and generalizability of our learned closures. Our first class of simulation experiments use the

advecting shock problem governed by the KdV-Burgers and the classic Burgers PDE, where the low-fidelity models are either missing terms or contain errors due to unresolved subgrid-scale processes. When presented with a function library containing terms of spatial derivatives of different orders and their combinations, grid-resolution, and the Reynolds number as the input to the closure terms, our leaned closure models find the known missing terms, rediscovers the leading discretization error and a correction to the non-linear advection term, and also Reynolds number independence. We find that training on data corresponding to just $3-4$ combinations of number of grid points and Reynolds number with particular boundary conditions is sufficient to ensure that the learned closures are generalizable and outperforms the popular Smagorinsky subgrid-scale closure model. Our second class of experiments is based on one-dimensional, non-autonomous ocean acidification models, which couple physical, biological, and carbonate states, processes and interactions in the ocean. In our experiments, the low-fidelity models have ambiguity in the functional form of certain biological processes and lacks in complexity due to missing intermediate state. The learned closure models are simultaneously able to learn an interpretable functional form of the ambiguous biological process with the Markovian closure term, and account for the missing state with the non-Markovian term. In terms of computational advantage, our new framework naturally lends itself to batching across computational grid points during the forward pass through the NNs in the closure terms, thus leading to potential performance speed-ups.

Our present work allows one to learn both Markovian and non-Markovian closure parameterization based on NNs, and at the same time, tackles the issues of generalizabity and interpretability which are often the bottleneck when it comes to using machine learning for computational physics problems. The generalizability and interpretability properties also makes it easier to justify the often computationally expensive training stage, thus enabling wider adoption.

Chapter 8

Conclusions and Future Work

8.1 Summary of the Thesis

Advanced predictive models are commonly used for a variety of research and societal needs. However, due to the complexity of the real-world phenomena, the level of scientific understanding, and computational cost consideration, models are always missing some scales, processes, or variables, which limits the utility of their predictions. Thus, in this thesis, we developed novel Bayesian learning and deep learning methods to learn and discover missing dynamics in existing / low-fidelity dynamical system models.

First, we started with a Bayesian approach, which is inherently advantageous for melding observations with models, as it provides the ability to take into account rigorously all the existing prior knowledge in the learning process, and accompany with the associated uncertainty estimates. We built upon and drastically extend the approach developed by Lu and Lermusiaux 2014 & 2021 [10, 8] and Lin, 2020 [9] for the simultaneous estimation of states and parameters along with discrimination among candidate models in high-dimensional stochastic dynamical systems using sparse observations. However, often none of the candidate models is exactly equal to the true model, or the functional form is yet completely elusive to scientists. Is is nonetheless common for the candidate models to be compatible with each other. For example, only certain functional terms in a model are unknown, or there are competing formulations for the

terms, or low complexity models are embedded in higher complexity models. These situations were addressed in two novel ways: first, using special stochastic parameters to unify all the candidates into a single general model; second, parameterizing unknown functions using stochastic piece-wise polynomial functions, allowing us to search in an infinite candidate space. Our new methodology not only seamlessly and rigorously discriminated between existing models, but also extrapolated out of the space of models to discover newer ones. In all cases, the results were generalizable and interpretable, and our Bayesian estimations provided much more than maximum likelihood estimates: they predicted and updated the complete joint probability distribution of states, parameters, and models. All of this was achieved just at the cost of a single stochastic model simulation with parameter estimation. When the observations are not sufficiently informative to learn and eliminate all but one model, parameter value, or state variable field, our Bayesian learning provided the correct multi-modal probability distributions. Algorithmically, our rigorous PDE-based Bayesian learning framework combined the Dynamically Orthogonal (DO) equations [11, 12, 13, 14, 15] with the Gaussian mixture model (GMM) DO filtering algorithm [16, 17] for the simultaneous nonlinear, non-Gaussian inference of the states, parameters, and model equations.

We showcased the performance and applicability of our Bayesian model learning framework using both, identical-twin and real-world data experiments. Our identical-twin experiments consisted of lower-trophic-level marine ecosystem and fish models setup in a two-dimensional idealized domain with flow past a seamount representing upwelling due to a sill or strait. Experiments had varying levels of complexity due to different learning objectives and flow and ecosystem dynamics. The flow dynamics encompassed steady, chaotic, nonhydrostatic features, and was itself uncertain in some experiments. The learning objectives included state and parameter estimation, discriminating between functional terms and model complexity, learning unknown functional terms from scratch, and interdisciplinary learning. We also demonstrated smoothing backward in time. In the real-world data experiment, we configured a one-dimensional coupled physical-biological-carbonate model to simulate the state

conditions on the last day of the second Gulf of Mexico and East Coast Carbon (GOMECC-2) research cruise in the Gulf of Maine region. Using the observed ocean acidification data, we learned and discovered a salinity based forcing term added to the total alkalinity (TA) equation to account for changes in TA due to advection of water masses of different salinity caused due to precipitation, riverine input, and other oceanographic processes. Simultaneously, we also estimated the multidisciplinary states and an uncertain parameter. Crucially, we provided probability distributions for each learned quantity including the learned model functions in all of the learning experiments.

In Bayesian learning, obtaining an accurate and informative prior is imperative. Thus, we developed new theory and techniques to improve uncertainty quantification in multidisciplinary settings using the DO methodology [11, 12, 13, 14, 15] used in our Bayesian model learning framework. The developed techniques were aimed at accurately handling stochastic boundary conditions, complex geometries, advection terms, and to augment the DO subspace as and when needed to capture the effects of the truncated modes. Further, we also discussed mutual information based observation planning to answer what, when, and where to measure to best achieve the learning objectives in resource-constrained environments.

On the deep learning side, we developed a novel, versatile, rigorous, and unified methodology to learn time-delayed closure parameterizations for missing dynamics. The need for non-Markovian closure parameterizations was justified using the Mori-Zwanzig formulation [160, 161, 162] and the presence of inherent delays in real-world systems [226], especially biological systems [169, 170]. To learn such non-Markovian closures, our new *neural closure models* (nCMs) extended neural-ordinary-differential-equations [33] to neural-delay-differential-equations. We also developed a novel extension to the nCMs framework which renders it readily interpretable and generalizable over computational grid resolution, boundary conditions, and initial conditions. Our developed generalized nCMs are based on neural-partial-delay-differential-equations (nPDDEs) that augmented low-fidelity models in their original PDE forms with both Markovian and non-Markovian closure parameterized with neural networks (NNs).

The melding in the continuous spatiotemporal space was then followed by numerical discretization, which ensured that the burden of generalization, along with computing the relevant spatial derivatives is carried by the numerical schemes, and not by the learned NNs. The space-time continuous form also made it very easy to interpret the learned closures. We derived the adjoint equations and network architectures needed to efficiently implement the nDDEs and nPDDEs, agnostic to the specifics of the time-integration schemes, across differentiable and non-differentiable computational physics codes, in different machine learning frameworks, and also for non-uniformly-spaced spatiotemporal training data.

Through simulation experiments, we showed that our neural closure methodology drastically improved the long-term predictive capability of low-fidelity models for the main classes of model truncations. Specifically, our neural closure models efficiently accounted for truncated modes in reduced-order-models, captured the effects of subgrid-scale processes in coarse models, and augmented the simplification of complex and non-autonomous physical-biogeochemical models. We also showed that there exists an optimal amount of past information to incorporate, and provided methodology to learn it from data during the training process. Computational advantages were also discussed.

Applications of our Bayesian learning and neural closure modeling framework are not just limited to the experiments shown in this thesis. They can be widely extended to other fields such as control theory, robotics, pharmacokinetic-pharmacodynamics, chemistry, economics, biological regulatory systems, etc.

8.2 Future Work

Scientific machine learning as a field has a long way to go, and the results and contributions presented in this thesis could be greatly extended in many different ways and applied to a variety of problems.

Possible future extensions for the Bayesian model learning framework involves, its incorporation in the recently-developed probabilistic Dynamically Orthogonal prim-

itive equation (DO-PE) regional ocean modeling system [235, 236, 237, 238]. The MSEAS DO-PE is combination of the MSEAS PE model [63, 64] with the DO methodology to perform probabilistic predictions in realistic ocean models, extending ensemble approaches [84, 239]. Such combination would enable the use real-world ocean observations collected by a variety of platforms [96] and would allow realistic model learning and also possibly the discovery new models still unknown to the scientific community. Our methodological results could also be applied to other ocean domains or to other disciplines. For example, it could be utilized to improve and learn plastic pollution models, which is also poses a threat to humanity [240]. It could also be useful for ocean acoustics prediction and inference [241, 242], for the optimization of reduced order models and onboard learning [243, 244, 245], and for the planning of underwater vehicles [246, 1, 247, 248, 249, 250, 251].

The nCMs framework could be extended to account for uncertainties, and noise in the training data. This would enable developing new model closures for stochastic dynamical systems. The optimal delay value learning can be further derived and implemented for the generalized nCMs framework, which would make it easier to use the framework out-of-the-box. For applications to realistic models and real-world ocean data, as for the Bayesian learning, the generalized nCM framework could be coupled with the MSEAS PE model and applied to other ocean areas and modeling domains.

Overall, there is a need to convert all the developed frameworks to robust software packages, and so enable efficient transfer of knowledge and sharing with other scientists and develop applications to new problems.

Appendix A

Dynamically Orthogonal (DO) Equations

In this appendix, we derive the dynamically orthogonal (DO) equations [11, 12, 14, 13] used in this paper for efficient reduced-dimension probabilistic evolution of high-dimensional stochastic dynamical systems with various sources of uncertainties.

Let us consider a general stochastic dynamical system which encompasses the different model uncertainty scenarios encountered in this paper for the biological tracer fields (or any other fields such as velocity, temperature, etc.). The stochastic dynamical system is defined on the domain $\mathcal{D}$, governing the dynamics of $\boldsymbol{\phi}(\boldsymbol{x},t;\omega) : \mathbb{R}^n \times [0,T] \to \mathbb{R}^{N_\phi}$ the stochastic state vector comprising N_ϕ tracer state fields, where ω is the realization index belonging to a measurable sample space Ω, and given by,

$$\begin{aligned} \frac{\partial \boldsymbol{\phi}(\boldsymbol{x},t;\omega)}{\partial t} &= \mathcal{L}[\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\theta}(\omega),\boldsymbol{\beta}(\omega),\boldsymbol{x},t;\omega] + \widehat{\mathcal{L}}[\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(\omega),\boldsymbol{x},t;\omega] \\ &\quad + \widetilde{\mathcal{L}}[\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\gamma}(\omega),\boldsymbol{x},t;\omega], \quad \boldsymbol{x}\in\mathcal{D},\ t\in[0,T],\ \omega\in\Omega\,, \\ \text{with} \quad & \boldsymbol{\phi}(\boldsymbol{x},0;\omega) = \boldsymbol{\phi}_o(\boldsymbol{x};\omega)\,, \\ \text{and} \quad & \mathcal{B}[\boldsymbol{\phi}(\boldsymbol{x},t;\omega)] = \boldsymbol{b}(\boldsymbol{x},t;\omega),\ \boldsymbol{x}\in\partial\mathcal{D},\ t\in[0,T],\ \omega\in\Omega\,. \end{aligned} \tag{A.1}$$

$\boldsymbol{\phi}_o(\boldsymbol{x};\omega)$, $\mathcal{B}$, and $\boldsymbol{b}(\boldsymbol{x},t;\omega)$ are the stochastic initial conditions, boundary condition operators, and boundary values respectively. The functional form of the first dynamics term $\mathcal{L}[\bullet]$ is assumed to be known, however contains N_θ uncertain parameters

$\boldsymbol{\theta}(\omega)$. The second term $\widehat{\mathcal{L}}[\bullet]$ is uncertain but belongs to a family of candidate functions, and parameterized using N_α special stochastic parameters $\boldsymbol{\alpha}(\omega)$. $\widehat{\mathcal{L}}[\bullet]$ could also contain uncertain regular parameters $\boldsymbol{\theta}(\omega)$. The third term $\widetilde{\mathcal{L}}[\bullet]$ has a functional form completely unknown, and is parameterized using N_γ stochastic expansion coefficients $\boldsymbol{\gamma}(\omega)$. In the above general stochastic dynamical system, it is also assumed that candidate models of different complexities are combined using N_β special stochastic parameters $\boldsymbol{\beta}(\omega)$. The $\beta_k(\omega)$'s multiplied with the original state variables (as described in Sect. 2.2.1), are assumed to be absorbed into ϕ_i's and not explicitly shown, however, $\beta_k(\omega)$'s can still appear on the right-hand-side (RHS), as in $\mathcal{L}[\bullet]$ and $\widehat{\mathcal{L}}[\bullet]$.

For efficient reduced-dimension probabilistic evolution of high-dimensional systems, the DO methodology [11, 12, 14, 13, 15] employs a generalized, time-dependent Karhunen-Loéve decomposition of a stochastic state vector up to arbitrary precision,

$$\boldsymbol{\phi}(\boldsymbol{x},t;\omega) = \bar{\boldsymbol{\phi}}(\boldsymbol{x},t) + \sum_{i=1}^{N_s} Y_i(t;\omega)\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t) \ . \tag{A.2}$$

The stochastic state $\boldsymbol{\phi}(\boldsymbol{x},t;\omega)$ is decomposed into a mean, $\bar{\boldsymbol{\phi}}(\boldsymbol{x},t) \in \mathbb{R}^{N_\phi}$, N_s deterministic modes, $\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t) \in \mathbb{R}^{N_\phi}$, and stochastic coefficients, $Y_i(t;\omega) \in \mathbb{R}$. We can define the stochastic subspace $\boldsymbol{V_S} = span\{\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t)\}_{i=1}^{N_s}$ as the linear space spanned by the N_s deterministic modes. They are chosen in such a way that the dominant uncertainty resides in $\boldsymbol{V_S}$. Hence, we employ orders of magnitude less number of modes as compared to the dimension of the discretized state variables or of the domain grid N_x, i.e. $N_s \ll N_\phi N_x$. Similarly, uncertain regular and special parameters can be split into mean and deviation part, $\boldsymbol{\theta}(\omega) = \bar{\boldsymbol{\theta}} + \boldsymbol{\mathfrak{D}}^\theta(\omega)$, $\boldsymbol{\alpha}(\omega) = \bar{\boldsymbol{\alpha}} + \boldsymbol{\mathfrak{D}}^\alpha(\omega)$, and $\boldsymbol{\beta}(\omega) = \bar{\boldsymbol{\beta}} + \boldsymbol{\mathfrak{D}}^\beta(\omega)$.

Non-linear terms on the RHS are handled using local Taylor series expansion around the statistical means of states and parameters. We use the 1st order Taylor

series expansion for the $\mathcal{L}[\bullet]$ and the $\widehat{\mathcal{L}}[\bullet]$ terms,

$$\begin{aligned}\mathcal{L}[\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\theta}(\omega),\boldsymbol{\beta}(\omega),\boldsymbol{x},t;\omega] \approx \mathcal{L}|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \left.\frac{\partial\mathcal{L}}{\partial\boldsymbol{\phi}}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \sum_{i=1}^{N_s}\tilde{\boldsymbol{\phi}}_i Y_i \\ + \sum_{i=1}^{N_\theta}\left.\frac{\partial\mathcal{L}}{\partial\theta_i}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}\mathfrak{D}_i^\theta + \sum_{i=1}^{N_\beta}\left.\frac{\partial\mathcal{L}}{\partial\beta}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}}\mathfrak{D}_i^\beta\ , \\ \widehat{\mathcal{L}}[\boldsymbol{\phi}(\boldsymbol{x},t;\omega),\boldsymbol{\alpha}(\omega),\boldsymbol{x},t;\omega] \approx \widehat{\mathcal{L}}|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}}}} + \left.\frac{\partial\widehat{\mathcal{L}}}{\partial\boldsymbol{\phi}}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}}}} \sum_{i=1}^{N_s}\tilde{\boldsymbol{\phi}}_i Y_i \\ + \sum_{i=1}^{N_\alpha}\left.\frac{\partial\widehat{\mathcal{L}}}{\partial\alpha_i}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}}}}\mathfrak{D}_i^\alpha\ .\end{aligned} \tag{A.3}$$

Using a higher-order polynomial approximation leads to higher accuracy for the DO evolution, however, at the same time increasing the computational costs. For a more detailed discussion on the scaling of computational costs with the order of polynomial approximation, please refer to Gupta, 2016 [92]. Handling the $\widetilde{\mathcal{L}}[\bullet]$ term is less straightforward because of the need to evaluate the interval in which a state realization value lies at every location in the domain (see Sect. 2.2.2). Thus, currently we evaluate the $\widetilde{\mathcal{L}}[\bullet]$ term for every state realization in a Monte-Carlo way. However, this could potentially be circumvented and made more efficient in the future by using techniques such as clustering [73].

To derive the DO equations, we substitute the KL decomposition (Eq. A.2) into the stochastic system (Eq. A.1). In order to get a closed-form system, we impose additional constraints on the modes. As shown in Sapsis and Lermusiaux, 2009 ([11]), an appropriate constraint is the DO condition: the rate of change of the stochastic subspace being orthogonal to itself, expressed as,

$$\frac{d\boldsymbol{V_S}}{dt} \perp \boldsymbol{V_S} \Leftrightarrow \left\langle \frac{\partial\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t)}{\partial t}, \tilde{\boldsymbol{\phi}}_j(\boldsymbol{x},t)\right\rangle = 0 \quad \forall i,j \in \{1,...,N_s\}\,, \tag{A.4}$$

where the operator $\langle \boldsymbol{a},\boldsymbol{b}\rangle$ represents the spatial inner-product of arbitrary vectors $\boldsymbol{a}=[a^1,a^2,...]^T$ and $\boldsymbol{b}=[b^1,b^2,...]^T$ defined by $\langle \boldsymbol{a},\boldsymbol{b}\rangle = \int_{\mathcal{D}}\sum_i (a^i b^i)d\mathcal{D}$. Note that the

DO condition (Eq. A.4) also implies the preservation of orthogonality for the basis $\{\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t)\}_{i=1}^{N_s}$ themselves, since,

$$\frac{\partial}{\partial t}\langle\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t),\tilde{\boldsymbol{\phi}}_j(\boldsymbol{x},t)\rangle = \left\langle\frac{\partial\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t)}{\partial t},\tilde{\boldsymbol{\phi}}_j(\boldsymbol{x},t)\right\rangle + \left\langle\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t),\frac{\partial\tilde{\boldsymbol{\phi}}_j(\boldsymbol{x},t)}{\partial t}\right\rangle = 0, \quad \forall i,j\in\{1,...,N_s\}\,. \tag{A.5}$$

Substituting the expansion (Eq. A.2) into the stochastic dynamical model (Eq. A.1) with the help of DO condition (Eq. A.4), a unique set of independent evolution equations can be derived for mean, modes, and stochastic coefficients. These are the DO evolution equations (omitting function arguments for brevity),

$$\begin{aligned}
\frac{\partial\bar{\boldsymbol{\phi}}}{\partial t} &= \mathcal{L}|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \widehat{\mathcal{L}}|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \mathbb{E}[\widetilde{\mathcal{L}}]\,,\\
\frac{\partial\tilde{\boldsymbol{\phi}}_i}{\partial t} &= \boldsymbol{Q}_i - \sum_{j=1}^{N_s}\langle\boldsymbol{Q}_i,\tilde{\boldsymbol{\phi}}_j\rangle\tilde{\boldsymbol{\phi}}_j\,,\\
\frac{dY_i}{dt} &= \sum_{m=1}^{N_s}\langle\boldsymbol{F}_m,\tilde{\boldsymbol{\phi}}_i\rangle Y_m + \sum_{m=1}^{N_\theta}\left\langle\frac{\partial\mathcal{L}}{\partial\theta_i}\Big|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}},\tilde{\boldsymbol{\phi}}_i\right\rangle\mathfrak{D}_m^\theta + \sum_{m=1}^{N_\beta}\left\langle\frac{\partial\mathcal{L}}{\partial\beta}\Big|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}},\tilde{\boldsymbol{\phi}}_i\right\rangle\mathfrak{D}_m^\beta\\
&\quad + \sum_{m=1}^{N_\theta}\left\langle\frac{\partial\widehat{\mathcal{L}}}{\partial\theta_i}\Big|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}},\tilde{\boldsymbol{\phi}}_i\right\rangle\mathfrak{D}_m^\theta + \sum_{m=1}^{N_\alpha}\left\langle\frac{\partial\widehat{\mathcal{L}}}{\partial\alpha_i}\Big|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}},\tilde{\boldsymbol{\phi}}_i\right\rangle\mathfrak{D}_m^\alpha + \sum_{m=1}^{N_\beta}\left\langle\frac{\partial\widehat{\mathcal{L}}}{\partial\beta_i}\Big|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}},\tilde{\boldsymbol{\phi}}_i\right\rangle\mathfrak{D}_m^\beta\\
&\quad + \left\langle\widetilde{\mathcal{L}} - \mathbb{E}[\widetilde{\mathcal{L}}],\tilde{\boldsymbol{\phi}}_i\right\rangle\,,
\end{aligned} \tag{A.6}$$

where,

$$
\begin{aligned}
\boldsymbol{Q}_i = & \left.\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_i + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\theta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}^\theta_n Y_j} \left.\frac{\partial \mathcal{L}}{\partial \theta_n}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\beta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}^\beta_n Y_j} \left.\frac{\partial \mathcal{L}}{\partial \beta_n}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \\
& + \left.\frac{\partial \widehat{\mathcal{L}}}{\partial \boldsymbol{\phi}}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_i + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\theta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}^\theta_n Y_j} \left.\frac{\partial \widehat{\mathcal{L}}}{\partial \theta_n}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\alpha} C^{-1}_{Y_iY_j} C_{\mathfrak{D}^\alpha_n Y_j} \left.\frac{\partial \widehat{\mathcal{L}}}{\partial \alpha_n}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \\
& + \sum_{j=1}^{N_s}\sum_{n=1}^{N_\beta} C^{-1}_{Y_iY_j} C_{\mathfrak{D}^\beta_n Y_j} \left.\frac{\partial \widehat{\mathcal{L}}}{\partial \beta_n}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} + \sum_{j=1}^{N_s} C^{-1}_{Y_iY_j} \mathbb{E}[Y_j \widetilde{\mathcal{L}}] \,, \\
\boldsymbol{F}_m = & \left.\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_m + \left.\frac{\partial \widehat{\mathcal{L}}}{\partial \boldsymbol{\phi}}\right|_{\substack{\boldsymbol{\phi}=\bar{\boldsymbol{\phi}},\\ \boldsymbol{\theta}=\bar{\boldsymbol{\theta}},\\ \boldsymbol{\alpha}=\bar{\boldsymbol{\alpha}},\\ \boldsymbol{\beta}=\bar{\boldsymbol{\beta}}}} \tilde{\boldsymbol{\phi}}_m \,,
\end{aligned}
\tag{A.7}
$$

where $\mathbb{E}[\bullet]$ represents the expectation operator, and $C^{-1}_{Y_iY_j}$ represents the inverse of the cross-covariance between i^{th} and j^{th} stochastic coefficients, where $C_{Y_iY_j}$ is given by,

$$
C_{Y_i,Y_j} = \mathbb{E}[Y_i(t;\omega)Y_j(t;\omega)] \,. \tag{A.8}
$$

We can also obtain the boundary condition for the mean field,

$$
\mathcal{B}[\bar{\boldsymbol{\phi}}(\boldsymbol{x},t)]|_{\boldsymbol{x}\in\partial\mathcal{D}} = \mathbb{E}[\boldsymbol{b}(\boldsymbol{x},t;\omega)] \,, \tag{A.9}
$$

and for the modes field as,

$$
\mathcal{B}[\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t)]|_{\boldsymbol{x}\in\partial\mathcal{D}} = \sum_{j=1}^{N_s} \mathbb{E}[Y_j(t;\omega)\boldsymbol{b}(\boldsymbol{x},t;\omega)] C^{-1}_{Y_iY_j} \,. \tag{A.10}
$$

The initial conditions can be found by approximating the initial field $\boldsymbol{\phi}_o(\boldsymbol{x};\omega)$ using the DO decomposition. Complete derivation of the DO equations, along with discussion on computational cost saving can be found in several of the existing papers

on DO methodology [11, 12, 14, 13]. For a more detailed discussion on handling stochastic boundary condition, please refer to Gupta, 2016 & 2022 ([92, 252]).

Furthermore, special attention needs to be given to the difference in the amount and magnitude of uncertainty in different state variables, which is often the case in multidisciplinary dynamics. This can be achieved by using appropriate scaling while defining the inner-product operator. Let $\tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t) = [\tilde{\phi}_i^1(\boldsymbol{x},t), ..., \tilde{\phi}_i^{N_\phi}(\boldsymbol{x},t)]$, and the inner-product be defined as,

$$\langle \tilde{\boldsymbol{\phi}}_i(\boldsymbol{x},t), \tilde{\boldsymbol{\phi}}_j(\boldsymbol{x},t) \rangle = \frac{1}{|\mathcal{D}|} \int_{\mathcal{D}} \sum_k^{N_\phi} \left(\frac{1}{\sigma_{nd,k}^2} \tilde{\phi}_i^k \tilde{\phi}_j^k \right) d\mathcal{D} \ , \tag{A.11}$$

where $|\mathcal{D}|$ is the area (volume) of the domain, while $\sigma_{nd,\bullet}$ represents the expected standard deviation of the corresponding state variable. $\sigma_{nd,\bullet}$ determines the relative weighting given to different state variables during the DO evolution.

Appendix B

Gaussian Mixture Model (GMM)-DO Filter

The DO methodology introduced in the Appendix A helps to effectively evolve uncertainty between assimilation steps, i.e. provide prior probability distributions for the state variables. For the assimilation step, we employ a framework based on Gaussian Mixture Models (in order to preserve non-Gaussian statistics of state variables) and Bayes law, called the GMM-DO filter. The GMM-DO filter consists of a recursive succession of two steps: a forecast step and an update step. Due to the affine transformation between stochastic coefficients and state variables, a Bayesian update of the state variable distribution can be achieved through an equivalent update of the stochastic coefficient distribution. Hence, the GMM-DO filter takes advantage of this fact to achieve efficient reduced-dimension Bayesian state variable inference [16, 17]. In this section, we will only focus on deriving the GMM-DO filter for state variables.

We start from either a discretized initial stochastic state distribution in the DO form or the posterior state distribution from the assimilation of data at time t_{k-1}, i.e. the GMM-DO posterior at time t_{k-1}, $\mathbf{\Phi}^a_{k-1} = \bar{\mathbf{\Phi}}^a_{k-1} + \tilde{\mathbf{\Phi}}^a_{k-1}\boldsymbol{Y}^a_{k-1}$ where, in general, $\bar{\mathbf{\Phi}}(t) \in \mathbb{R}^{N_\phi N_x}$ represents the discretized mean, $\tilde{\mathbf{\Phi}}(t) \in \mathbb{R}^{N_\phi N_x \times N_s}$ represents the matrix of discretized modes and $\boldsymbol{Y}(t;\omega) \in \mathbb{R}^{N_s}$ or $\boldsymbol{Y}(t) \in \mathbb{R}^{N_s \times N_r}$ represents stochastic coefficient matrix with N_r being the number of Monte-Carlo samples. Next, we use the DO equations (A.6 & A.7) to evolve the probabilistic description of the state

vector in time, arriving at the forecast for observation time t_k, $\boldsymbol{\Phi}_k^f = \bar{\boldsymbol{\Phi}}_k^f + \tilde{\boldsymbol{\Phi}}_k^f \boldsymbol{Y}_k^f$. Observations are made available at time t_k in accordance with the observation model equation (Eq. 2.3; for brevity, we drop the subscript time index k). Now, the overall goal is to update the mean ($\bar{\boldsymbol{\Phi}}^f$) and stochastic coefficients ($\boldsymbol{Y}^f$) of the forecast in accordance with the realized observations $\boldsymbol{y}$ to obtain GMM-DO estimate of the posterior mean ($\bar{\boldsymbol{\Phi}}^a$) and stochastic coefficients ($\boldsymbol{Y}^a$).

The first step is approximating the prior probability distribution of the stochastic coefficients in the DO subspace using a GMM,

$$p_{\boldsymbol{Y}^f}(Y^f) \approx \sum_{j=1}^{N_{\text{GMM}}} \pi_{\boldsymbol{Y},j}^f \times \mathcal{N}(Y^f; \boldsymbol{\mu}_{\boldsymbol{Y},j}^f, \boldsymbol{\Sigma}_{\boldsymbol{Y},j}^f) \qquad \forall Y^f \in \mathbb{R}^{N_s}\,, \tag{B.1}$$

where N_{GMM} is the to-be-determined number of GMM components, $\pi_{\boldsymbol{Y},j}^f \in [0,1]$ the j^{th} component weight (also $\sum_{j=1}^{N_{\text{GMM}}} \pi_{\boldsymbol{Y},j}^f = 1$), $\boldsymbol{\mu}_{\boldsymbol{Y},j}^f$ the j^{th} component mean vector and $\boldsymbol{\Sigma}_{\boldsymbol{Y},j}^f$ the j^{th} component covariance matrix. This approximation is found by performing a semiparametric fit to the Monte-Carlo samples used to numerically evolve the stochastic coefficients. Specifically, the expectation-maximization (EM) algorithm for GMMs [253] is used to find maximum likelihood estimate for the GMM parameters $\pi_{\boldsymbol{Y},j}^f$, $\boldsymbol{\mu}_{\boldsymbol{Y},j}^f$ and $\boldsymbol{\Sigma}_{\boldsymbol{Y},j}^f$, while the selection of the number of GMM components (N_{GMM}) is determined by the Bayesian Information Criterion (BIC) [254] by successively fitting GMMs of increasing complexity (i.e. GMM = 1, 2, 3, ...) until a minimum of the BIC is obtained.

Finally, we perform a Bayesian update of the GMM prior of stochastic coefficients based on the Gaussian observation model (Eq. 2.3) to get another GMM by conjugacy [16]. The posterior stochastic coefficient distribution is given by,

$$p_{\boldsymbol{Y}^a}(Y^a) \approx \sum_{j=1}^{N_{\text{GMM}}} \pi_{\boldsymbol{Y},j}^a \times \mathcal{N}(Y^a; \boldsymbol{\mu}_{\boldsymbol{Y},j}^a, \boldsymbol{\Sigma}_{\boldsymbol{Y},j}^a)\,, \qquad \forall Y^a \in \mathbb{R}^{N_s}\,, \tag{B.2}$$

where,

$$
\begin{aligned}
\pi^a_{\boldsymbol{Y},j} &= \frac{\pi^f_{\boldsymbol{Y},j} \times \mathcal{N}(\tilde{\boldsymbol{y}}; \tilde{\boldsymbol{H}}\boldsymbol{\mu}^f_{\boldsymbol{Y},j}, \tilde{\boldsymbol{H}}\boldsymbol{\Sigma}^f_{\boldsymbol{Y},j}\tilde{\boldsymbol{H}}^T + \boldsymbol{R})}{\sum_{m=1}^{N_{\text{GMM}}} \pi^f_{\boldsymbol{Y},m} \times \mathcal{N}(\tilde{\boldsymbol{y}}; \tilde{\boldsymbol{H}}\boldsymbol{\mu}^f_{\boldsymbol{Y},m}, \tilde{\boldsymbol{H}}\boldsymbol{\Sigma}^f_{\boldsymbol{Y},m}\tilde{\boldsymbol{H}}^T + \boldsymbol{R})} \ , \qquad \forall j \in \{1, ..., N_{\text{GMM}}\} \ , \\
\boldsymbol{\mu}^a_{\boldsymbol{Y},j} &= \hat{\boldsymbol{\mu}}^a_{\boldsymbol{Y},j} - \sum_{m=1}^{N_{\text{GMM}}} \pi^a_{\boldsymbol{Y},m} \times \hat{\boldsymbol{\mu}}^a_{\boldsymbol{Y},m} \ , \qquad \forall j \in \{1, ..., N_{\text{GMM}}\} \ , \\
\boldsymbol{\Sigma}^a_{\boldsymbol{Y},j} &= (\boldsymbol{I} - \tilde{\boldsymbol{K}}_j\tilde{\boldsymbol{H}})\boldsymbol{\Sigma}^f_{\boldsymbol{Y},j} \ , \qquad \forall j \in \{1, ..., N_{\text{GMM}}\} \ ,
\end{aligned}
\tag{B.3}
$$

with the following definitions,

$$
\begin{aligned}
\tilde{\boldsymbol{H}} &= \boldsymbol{H}\tilde{\boldsymbol{\Phi}} \ , \\
\tilde{\boldsymbol{y}} &= \boldsymbol{y} - \boldsymbol{H}\bar{\boldsymbol{\Phi}}^f \ , \\
\hat{\boldsymbol{\mu}}^a_{\boldsymbol{Y},j} &= \boldsymbol{\mu}^f_{\boldsymbol{Y},j} + \tilde{\boldsymbol{K}}_j(\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{H}}\boldsymbol{\mu}^f_{\boldsymbol{Y},j}) \ , \qquad \forall j \in \{1, ..., N_{\text{GMM}}\} \ , \\
\tilde{\boldsymbol{K}}_j &= \boldsymbol{\Sigma}^f_{\boldsymbol{Y},j}\tilde{\boldsymbol{H}}^T(\tilde{\boldsymbol{H}}\boldsymbol{\Sigma}^f_{\boldsymbol{Y},j}\tilde{\boldsymbol{H}}^T + \boldsymbol{R})^{-1} \equiv \tilde{\boldsymbol{\Phi}}^T\boldsymbol{K}_j \ , \qquad \forall j \in \{1, ..., N_{\text{GMM}}\} \ .
\end{aligned}
\tag{B.4}
$$

Using an affine transformation, we can show that the posterior GMM stochastic coefficient distribution (Eq. B.2) is equivalent to the posterior GMM state space distribution, if the state vector mean is updated according to,

$$
\bar{\boldsymbol{\Phi}}^a = \bar{\boldsymbol{\Phi}}^f + \tilde{\boldsymbol{\Phi}} \sum_{j=1}^{N_{\text{GMM}}} \pi^a_{\boldsymbol{Y},j} \times \hat{\boldsymbol{\mu}}^a_{\boldsymbol{Y},j} \ . \tag{B.5}
$$

In this whole update process, no matrices were manipulated of size larger than $N_\phi N_x \times S \ll (N_\phi N_x)^2$, thus, making this method computationally feasible for large-dimensional systems.

At last, new Monte-Carlo samples are drawn from the posterior GMM stochastic coefficient distribution (Eq. B.2) and are dynamically evolved using the DO evolution equations until new observations come in and the filtering process is repeated. Hence, the GMM-DO filter along with the DO evolution equations provide an efficient and computationally feasible Bayesian inference methodology for high-dimensional, non-linear stochastic dynamical systems.

Appendix C

State Augmentation

In order to perform simultaneous estimation of uncertain parameters and states, we employ state augmentation [58]. We start by decomposing the stochastic regular parameters ($\boldsymbol{\theta}(\omega) \in \mathbb{R}^{N_\theta}$), special parameters ($\boldsymbol{\alpha}(\omega) \in \mathbb{R}^{N_\alpha}$ and $\boldsymbol{\beta}(\omega) \in \mathbb{R}^{N_\beta}$), and expansion coefficients ($\boldsymbol{\gamma}(\omega) \in \mathbb{R}^{N_\gamma}$) into their means and uncertain parts,

$$\begin{aligned} \boldsymbol{\theta}(\omega) &= \bar{\boldsymbol{\theta}} + \mathfrak{D}^{\boldsymbol{\theta}}(\omega) \ , \\ \boldsymbol{\alpha}(\omega) &= \bar{\boldsymbol{\alpha}} + \mathfrak{D}^{\boldsymbol{\alpha}}(\omega) \ , \\ \boldsymbol{\beta}(\omega) &= \bar{\boldsymbol{\beta}} + \mathfrak{D}^{\boldsymbol{\beta}}(\omega) \ , \\ \boldsymbol{\gamma}(\omega) &= \bar{\boldsymbol{\gamma}} + \mathfrak{D}^{\boldsymbol{\gamma}}(\omega) \ . \end{aligned} \tag{C.1}$$

The augmented state vector can be written as,

$$\boldsymbol{\Phi}_{\text{aug}}(t;\omega) = \begin{bmatrix} \boldsymbol{\theta}(\omega) \\ \boldsymbol{\alpha}(\omega) \\ \boldsymbol{\beta}(\omega) \\ \boldsymbol{\gamma}(\omega) \\ \boldsymbol{\Phi}(t;\omega) \end{bmatrix} \in \mathbb{R}^{N_\phi N_x + N_\theta + N_\alpha + N_\beta + N_\gamma} \ . \tag{C.2}$$

Now, let us write the DO decomposition for this new augmented system. We define a new coefficient matrix in which each parameter uncertainty amounts to an additional

scalar stochastic coefficient,

$$\boldsymbol{DY}(t;\omega) = \left[\boldsymbol{\mathfrak{D}}^{\boldsymbol{\theta}}(\omega)|\boldsymbol{\mathfrak{D}}^{\boldsymbol{\alpha}}(\omega)|\boldsymbol{\mathfrak{D}}^{\boldsymbol{\beta}}(\omega)|\boldsymbol{\mathfrak{D}}^{\boldsymbol{\gamma}}(\omega)|\boldsymbol{Y}(t;\omega)\right] \in \mathbb{R}^{N_s+N_\theta+N_\alpha+N_\beta+N_\gamma} , \quad \text{(C.3)}$$

a new modes matrix with parameters having unit modes,

$$\tilde{\boldsymbol{\Phi}}_{\text{aug}}(t) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \tilde{\boldsymbol{\Phi}}(t) \end{bmatrix} \in \mathbb{R}^{(N_\phi N_x+N_\theta+N_\alpha+N_\beta+N_\gamma)\times(N_s+N_\theta+N_\alpha+N_\beta+N_\gamma)} , \quad \text{(C.4)}$$

and a new augmented mean vector,

$$\bar{\boldsymbol{\Phi}}_{\text{aug}}(t) = \begin{bmatrix} \bar{\boldsymbol{\theta}} \\ \bar{\boldsymbol{\alpha}} \\ \bar{\boldsymbol{\beta}} \\ \bar{\boldsymbol{\gamma}} \\ \bar{\boldsymbol{\Phi}}(t) \end{bmatrix} \in \mathbb{R}^{N_\phi N_x+N_\theta+N_\alpha+N_\beta+N_\gamma} . \quad \text{(C.5)}$$

Thus, the DO decomposition of the augmented state is given by,

$$\begin{aligned} \boldsymbol{\Phi}_{\text{aug}}(t;\omega) &= \bar{\boldsymbol{\Phi}}_{\text{aug}}(t) + \sum_{i=1}^{\substack{N_s+N_\theta+N_\alpha \\ +N_\beta+N_\gamma}} \tilde{\boldsymbol{\Phi}}_{\text{aug},i}(t) DY_i(t;\omega) \\ &= \bar{\boldsymbol{\Phi}}_{\text{aug}}(t) + \tilde{\boldsymbol{\Phi}}_{\text{aug}}(t)\boldsymbol{DY}(t;\omega) . \end{aligned} \quad \text{(C.6)}$$

We can also define the new observation model as,

$$\begin{aligned} \boldsymbol{\mathcal{Y}} &= \begin{bmatrix} \boldsymbol{0} & \boldsymbol{H} \end{bmatrix} \boldsymbol{\Phi}_{\text{aug}} + \boldsymbol{V}, \qquad \boldsymbol{V} \sim \mathcal{N}(0, \boldsymbol{R}) \\ &= \boldsymbol{H}_{\text{aug}} \boldsymbol{\Phi}_{\text{aug}} + \boldsymbol{V} , \end{aligned} \quad \text{(C.7)}$$

where $\boldsymbol{H}$ is the original observation matrix, and $\boldsymbol{H}_{\text{aug}} \in \mathbb{R}^{N_y \times (N_\phi N_x+N_\theta+N_\alpha+N_\beta+N_\gamma)}$ the augmented observation matrix, while $\boldsymbol{\Phi}_{\text{aug}}$ is the augmented state ensemble.

We can consider the above augmented state vector as forecast for time t_k, and follow the GMM-DO algorithm presented in appendix B to obtain posterior distributions of the parameters and state variables. One of the main advantages of the above

methodology is the fact that it does not share the same stochastic coefficients between the states and parameters but still captures their joint distribution. Sometimes there can be orders of magnitude difference between state variables and parameters. In such cases, sharing of stochastic coefficients can lead to inefficient DO representation and evolution as they might require additional modes to effectively capture and evolve the joint uncertainty between states and parameters, thus increasing our computational cost. It might also be the case, that the presence of large numbers of uncertain parameters requires additional GMM components while fitting the augmented coefficient matrix. Because assimilation happens only at sparse times, this would not cause any significant increase in the overall computational costs.

Appendix D

Supplementary Information: Neural Closure Models for Dynamical Systems

D.1 Mori-Zwanzig Formulation

Without loss of generality, the full nonlinear dynamical system model is written as,

$$\frac{du_k(t)}{dt} = R_k(u(t), t)\,, \quad \text{with} \quad u_k(0) = u_{0k}\,, \quad k \in \mathfrak{F}\,. \tag{D.1}$$

The full state vector is $u = (\{u_k\}),\ k \in \mathfrak{F} = \mathfrak{R} \cup \mathfrak{U}$, where $\mathfrak{R}$ is the set corresponding to the resolved variables (e.g. coarse field or reduced variables), and $\mathfrak{U}$ the set corresponding to the unresolved variables (e.g. subgrid field or complement variables), which as a union, $\mathfrak{F}$, form the set for full space of variables. We also denote $u = \{\hat{u}, \tilde{u}\}$ where $\hat{u} = (\{u_k\}),\ k \in \mathfrak{R}$ and $\tilde{u} = (\{u_k\}),\ k \in \mathfrak{U}$. Similarly, $u_0 = \{\hat{u}_0, \tilde{u}_0\}$, with $\hat{u}_0 = (\{u_{0k}\}),\ \in \mathfrak{R}$ and $\tilde{u}_0 = (\{u_{0k}\}),\ k \in \mathfrak{U}$.

We can write the above non-linear system of ODEs (Eq. D.1) exactly as a system of linear PDEs by casting it in the Liouville form,

$$\frac{\partial \phi_k}{\partial t} = L\phi_k\,, \quad \text{with} \quad \phi_k(u_0, 0) = u_{0k}\,, \quad k \in \mathfrak{F}\,, \tag{D.2}$$

where the Liouville operator is $L = \sum_{i \in \mathfrak{F}} R_i(u_0) \frac{\partial}{\partial u_{0i}}$, with R_i denoting element i of the full model dynamics (Eq. D.1). The solution of Eq. D.2 is given by $u_k(u_0, t) = \phi_k(u_0, t) = e^{tL}\phi_k(u_0, 0)$. Hence, we can also rewrite Eq. D.2 as,

$$\frac{\partial}{\partial t} e^{tL} u_{0k} = L e^{tL} u_{0k}\,, \qquad k \in \mathfrak{F}\,. \tag{D.3}$$

Now, let P be a orthogonal projection on the space of functions of the resolved initial conditions $\hat{u}_0$, such that, for any nonlinear function $h(u_0) = h(\{\hat{u}_0, \tilde{u}_0\})$, then $P(h(u_0)) = h(\hat{u}_0)$. Similarly, $Q = I - P$ is the projection on the null space of P. It is important to note that the projectors P and Q used in this formulation are fundamentally different from L_2 projectors. Using the Dyson's formula $e^{tL} = e^{tQL} + \int_0^t e^{(t-s)L} P L e^{sQL} ds$, and noting that L and e^{tL} commute, we can then exactly rewrite Eq. D.3 as,

$$\frac{\partial}{\partial t} e^{tL} u_{0k} = e^{tL} P L u_{0k} + e^{tQL} Q L u_{0k} + \int_0^t e^{(t-s)L} P L e^{sQL} Q L u_{0k} ds\,, \quad k \in \mathfrak{R}\,, \tag{D.4}$$

which is called the Mori-Zwanzig (MZ) formulation. Importantly, the above equation is an exact representation of Eq. D.3 for the resolved components. For convenience, we denote $F_k(u_0, t) = e^{tQL} Q L u_{0k}$ and $K_k(u_0, t) = PLF_k(u_0, t)$, and thus further rewrite Eq. D.4 as,

$$\frac{\partial}{\partial t} u_k(u_0, t) = \underbrace{R_k(\hat{u}(u_0, t))}_{\text{Markovian}} + \underbrace{F_k(u_0, t)}_{\text{Noise}} + \underbrace{\int_0^t K_k(\hat{u}(u_0, t-s), s) ds}_{\text{Memory}}, \quad k \in \mathfrak{R}\,, \tag{D.5}$$

where R_k is again the same as that in the full model dynamics given by Eq. D.1. Eq. D.5 provides useful guidance for closure modeling. The first term in Eq. D.5 is the Markovian term dependent only on the values of the variables at the present time, while the closure consists of two terms: the noise term and a memory term that is non-Markovian. We can further simplify Eq. D.5 by applying the P projection and using the fact that the noise term lives in the null space of P for all times, which could be easily proved. For ROMs with initial conditions devoid of any unresolved

dynamics, i.e. $\tilde{u}_0 = 0$ and thus $u_0 = \hat{u}_0$, we then retain the exact dynamics after the projection step, noticing in this case that $Pu_k(u_0, t) = u_k(\hat{u}_0, t), \forall k \in \mathfrak{R}$,

$$\frac{\partial}{\partial t} u_k(\hat{u}_0, t) = PR_k(\hat{u}(\hat{u}_0, t)) + P \int_0^t K_k(\hat{u}(\hat{u}_0, t-s), s) ds, \quad k \in \mathfrak{R}\,. \tag{D.6}$$

Hence, for such systems, the closure model would only consider the non-Markovian memory term. The above derivation of the MZ formulation has been adapted from [160, 2, 162].

D.2 Adjoint Equations for Neural Delay Differential Equations

Here, we provide a detailed derivation of adjoint equations for neural DDEs with discrete and distributed delays. For related derivations, we refer to [255, 212].

D.2.1 Discrete-nDDE

The neural-network parameterized discrete DDE is given by,

$$\begin{aligned} \frac{du(t)}{dt} &= f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta), \quad t \in (0, T] \\ u(t) &= h(t), \quad t \leq 0 \end{aligned} \tag{D.7}$$

where $\tau_1, ..., \tau_K$ are K number of discrete delays and $f_{RNN}(\bullet, t; \theta)$ is any recurrent architecture with trainable parameters θ. Let data be available at M times, $T_1 < ... < T_M \leq T$. Our goal is to optimize the total loss function, $\mathcal{L} = \int_0^T \sum_{i=1}^M l(u(t))\delta(t - T_i)dt$ (where $l(\bullet)$ are scalar loss functions such as mean-squared-error (MSE), and $\delta(t)$ is the Kronecker delta function), given the data and nDDE.

We first start by writing the Lagrangian for the above system,

$$\begin{aligned} L =& \mathcal{L}(u(t)) + \int_0^T \lambda^T(t) \left(d_t u(t) - f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta) \right) dt \\ &+ \int_{-\tau_K}^0 \mu^T(t)(u(t) - h(t))dt \end{aligned} \tag{D.8}$$

where $\lambda(t)$ and $\mu(t)$ are the Lagrangian variables, and where, for brevity, we use $\frac{\partial}{\partial(\bullet)} \equiv \partial_{(\bullet)}$ and $\frac{d}{d(\bullet)} = d_{(\bullet)}$ from now on. We also assume that the loss function ($\mathcal{L}$) and the initial conditions ($h(t),\ t \leq 0$) are independent of θ. Hence, the derivative of the Lagrangian w.r.t. θ is given by,

$$\begin{aligned} d_\theta L =& \partial_{u(t)} \mathcal{L}(u(t)) d_\theta u(t) + \int_0^T \lambda^T (d_\theta d_t u(t) - \partial_{u(t)} f_{RNN}(\bullet, t; \theta) d_\theta u(t) \\ &- \partial_{u(t-\tau_1)} f_{RNN}(\bullet, t; \theta) d_\theta u(t-\tau_1) ... - \partial_{u(t-\tau_K)} f_{RNN}(\bullet, t; \theta) d_\theta u(t-\tau_K) \\ &- \partial_\theta f_{RNN}(\bullet, t; \theta)) dt. \end{aligned} \tag{D.9}$$

Using integration-by-parts, we can write,

$$\int_0^T \lambda^T d_\theta d_t u(t) dt = \lambda^T(T) d_\theta u(T) - \lambda^T(0) d_\theta u(0) - \int_0^T d_t \lambda^T(t) d_\theta u(t) dt \tag{D.10}$$

and by change of variables,

$$
\begin{aligned}
&\int_0^T \lambda^T(t)\partial_{u(t-\tau_i)} f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta) d_\theta u(t-\tau_i) dt \\
&= \int_{-\tau_i}^{T-\tau_i} \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) d_\theta u(t) dt \\
&= \int_0^T \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) d_\theta u(t) dt \\
&\quad + \int_{-\tau_i}^{0} \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) d_\theta u(t) dt \\
&\quad - \int_{T-\tau_i}^{T} \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) d_\theta u(t) dt \\
&= \int_0^T \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) d_\theta u(t) dt \\
&\quad + \int_{-\tau_i}^{0} \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) d_\theta u(t) dt \,.
\end{aligned}
\tag{D.11}
$$

We further assume $\lambda(t) = 0,\ t \geq T$. Inserting everything back into Eq. D.9, we obtain,

$$
\begin{aligned}
d_\theta L = &\int_0^T \left(\sum_{i=1}^{M} \partial_{u(t)} l(u(t)) \delta(t - T_i) - d_t \lambda^T(t) - \lambda^T(t) \partial_{u(t)} f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta) \right. \\
&\left. - \sum_{i=1}^{K} \lambda^T(t+\tau_i) \partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta) \right) d_\theta u(t) dt \\
&- \int_0^T \lambda^T(t) \partial_\theta f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta) dt + \lambda^T(T) d_\theta u(T) - \lambda^T(0) d_\theta u(0) \\
&+ \sum_{i=1}^{K} \int_{-\tau_i}^{0} \lambda^T(t+\tau_i) \partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i) d_\theta u(t) dt \,.
\end{aligned}
\tag{D.12}
$$

The last two term in the above equation is zero because of the user-defined initial condition are independent of θ. Further, we aim to eliminate of $d_\theta u(t)$ everywhere, because avoiding the need to compute it explicitly is the main premise of the adjoint

method. We can do this if we assume,

$$
\begin{aligned}
&d_t\lambda^T(t) = \sum_{i=1}^{M} \partial_{u(t)} l(u(t))\delta(t-T_i) - \lambda^T(t)\partial_{u(t)} f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta) \\
&- \sum_{i=1}^{K} \lambda^T(t+\tau_i)\partial_{u(t)} f_{RNN}(u(t+\tau_i), u(t-\tau_1+\tau_i), ..., u(t-\tau_K+\tau_i), t+\tau_i; \theta),\ t \in [0, T) \\
&\lambda(t) = 0, \qquad t \geq T
\end{aligned}
\tag{D.13}
$$

Finally, after solving the above adjoint equation in $\lambda(t)$, we can compute the required derivative $d_\theta L$ as,

$$
d_\theta L = -\int_0^T \lambda^T(t)\partial_\theta f_{RNN}(u(t), u(t-\tau_1), ..., u(t-\tau_K), t; \theta)dt\ . \tag{D.14}
$$

D.2.2 Distributed-nDDE

The neural network parameterized distributed DDE is given by,

$$
\begin{aligned}
&\frac{du(t)}{dt} = f_{NN}\left(u(t), \int_{t-\tau_2}^{t-\tau_1} g_{NN}(u(\tau), \tau; \phi)d\tau, t; \theta\right), \quad t \in (0, T] \\
&u(t) = h(t), \quad t \leq 0
\end{aligned}
\tag{D.15}
$$

where $\tau_2 \geq \tau_1$ are the delay amounts; and $f_{NN}(\bullet; \theta)$, $g_{NN}(\bullet; \phi)$ are neural-networks with trainable parameters θ, ϕ respectively. By introducing a new variable $y(t) = \int_{t-\tau_2}^{t-\tau_1} g_{NN}(u(\tau), \tau; \phi)d\tau$, we can rewrite the above equation as a coupled discrete DDEs,

$$
\begin{aligned}
&\frac{du(t)}{dt} = f_{NN}\left(u(t), y(t), t; \theta\right), \quad t \in (0, T] \\
&\frac{dy(t)}{dt} = g_{NN}(u(t-\tau_1), t-\tau_1; \phi) - g_{NN}(u(t-\tau_2), t-\tau_2; \phi), \quad t \in (0, T] \\
&u(t) = h(t), \quad \tau_2 \leq t \leq 0 \\
&y(0) = \int_{-\tau_2}^{-\tau_1} g_{NN}(h(t), t; \phi)dt
\end{aligned}
\tag{D.16}
$$

Also, let data be available at M times, $T_1 < ... < T_M \leq T$. Our goal is to optimize the scalar loss function, $\mathcal{L}(u(t)) = \int_0^T \sum_{i=1}^M l(u(t))\delta(t - T_i)dt$, given the data and nDDE.

We will again start by writing the Lagrangian for this setup,

$$\begin{aligned} L =&\mathcal{L}(u(t)) + \int_0^T \lambda^T(t)(d_t u(t) - f_{NN}(u(t), y(t), t; \theta))\, dt \\ &+ \int_0^T \mu^T(t)\left(d_t y(t) - g_{NN}(u(t-\tau_1), t-\tau_1; \phi) + g_{NN}(u(t-\tau_2), t-\tau_2; \phi)\right) dt \\ &+ \int_{-\tau_2}^0 \gamma^T(t)(u(t) - h(t))dt + \alpha^T \left(y(0) - \int_{-\tau_2}^{-\tau_1} g_{NN}(h(t), t; \phi)dt\right), \end{aligned} \tag{D.17}$$

where $\lambda(t), \mu(t), \gamma(t)$, and α are the Lagrangian variables. Now in this case, we need to obtain the derivatives of the Lagrangian w.r.t. both θ and ϕ. We will first obtain $d_\theta L$,

$$\begin{aligned} d_\theta L =&\partial_{u(t)}\mathcal{L}(u(t))d_\theta u(t) + \int_0^T \lambda^T(t)\left(d_\theta d_t u(t) - \partial_{u(t)} f_{NN}(u(t), y(t), t; \theta)d_\theta u(t)\right. \\ &\left. -\partial_{y(t)} f_{NN}(u(t), y(t), t; \theta)d_\theta y(t) - \partial_\theta f_{NN}(u(t), y(t), t; \theta)\right) dt \\ &+ \int_0^T \mu^T(t)\left(d_\theta d_t y(t) - \partial_{u(t-\tau_1)} g_{NN}(u(t-\tau_1), t-\tau_1; \phi)d_\theta u(t-\tau_1)\right. \\ &\left. +\partial_{u(t-\tau_2)} g_{NN}(u(t-\tau_2), t-\tau_2; \phi)d_\theta u(t-\tau_2)\right) dt\,. \end{aligned} \tag{D.18}$$

Using integration-by-parts, we can write,

$$\begin{aligned} \int_0^T \lambda^T(t)d_\theta d_t u(t)dt =&\lambda^T(T)d_\theta u(T) - \lambda^T(0)d_\theta u(0) - \int_0^T d_t\lambda^T(t)d_\theta u(t)dt \\ \int_0^T \mu^T(t)d_\theta d_t y(t)dt =&\mu^T(T)d_\theta y(T) - \mu^T(0)d_\theta y(0) - \int_0^T d_t\mu^T(t)d_\theta y(t)dt \end{aligned} \tag{D.19}$$

and by change of variables,

$$\begin{aligned}
&\int_0^T \mu^T(t)\partial_{u(t-\tau_i)} g_{NN}(u(t-\tau_i),t-\tau_i;\phi) d_\theta u(t-\tau_i) dt \\
&\qquad = \int_{-\tau_i}^{T-\tau_i} \mu^T(t+\tau_i)\partial_{u(t)} g_{NN}(u(t),t;\phi) d_\theta u(t) dt \\
&\qquad = \int_0^T \mu^T(t+\tau_i)\partial_{u(t)} g_{NN}(u(t),t;\phi) d_\theta u(t) dt \\
&\qquad\quad + \int_{-\tau_i}^0 \mu^T(t+\tau_i)\partial_{u(t)} g_{NN}(u(t),t;\phi) d_\theta u(t) dt \\
&\qquad\quad - \int_{T-\tau_i}^T \mu^T(t+\tau_i)\partial_{u(t)} g_{NN}(u(t),t;\phi) d_\theta u(t) dt \\
&\qquad = \int_0^T \mu^T(t+\tau_i)\partial_{u(t)} g_{NN}(u(t),t;\phi) d_\theta u(t) dt \\
&\qquad\quad + \int_{-\tau_i}^0 \mu^T(t+\tau_i)\partial_{u(t)} g_{NN}(u(t),t;\phi) d_\theta u(t) dt
\end{aligned} \tag{D.20}$$

We further assume $\mu(t) = 0,\ t \geq T$. Inserting everything back (Eqs. D.19 and D.20) into Eq. D.18, and keeping in mind that the initial condition $h(t)$ is independent of θ, we obtain,

$$\begin{aligned}
d_\theta L = &\int_0^T \Bigg(\sum_{i=1}^M \partial_{u(t)} l(u(t)) \delta(t - T_i) - d_t \lambda^T(t) - \lambda^T(t) \partial_{u(t)} f_{NN}(u(t), y(t), t; \theta) \\
&- \mu^T(t+\tau_1)\partial_{u(t)} g_{NN}(u(t),t;\phi) + \mu^T(t+\tau_2)\partial_{u(t)} g_{NN}(u(t),t;\phi) \Bigg) d_\theta u(t) dt \\
&- \int_0^T \lambda^T(t) \partial_\theta f_{NN}(u(t), y(t), t; \theta) dt \\
&+ \int_0^T (-d_t \mu^T(t) - \lambda^T(t) \partial_{y(t)} f_{NN}(u(t), y(t), t; \theta)) d_\theta y(t) dt \\
&+ \lambda^T(T) d_\theta u(T) + \mu^T(T) d_\theta y(T) \ .
\end{aligned} \tag{D.21}$$

Again, the objective is to avoid the need to compute $d_\theta u(t)$ and $d_\theta y(t)$, hence we assume, $\mu(t) = 0,\ t \geq T$; and $\lambda(T) = 0$. We can write the following coupled adjoint

equations,

$$\begin{aligned} d_t\lambda^T(t) =& \sum_{i=1}^{M} \partial_{u(t)} l(u(t))\delta(t-T_i) - \lambda^T(t)\partial_{u(t)} f_{NN}(u(t),y(t),t;\theta) \\ & - \mu^T(t+\tau_1)\partial_{u(t)} g_{NN}(u(t),t;\phi) \\ & + \mu^T(t+\tau_2)\partial_{u(t)} g_{NN}(u(t),t;\phi)\,, \qquad t\in[0,T) \\ d_t\mu^T(t) =& -\lambda^T(t)\partial_{y(t)} f_{NN}(u(t),y(t),t;\theta)\,, \qquad t\in[0,T) \\ \lambda^T(t) =& \ 0 \quad \text{and} \quad \mu^T(t) = 0, \qquad t\geq T\ . \end{aligned} \tag{D.22}$$

Finally after solving the coupled adjoint equations in $\lambda(t)$ and $\mu(t)$, we can compute the required derivative, $d_\theta L$ as,

$$d_\theta L = -\int_0^T \lambda^T(t)\partial_\theta f_{NN}(u(t),y(t),t;\theta)dt\,. \tag{D.23}$$

Now, we find the derivative of the Lagrangian w.r.t ϕ,

$$\begin{aligned} d_\phi L =& \partial_{u(t)}\mathcal{L}(u(t))d_\phi u(t) + \int_0^T \lambda^T(t)\left(d_\phi d_t u(t) - \partial_{u(t)} f_{NN}(u(t),y(t),t;\theta)d_\phi u(t)\right. \\ & \left. -\partial_{y(t)} f_{NN}(u(t),y(t),t;\theta)d_\phi y(t)\right)dt + \int_0^T \mu^T(t)\left(d_\phi d_t y(t)\right. \\ & -\partial_{u(t-\tau_1)} g_{NN}(u(t-\tau_1),t-\tau_1;\phi)d_\phi u(t-\tau_1) \\ & +\partial_{u(t-\tau_2)} g_{NN}(u(t-\tau_2),t-\tau_2;\phi)d_\phi u(t-\tau_2) \\ & \left. -\partial_\phi g_{NN}(u(t-\tau_1),t-\tau_1;\phi) + \partial_\phi g_{NN}(u(t-\tau_2),t-\tau_2;\phi)\right)dt \\ & + \alpha^T d_\phi y(0) - \alpha^T \int_{-\tau_2}^{-\tau_1} \partial_\phi g_{NN}(h(t),t;\phi)dt\,. \end{aligned} \tag{D.24}$$

using integration-by-parts and change of variables, we can write,

$$\begin{aligned}
&\int_0^T \lambda^T(t) d_\phi d_t u(t) dt = \lambda^T(T) d_\phi u(T) - \lambda^T(0) d_\phi u(0) - \int_0^T d_t \lambda^T(t) d_\phi u(t) dt \,. \\
&\int_0^T \mu^T(t) d_\phi d_t y(t) dt = \mu^T(T) d_\phi y(T) - \mu^T(0) d_\phi y(0) - \int_0^T d_t \mu^T(t) d_\phi y(t) dt \,. \\
&\int_0^T \mu^T(t) \partial_{u(t-\tau_i)} g_{NN}(u(t-\tau_i), t-\tau_i; \phi) d_\phi u(t-\tau_i) dt = \\
&\qquad\qquad \int_0^T \mu^T(t+\tau_i) \partial_{u(t)} g_{NN}(u(t), t; \phi) d_\phi u(t) dt \\
&\qquad\qquad + \int_{-\tau_i}^0 \mu^T(t+\tau_i) \partial_{u(t)} g_{NN}(u(t), t; \phi) d_\phi u(t) dt \,.
\end{aligned} \tag{D.25}$$

Substituting these in Eq. D.24, and keeping in mind that the initial condition $h(t)$ is independent of ϕ, we obtain,

$$\begin{aligned}
d_\phi L = &\int_0^T \Bigg(\sum_{i=1}^M \partial_{u(t)} l(u(t)) \delta(t - T_i) - d_t \lambda^T(t) - \lambda^T(t) \partial_{u(t)} f_{NN}(u(t), y(t), t; \theta) \\
&- \mu^T(t+\tau_1) \partial_{u(t)} g_{NN}(u(t), t; \phi) + \mu^T(t+\tau_2) \partial_{u(t)} g_{NN}(u(t), t; \phi) \Bigg) d_\phi u(t) dt \\
&+ \int_0^T \mu^T(t) \left(-\partial_\phi g_{NN}(u(t-\tau_1), t-\tau_1; \phi) + \partial_\phi g_{NN}(u(t-\tau_2), t-\tau_2; \phi) \right) dt \\
&+ \int_0^T (-d_t \mu^T(t) - \lambda^T(t) \partial_{y(t)} f_{NN}(u(t), y(t), t; \theta)) d_\phi y(t) dt \\
&- \mu^T(0) d_\phi y(0) + \alpha^T d_\phi y(0) - \alpha^T \int_{-\tau_2}^{-\tau_1} \partial_\phi g_{NN}(h(t), t; \phi) dt
\end{aligned} \tag{D.26}$$

As we already satisfy the adjoint equations (Eq. D.22), and letting $\alpha^T = \mu^T(0)$, we arrive at the expression for $d_\phi L$,

$$\begin{aligned}
d_\phi L = &- \int_0^T \mu^T(t) \left(\partial_\phi g_{NN}(u(t-\tau_1), t-\tau_1; \phi) - \partial_\phi g_{NN}(u(t-\tau_2), t-\tau_2; \phi) \right) dt \\
&- \mu^T(0) \int_{-\tau_2}^{-\tau_1} \partial_\phi g_{NN}(h(t), t; \phi) dt \,.
\end{aligned} \tag{D.27}$$

D.3 Experimental Setup

D.3.1 Architectures

In Tables D.1 & D.2 we provide architectural details of the various neural closure models used in the main text. We also provide the variation of training and validation loss with training epochs corresponding to these architectures in Fig. D-1. These results were picked among multiple repeats of training done with exactly the same hyperparameters (described next), with 3-5 repeats for discrete-nDDE in all the different experimental cases; 3-5 repeats for nODE and distributed-nDDE in experiments-1 & 3b; and 10-15 repeats for nODE and distributed-nDDE in experiments-2 & 3a (the same which were used for optimal delay length analysis).

D.3.2 Hyperparameters

The values of the various training hyperparameters used in the experiments are listed next. In all the experiments, the number of iterations per epoch are calculated by dividing the number of time-steps in the training period by batch-size multiplied the length of short time-sequences, adding 1, and rounding up to the next integer.

Experiments-1: For training, we randomly select short time-sequences spanning only 6 time-steps and extract data at every other time-step to form batches of size 2; 18 iterations per epoch; exponentially decaying learning rate (LR) schedule (initial LR of 0.075, decay rate of 0.97, and 18 decay steps); *RMSprop* optimizer; and end training at 200 epochs.

Experiments-2: We use a batch size of 8 created by randomly selecting short time-sequences spanning 6 time-steps and extracting data at every other time-step; 4 iterations per epoch; exponentially decaying learning rate (LR) schedule with initial LR of 0.075, decay rate of 0.97, and 4 decay steps; *RMSprop* optimizer; and end training at 250 epochs.

Experiments-3a: We use a batch size of 4 created by randomly selecting short time-sequences spanning 6 time-steps and extracting data at every other time-step; 26

Category	Experiment - 1				Experiment - 2			
	Architecture	**Act.**	**Delays**	**Trainable Parameters**	**Architecture**	**Act.**	**Delays**	**Trainable Parameters**
nODE (No-Delays)	f_{NN}		None	158	f_{NN}		None	424
	Input layer with 3 neurons	none			Input of size 25×1	none		
	5 FC hidden layer with 5 neurons	tanh			Conv1D layer with $KS = 3 \times 4$, $S = 1$	swish		
	FC output layer with 3 neurons	linear			4 Conv1D layer with $KS = 3 \times 5$, $S = 1$	swish		
					Conv1D-T layer with $KS = 3 \times 3$, $S = 1$	swish		
					3 Conv1D-T layer with $KS = 3 \times 2$, $S = 1$	swish		
					Conv1D-T output layer with $KS = 3 \times 1$, $S = 1$	linear		
Discrete-nDDE	f_{RNN}		$\tau_1 = 0.025$, $\tau_2 = 0.05$, ..., $\tau_6 = 0.15$	63	f_{RNN}		$\tau_1 = 0.025$, $\tau_2 = 0.05$, ..., $\tau_6 = 0.15$	110
	Input layer with 3 neurons	none			Input of size 25×1	none		
	Simple RNN layer with 5 neurons	tanh			Simple RNN Conv1D layer with $KS = 3 \times 3$, $S = 1$	swish		
	FC output layer with 3 neurons	linear			Conv1D layer with $KS = 3 \times 2$, $S = 1$	swish		
					Conv1D-T layer with $KS = 3 \times 2$, $S = 1$	swish		
					Conv1D-T output layer with $KS = 3 \times 1$, $S = 1$	linear		
Distributed-nDDE	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 0.075$	110	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 0.075$	361
	Input layer with 5 neurons	none			Input of size 25×2	none		
	2 FC hidden layer with 5 neurons	tanh			Conv1D layer with $KS = 3 \times 4$, $S = 1$	swish		
	FC output layer with 3 neurons	linear			2 Conv1D layer with $KS = 3 \times 5$, $S = 1$	swish		
					Conv1D-T layer with $KS = 3 \times 3$, $S = 1$	swish		
					Conv1D-T layer with $KS = 3 \times 2$, $S = 1$	swish		
					Conv1D-T output layer with $KS = 3 \times 1$, $S = 1$	linear		
	g_{NN}				g_{NN}			
	Input layer with 3 neurons	none			Input of size 25×1	none		
	2 FC hidden layer with 3 neurons	tanh			Conv1D layer with $KS = 3 \times 2$, $S = 1$	swish		
	FC output layer with 2 neurons	linear			Conv1D layer with $KS = 3 \times 3$, $S = 1$	swish		
					Conv1D-T layer with $KS = 3 \times 3$, $S = 1$	swish		
					Conv1D-T output layer with $KS = 3 \times 1$, $S = 1$	linear		

Table D.1: Architectures for different neural closure models used in Experiments-1 and 2. FC stands for fully-connected, Conv1D for convolutional-1D, and Conv1D-T for convolutional-1D transpose layers. The size of the convolutional layer filters is mentioned by the kernel size (KS; where the first dimension corresponds to the receptive field, and second to the number of channels), along with the number of strides (S).

Category	Experiment - 3a				Experiment - 3b			
	Architecture	**Act.**	**Delays**	**Trainable Parameters**	**Architecture**	**Act.**	**Delays**	**Trainable Parameters**
nODE (No-Delays)	f_{NN}		None	317	f_{NN}		None	987
	Input layer with 3 neurons	none			Input of size 20×3	none		
	6 FC hidden layer with 7 neurons	tanh			AddExtraChannels{z, I(z, t)}	none		
	FC hidden layer with 1 neuron	linear			Conv1D layers with $KS = 1 \times 5$, $S = 1$; $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 9$, $S = 1$; $KS = 1 \times 11$, $S = 1$; $KS = 1 \times 13$, $S = 1$; $KS = 1 \times 13$, $S = 1$; $KS = 1 \times 11$, $S = 1$; $KS = 1 \times 9$, $S = 1$; $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 5$, $S = 1$; $KS = 1 \times 3$, $S = 1$	swish		
	BioConstrainLayer with output of size 3	linear			Conv1D layer with $KS = 1 \times 1$, $S = 1$	linear		
					BioConstrainLayer with output of size 20×3	linear		
Discrete-nDDE	f_{RNN}		$\tau_1 = 0.75$, $\tau_2 = 1.5$, ..., $\tau_6 = 4.5$	142	f_{RNN}		$\tau_1 = 0.5$, $\tau_2 = 1.0$, $\tau_3 = 1.5$, $\tau_4 = 2.0$	426
	Input layer with 3 neurons	none			Input of size 20×3	none		
	Simple RNN layer with 7 neurons	tanh			Simple RNN Conv1D layer with $KS = 1 \times 5$, $S = 1$	swish		
	FC hidden layer with 7 neurons	tanh			AddExtraChannels{z, I(z, t)}	none		
	FC hidden layer with 1 neuron	linear			Conv1D layers with $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 9$, $S = 1$; $KS = 1 \times 9$, $S = 1$; $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 5$, $S = 1$; $KS = 1 \times 3$, $S = 1$	swish		
	BioConstrainLayer with output of size 3	linear			Conv1D layer with $KS = 1 \times 1$, $S = 1$	linear		
					BioConstrainLayer with output of size 20×3	linear		

Table D.2: Architectures for different neural closure models used in Experiments-3a and 3b. FC stands for fully-connected, and Conv1D for convolutional-1D layers. The size of the convolutional layer filters is mentioned by the kernel size (KS; where the first dimension corresponds to the receptive field, and second to the number of channels), along with the number of strides (S). While *AddExtraChannels* and *BioConstrainLayer* are custom layers described in the main text (Secs. 6.3.3 & 6.3.4). *(Cont.)*

Distributed-nDDE	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 2.5$	195	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 2.0$	477
	Input layer with 11 neurons	none			Input of size 20×5	none		
	2 FC hidden layer with 7 neurons	tanh			AddExtraChannels{z, I(z, t)}	none		
	FC hidden layer with 1 neurons	linear			Conv1D layers with $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 9$, $S = 1$; $KS = 1 \times 9$, $S = 1$; $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 5$, $S = 1$; $KS = 1 \times 3$, $S = 1$	swish		
	BioConstrainLayer with output of size 3	linear			Conv1D layer with $KS = 1 \times 1$, $S = 1$	linear		
					BioConstrainLayer with output of size 20×3	linear		
	g_{NN}				g_{NN}			
	Input layer with 3 neurons	none			Input of size 20×3	none		
	2 FC hidden layer with 5 neurons	tanh			Conv1D layers with $KS = 1 \times 3$, $S = 1$; $KS = 1 \times 5$, $S = 1$; $KS = 1 \times 7$, $S = 1$; $KS = 1 \times 5$, $S = 1$	swish		
	FC output layer with 4 neurons	linear			Conv1D output layer with $KS = 1 \times 2$, $S = 1$	linear		

Table D.2: Architectures for different neural closure models used in Experiments-3a and 3b. FC stands for fully-connected, and Conv1D for convolutional-1D layers. The size of the convolutional layer filters is mentioned by the kernel size (KS; where the first dimension corresponds to the receptive field, and second to the number of channels), along with the number of strides (S). While *AddExtraChannels* and *BioConstrainLayer* are custom layers described in the main text (Secs. 6.3.3 & 6.3.4).

iterations per epoch; exponentially decaying learning rate (LR) schedule with initial LR of 0.05, decay rate of 0.97, and 26 decay steps; *RMSprop* optimizer; and end training at 350 epochs.

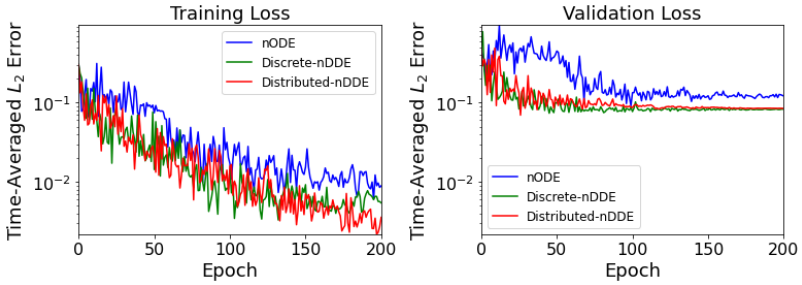
Experiments-3b: We use a batch size of 8 (4 for only distributed-nDDE) created by randomly selecting short time-sequences spanning 6 time-steps and extracting data at every other time-step; 8 (14 for only distributed-nDDE) iterations per epoch; exponentially decaying learning rate (LR) schedule with initial LR of 0.05, decay rate of 0.97, and 8 (14 for only distributed-nDDE) decay steps; *RMSprop* optimizer; and end training at 200 epochs.

D.3.3 Sensitivity to Network Size and Training Period Length

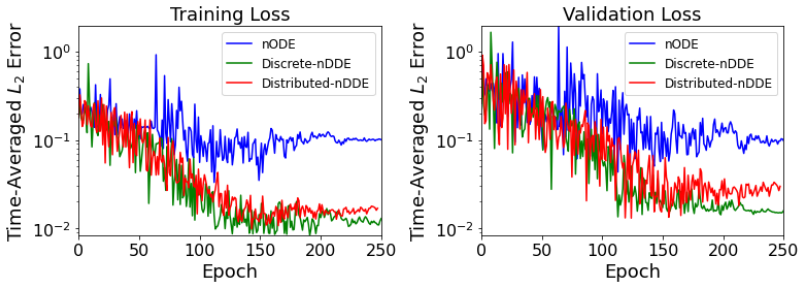
We performed various hyperparameter studies for all the different experimental cases presented in the main paper. However, here we only show the effect of network size and training period length on the performance of experiments-1 with distributed-nDDE closure.

First, we varied the length of the training period while keeping the architecture and other hyperparameters the same (as mentioned above, Secs. D.3.1 & D.3.2). We chose 5 different training lengths (all starting from $t = 0$), with the longest encompassing one time-period for the coefficient corresponding to mode 3. In Fig. D-2a, we provide the root-mean-square-error (RMSE) as it evolves with time, spanning the training, validation, and prediction periods. We can notice, that in each case, the network is able to exactly match the true coefficients upto the end of training period. While the long-term performance drastically improves on providing more-and-more training data by increasing the training period length.

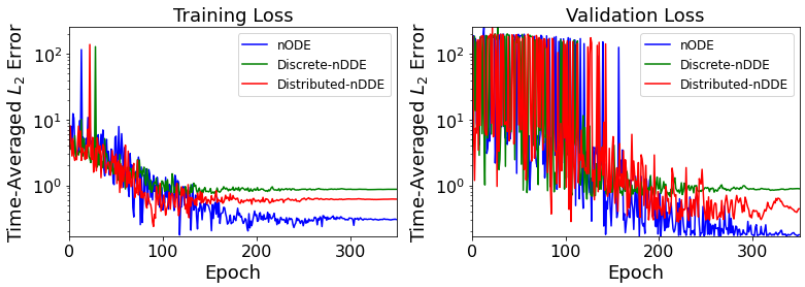
Second, we only vary the depth of the network, while keeping all other training details exactly the same (as mentioned above, Sec. D.3.2). We chose three different network sizes by changing the number of hidden layers, with architectural details presented in Table D.3. Often, smaller networks are underparameterized limiting their expressivity, while overly large networks might become overparameterized limiting their generalizability for predictions. In Fig. D-2b, we provide variation of training



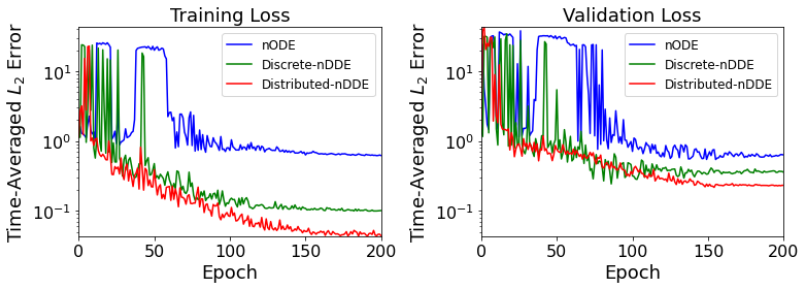
(a) Experiment-1



(b) Experiment-2



(c) Experiment-3a



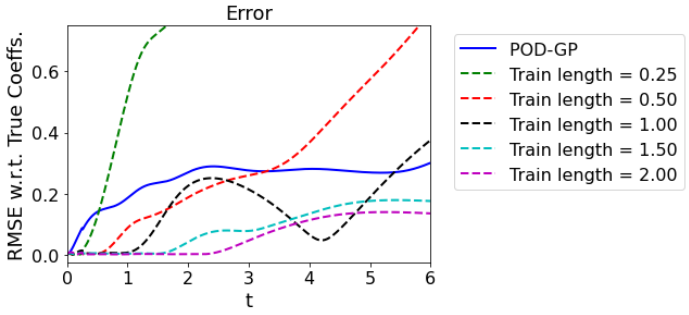
(d) Experiment-3b

Figure D-1: Variation with epochs of training (*left column*), and validation (*right column*) time-averaged L_2 loss for the three neural closure models, while training for each of the Experiments-1, 2, 3a, and 3b. These results accompany Figs. 6-3, 6-5, 6-8, & 6-9 in the main text, and the architectures detailed in Tables D.1 & D.2 .

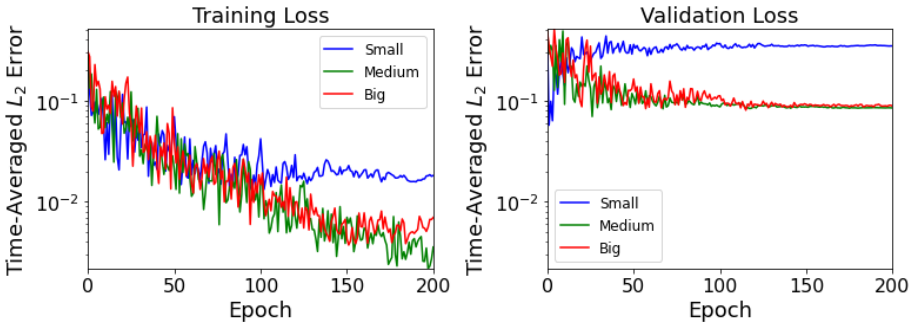
Category	**Architecture**	**Act.**	**Delays**	**Trainable Parameters**
Small	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 0.075$	38
	Input layer with 5 neurons	none		
	FC output layer with 3 neurons	linear		
	g_{NN}			
	Input layer with 3 neurons	none		
	FC hidden layer with 3 neurons	tanh		
	FC output layer with 2 neurons	linear		
Medium	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 0.075$	110
	Input layer with 5 neurons	none		
	2 FC hidden layer with 5 neurons	tanh		
	FC output layer with 3 neurons	linear		
	g_{NN}			
	Input layer with 3 neurons	none		
	2 FC hidden layer with 3 neurons	tanh		
	FC output layer with 2 neurons	linear		
Big	f_{NN}		$\tau_1 = 0.0$, $\tau_2 = 0.075$	152
	Input layer with 5 neurons	none		
	3 FC hidden layer with 5 neurons	tanh		
	FC output layer with 3 neurons	linear		
	g_{NN}			
	Input layer with 3 neurons	none		
	3 FC hidden layer with 3 neurons	tanh		
	FC output layer with 2 neurons	linear		

Table D.3: Architectures of different sizes for distributed-nDDE used in hyperparameter sensitivity study for Experiments-1.

and validation losses with training epochs, and can notice that the small network struggles to close the system.



(a) Effect on performance for change in training period length



(b) Change in time-average L_2 loss with change in the network size

Figure D-2: Experiments-1 sensitivity to network size and training period length. (a): Evolution of root-mean-squared-error (RMSE) of coefficients for distributed-nDDEs trained with different training period length, and with same architectures and other hyperparameter values. These results correspond to the distributed-nDDE architecture detailed in Table D.1. (b): Variation with epochs of training (*left*), and validation (*right*) time-averaged L_2 loss for the three different sized distributed-nDDE architectures detailed in Table D.3.

Appendix E

Learning the Optimal Delay for Neural Closure Models

In the earlier developed framework of *neural closure models* (nCMs; [229]), we were successfully able to use the neural delay differential equations (nDDEs) to learn non-Markovian closure parameterizations for known-physics/low-fidelity models. Using a series of experiments we demonstrated the existence of an optimal amount of past information to incorporate for a specified architecture which was related to the relevant time scaled present in the dynamical system. The delay-period lengths in the existing framework are treated as hyperparameters, which in-turn are tuned by brute-force search based on performance over the validation time-period. An initial estimate of the range of delay-period lengths to consider can be obtained from properties of the given dynamical system such as the main time scales, e.g. physical and biological times scales, and main decorrelation times of state variables. The estimation of time-delays has always been elusive and impaired the use of delay differential equations for real-world systems. A survey of the existing time-delay estimation methods can be found in Björklund, 2003 [256], however they are limited by various model assumptions, thus limiting their applicability to our framework.

In order to seamlessly estimate the optimal delay-period length, we propose to learn it from the data along with the other trainable neural-network weights. We evaluate the performance of the optimal delay learning extension of our existing frame-

work using a series of experiments consisting of a two-variable system with known delay, and the advecting shock problem governed by the Burgers equation (section 6.3.2).

E.1 Theory and Methodology

We will limit ourselves to the case of nDDEs with distributed delays and $\tau_1 = 0$ fixed. Thus, apart from the neural-network weights, we will have $\tau_2 \equiv \tau$ as a trainable parameter. Based on equation 6.14, the considered distributed-nDDE is given by,

$$\begin{aligned} \frac{du(t)}{dt} &= f_{NN}\left(u(t), y(t), t; \theta\right), \quad t \in (0, T] \\ \frac{dy(t)}{dt} &= g_{NN}(u(t), t; \phi) - g_{NN}(u(t-\tau), t-\tau; \phi), \quad t \in (0, T] \\ \text{with} \quad u(t) &= h(t), \quad \tau \leq t \leq 0\,, \quad \text{and} \quad y(0) = \int_{-\tau}^{0} g_{NN}(h(t), t; \phi) dt\,. \end{aligned} \tag{E.1}$$

Again assuming a scalar loss function given by, $\mathcal{L} = \int_0^T \sum_{i=1}^M l(u(t))\delta(t - T_i)dt$, for the available data at M times, $0 \leq T_1 < ... < T_M \leq T$, the Lagrangian for the above system is,

$$\begin{aligned} L =& \mathcal{L}(u(t)) + \int_0^T \lambda^T(t)(d_t u(t) - f_{NN}(u(t), y(t), t; \theta))\, dt \\ &+ \int_0^T \mu^T(t)\left(d_t y(t) - g_{NN}(u(t), t; \phi) + g_{NN}(u(t-\tau), t-\tau; \phi)\right) dt \\ &+ \int_{-\tau}^0 \gamma^T(t)(u(t) - h(t))dt + \alpha^T \left(y(0) - \int_{-\tau}^0 g_{NN}(h(t), t; \phi)dt\right). \end{aligned} \tag{E.2}$$

Adjoint equations for the Lagrangian variables, $\lambda(t)$ and $\mu(t)$, and gradients for the neural-network weights, $d_\theta L$ and $d_\phi L$, are the same as derived earlier, equation 6.16 with $\tau_1 = 0$ and $\tau_2 \equiv \tau$. However, in the current section, the goal is to additionally

derive $d_\tau L$. Taking the derivative of equation E.2 w.r.t. τ, we get,

$$\begin{aligned} d_\tau L =& \partial_{u(t)}\mathcal{L}(u(t))d_\tau u(t) + \int_0^T \lambda^T(t)\left(d_\tau d_t u(t) - \partial_{u(t)} f_{NN}(u(t), y(t), t; \theta) d_\tau u(t)\right. \\ & \left. - \partial_{y(t)} f_{NN}(u(t), y(t), t; \theta) d_\tau y(t)\right) dt \\ & + \int_0^T \mu^T(t)\left(d_\tau d_t y(t) - \partial_{u(t)} g_{NN}(u(t), t; \phi) d_\tau u(t)\right. \\ & \left. + \partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi) d_\tau u(t-\tau) + \partial_{t-\tau} g_{NN}(u(t-\tau), t-\tau; \phi) d_\tau (t-\tau)\right) dt \\ & + \alpha^T \left(d_\tau y(0) + g_{NN}(h(-\tau), -\tau; \phi) d_\tau(-\tau)\right) , \end{aligned} \tag{E.3}$$

where, for brevity of notation, we use $u(t) \equiv u(t, \tau, \theta, \phi)$ and $y(t) \equiv y(t, \tau, \theta, \phi)$). Using integration-by-parts, we can write,

$$\begin{aligned} \int_0^T \lambda^T(t) d_\tau d_t u(t) dt =& \lambda^T(T) d_\tau u(T) - \lambda^T(0) d_\tau u(0) - \int_0^T d_t \lambda^T(t) d_\tau u(t) dt , \\ \int_0^T \mu^T(t) d_\tau d_t y(t) dt =& \mu^T(T) d_\tau y(T) - \mu^T(0) d_\tau y(0) - \int_0^T d_t \mu^T(t) d_\tau y(t) dt , \end{aligned} \tag{E.4}$$

and recall that $\lambda(T) = 0$, $d_\tau u(0) = 0$, and $\mu(T) = 0$. Further using a change of variable yields,

$$\begin{aligned} & \int_0^T \mu^T(t) \partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi) d_\tau u(t-\tau) dt \\ & = \int_0^T \mu^T(t) \partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi) \left(\partial_{t-\tau} u(t-\tau) d_\tau (t-\tau) + \partial_\tau u(t-\tau)\right) dt . \end{aligned} \tag{E.5}$$

Notice, $\partial_{t-\tau}u(t-\tau) \equiv d_t u(t-\tau)$, and if $d_\tau u(t) = z(t)$, then $\partial_\tau u(t-\tau) = z(t-\tau)$. Additionally, letting $s = t - \tau$, we get,

$$\begin{aligned}
&\int_0^T \mu^T(t)\partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi)\left(\partial_{t-\tau}u(t-\tau)d_\tau(t-\tau) + \partial_\tau u(t-\tau)\right)dt \\
&= \int_0^T \mu^T(t)\partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi)\left(-d_t u(t-\tau) + z(t-\tau)\right)dt \\
&= \int_0^T \mu^T(t)\partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi)\left(-d_t u(t-\tau)\right)dt \\
&+ \int_{-\tau}^{T-\tau} \mu^T(s+\tau)\partial_{u(s)} g_{NN}(u(s), s; \phi) z(s) ds \\
&= \int_0^T \mu^T(t)\partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi)\left(-d_t u(t-\tau, \tau)\right)dt \\
&+ \int_0^T \mu^T(t+\tau)\partial_{u(t)} g_{NN}(u(t), t; \phi) z(t) dt\ .
\end{aligned} \tag{E.6}$$

Plugging everything back, and using the adjoint equations, we are left with,

$$\begin{aligned}
d_\tau L &= \int_0^T \mu^T(t)\partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi)\left(-d_t u(t-\tau)\right)dt \\
&\quad - \mu^T(0) g_{NN}(h(-\tau), -\tau; \phi) - \int_0^T \mu^T(t)\partial_{t-\tau} g_{NN}(u(t-\tau), t-\tau; \phi) dt \\
&= -\int_0^T \mu^T(t)\partial_{u(t-\tau)} g_{NN}(u(t-\tau), t-\tau; \phi) d_t u(t-\tau) dt \\
&\quad - \mu^T(0) g_{NN}(h(-\tau), -\tau; \phi) - \int_0^T \mu^T(t)\partial_{t-\tau} g_{NN}(u(t-\tau), t-\tau; \phi) dt\,.
\end{aligned} \tag{E.7}$$

The above expression can be used in the stochastic gradient descent step to update τ.

E.2 Application Results and Discussion

Next, we use a series of experiments to demonstrate the feasibility of learning the optimal delay value simultaneously with the weights of the neural-networks.

E.2.1 Experiments 1: 2D Spiral

In the first set of experiments, we use a simple two-variable dynamical system with distributed delays, given by,

$$\begin{bmatrix} d_t u_1(t) \\ d_t u_2(t) \end{bmatrix} = \begin{bmatrix} -0.1 & 1.0 \\ -1.0 & -0.1 \end{bmatrix}^T \left(\begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} - \int_{t-1.9}^{t} \begin{bmatrix} 0.1 & -1.0 \\ -1.0 & 0.1 \end{bmatrix}^T \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} dt \right) \quad \text{(E.8)}$$

The solution of the above DDE system with initial conditions, $u_1(0) = 1.0$ and $u_2(0) = 0.0$, is presented in figure E-1. We use the *dopri5* [134] scheme for time-integration. For our learning experiments, we assume that the delay value and entries of the matrices in equation E.8 are unknown. Further, training data is available up until $t = 40$ at every 0.05 time-steps. The system with unknown parameters and delay takes the form,

$$\begin{bmatrix} d_t u_1(t) \\ d_t u_2(t) \end{bmatrix} = A_1 \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} + A_2 \int_{t-\tau}^{t} B \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} dt \quad \text{(E.9)}$$

where A_1, A_2, B, and τ will be learned from the training data. The initial conditions are the same as the true system. For training, we randomly select short time sequences spanning 75 time-steps (batch-time) and extract data at every other time-step to form batches of size 2; we use 6 iterations per epoch; we use an exponentially decaying learning rate (LR) schedule with initial LR of 0.075, decay rate of 0.97, and 6 decay steps; the RMSprop optimizer is used; we train for a total of 200 epochs. We further bound the value of τ with an upper limit of 3.0 and use L_2 regularization with a factor of 0.0005. We learned $\tau_{learned} = 1.92$, and on repeated training with slightly different hyperparameters, it remained within 2% of the true delay value. Prediction up until time $t = 8.0$ using the learned system is presented in figure E-1 for comparison with the truth.

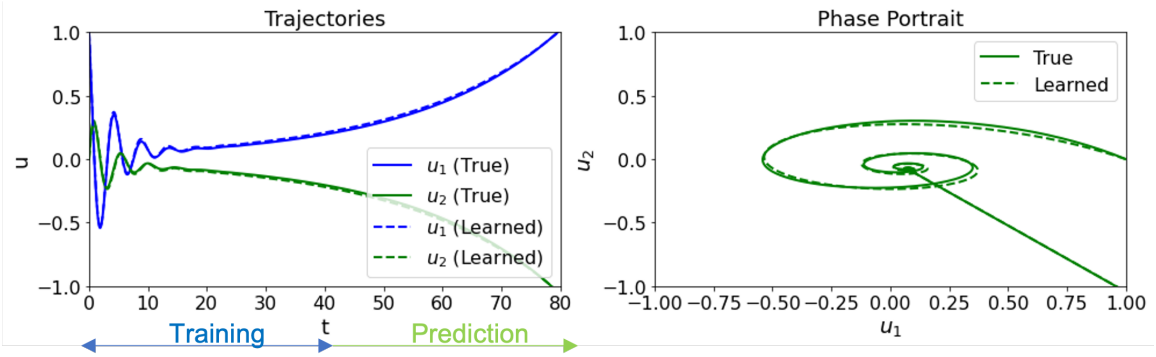


Figure E-1: Comparison of the true and learned two-variable dynamical system defined by equations E.8 & E.9 respectively. The left plot provides the temporal trajectories of the two state variables and the right plot provides the corresponding phase portrait. We train using data only up until $t = 40$ and make predictions from $t = 40$ to $t = 80$.

E.2.2 Experiments 2: Advecting shock - subgrid-scale processes

In the second set of experiments we again use the setup of the advecting shock problem governed by the Burgers equation from earlier (section 6.3.2). Keeping everything the same, such as the architecture, low-fidelity model, high-fidelity data, and barring some of the hyperparameters, our goal is to additionally learn the delay value for the distributed-nDDE closure. We perform 8 identical training experiments, with a batch-time of 75 time-steps, extracting data at every other time-step to form batches of size 2, with 3 iterations per epoch. We use an upper limit of 1.0 for τ, and an L_2 regularization with a factor of 0.0001. The evolution of $\tau_{learned}$ as training progresses is provided in figure E-2. It can be noted that the learned delay value converges to ~ 0.4 which is of the order of the advection time-scale in the problem. The batch-time hyperparameter had the most impact on convergence of the delay value. For the batch-time, a value greater than a factor of 10 was needed compared to the experiments in which delay value was an user-defined hyperparameter (section 6.3.2). During every iteration, the discrepancy between the predicted state and the true / high-fidelity state helped in computing the gradients needed to update the trainable parameters. Further, it takes longer time-integration for discrepancies to

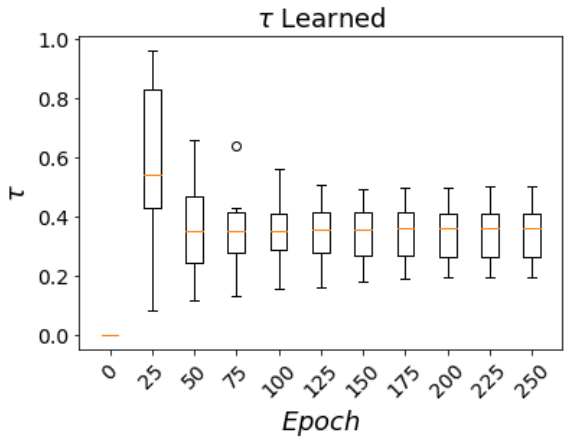


Figure E-2: The evolution of the learned delay value as a function of training epoch for the distributed-nDDE closure used for learning subgrid-scale processes in Burgers' equation. We use boxplots to provide statistical summaries for multiple training repeats done for the same set of hyperparameters.

sneak in due to a small change in the delay value as compared to the weights of the neural-networks, thus requiring a larger batch-time.

Appendix F

Supplementary Information: Generalized Neural Closure Models with Interpretability

F.1 Adjoint Equations for Neural Partial Delay Differential Equations

In this section we provide a detailed derivation of adjoint equations for neural partial delay differential equations (nPDDEs). The below derivation is inspired by the adjoint equation derivation for a general PDE by Li and Petzold, 2004 [257] and Cao et al., 2002 [258]. Our nPDDE is of the form,

$$\begin{aligned}\frac{\partial u(x,t)}{\partial t} =& \mathcal{F}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., \frac{\partial^d u(x,t)}{\partial x^d}, x, t; \phi\right) \\ &+ \int_{t-\tau}^{t} \mathcal{G}_{NN}\left(u(x,s), \frac{\partial u(x,s)}{\partial x}, \frac{\partial^2 u(x,s)}{\partial x^2}, ..., \frac{\partial^d u(x,s)}{\partial x^d}, x, s; \theta\right) ds\,, \\ & \qquad\qquad x \in \Omega,\ t \geq 0, \\ u(x,t) = h(x,t),\ t \leq 0 \quad &\text{and} \quad \mathcal{B}(u(x,t)) = g(x,t),\ x \in \partial\Omega,\ t \geq 0\,.\end{aligned} \tag{F.1}$$

As compared to PDEs, PDDEs require specification of a history function ($h(x,t)$) for the initial conditions. $\mathcal{F}_{NN}(\bullet;\phi)$ and $\mathcal{G}_{NN}(\bullet;\theta)$ are two neural networks (NNs) parameterized by ϕ and θ, respectively. We consider the presence of an arbitrary number of spatial derivatives, with the highest order defined by $d \in \mathbb{Z}^+$. We can rewrite the above equation F.1 as an equivalent system of coupled PDDEs with discrete delays,

$$\begin{aligned}
\frac{\partial u(x,t)}{\partial t} =& \mathcal{F}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., \frac{\partial^d u(x,t)}{\partial x^d}, x, t; \phi\right) + y(x,t)\ , \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x \in \Omega,\ t \geq 0, \\
\frac{\partial y(x,t)}{\partial t} =& \mathcal{G}_{NN}\left(u(x,t), \frac{\partial u(x,t)}{\partial x}, \frac{\partial^2 u(x,t)}{\partial x^2}, ..., \frac{\partial^d u(x,t)}{\partial x^d}, x, t; \theta\right) \\
& - \mathcal{G}_{NN}\left(u(x,t-\tau), \frac{\partial u(x,t-\tau)}{\partial x}, \frac{\partial^2 u(x,t-\tau)}{\partial x^2}, ..., \frac{\partial^d u(x,t-\tau)}{\partial x^d}, x, t-\tau; \theta\right)\ , \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x \in \Omega,\ t \geq 0, \\
u(x,t) =& h(x,t), t \leq 0 \quad \text{and} \quad \mathcal{B}(u(x,t)) = g(x,t), x \in \partial\Omega, t \geq 0 \\
y(x,0) =& \int_{-\tau}^{0} \mathcal{G}_{NN}\left(h(x,s), \frac{\partial h(x,s)}{\partial x}, \frac{\partial^2 h(x,s)}{\partial x^2}, ..., \frac{\partial^d h(x,s)}{\partial x^d} x, s; \theta\right) ds\,.
\end{aligned} \tag{F.2}$$

Let us assume that high-fidelity data is available at M discrete times, $T_1 < ... < T_M \leq T$, and at $N(T_i)$ spatial locations ($x_k^{T_i} \in \Omega, \forall k \in 1, ..., N(T_i)$) for each of the times. We define the scalar loss function as $L = \frac{1}{M}\sum_{i=1}^{M} \frac{1}{N(T_i)} \sum_{k=1}^{N(T_i)} l(u(x_k^{T_i}, T_i)) \equiv \int_0^T \frac{1}{M}\sum_{i=1}^{M} \int_\Omega \frac{1}{N(T_i)} \sum_{k=1}^{N(T_i)} l(u(x,t))\delta(x - x_k^{T_i})\delta(t - T_i)dxdt \equiv \int_0^T \frac{1}{M}\sum_{i=1}^{M} \frac{1}{|\Omega|} \int_\Omega \hat{l}(u(x,t))\delta(t - T_i)dxdt$, where $l(\bullet)$ are scalar loss functions such as mean-squared-error (MSE), and $\delta(\bullet)$ is the Kronecker delta function. In order to derive the adjoint PDEs, we start with the Lagrangian corresponding to the above system,

$$\begin{aligned}
\mathbb{L} =& L(u(x,t)) + \int_0^T \int_\Omega \lambda^T(x,t) \left(\partial_t u(x,t) - \mathcal{F}_{NN}(\bullet, t; \phi) - y(x,t)\right) dxdt \\
& + \int_0^T \int_\Omega \mu^T(x,t) \left(\partial_t y(x,t) - \mathcal{G}_{NN}(\bullet, t; \theta) + \mathcal{G}_{NN}(\bullet, t-\tau; \theta)\right) dxdt \\
& + \int_\Omega \alpha^T(x) \left(y(x,0) - \int_{-\tau}^{0} \mathcal{G}_{NN}(h(x,t), \partial_x h(x,t), \partial_{x^2} h(x,t), ..., \partial_{x^d} h(x,t), x, t; \theta) dt\right) dx\,,
\end{aligned} \tag{F.3}$$

where $\lambda(x,t)$, $\mu(x,t)$ and $\alpha(x)$ are the Lagrangian variables. We start by taking the derivative of the Lagrangian (equation F.3) w.r.t. θ (for brevity we denote, $\partial/\partial(\bullet) \equiv \partial_{(\bullet)}$, and $d/d(\bullet) \equiv d_{(\bullet)}$),

$$
\begin{aligned}
d_\theta \mathbb{L} = \int_0^T \int_\Omega \frac{1}{M} \frac{1}{|\Omega|} \sum_{i=1}^{M} \partial_{u(x,t)} \hat{l}(u(x,t)) \delta(t - T_i) d_\theta u(x,t) dx dt + \int_0^T \int_\Omega \lambda^T(x,t) \left(\partial_t d_\theta u(x,t) \right. \\
- \partial_{u(x,t)} \mathcal{F}_{NN}(\bullet, t; \phi) d_\theta u(x,t) - \partial_{\partial_x u(x,t)} \mathcal{F}_{NN}(\bullet, t; \phi) d_\theta \partial_x u(x,t) \\
\left. - \partial_{\partial_{xx} u(x,t)} \mathcal{F}_{NN}(\bullet, t; \phi) d_\theta \partial_{xx} u(x,t) - d_\theta y(x,t) \right) dx dt + \int_0^T \int_\Omega \mu^T(x,t) \left(\partial_t d_\theta y(x,t) \right. \\
- \partial_{u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta u(x,t) - \partial_{\partial_x u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta \partial_x u(x,t) \\
- \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta \partial_{xx} u(x,t) \\
- \partial_\theta \mathcal{G}_{NN}(\bullet, t; \theta) + \partial_{u(x,t)} \mathcal{G}_{NN}(\bullet, t - \tau; \theta) d_\theta u(x, t - \tau) \\
+ \partial_{\partial_x u(x,t-\tau)} \mathcal{G}_{NN}(\bullet, t - \tau; \theta) d_\theta \partial_x u(x, t - \tau) \\
\left. + \partial_{\partial_{xx} u(x,t-\tau)} \mathcal{G}_{NN}(\bullet, t - \tau; \theta) d_\theta \partial_{xx} u(x, t - \tau) + \partial_\theta \mathcal{G}_{NN}(\bullet, t - \tau; \theta) \right) dx dt \\
+ \int_\Omega \alpha^T(x) \left(d_\theta y(x,0) - \int_{-\tau}^{0} \partial_\theta G_{NN}(h(x,t), \partial_x h(x,t), \partial_{xx} h(x,t), x, t; \theta) dt \right) dx \,.
\end{aligned}
\tag{F.4}
$$

Using integration-by-parts, we get,

$$
\int_0^T \int_\Omega \lambda^T(x,t) \partial_t d_\theta u(x,t) dx dt = \int_\Omega [\lambda^T(x,t) d_\theta u(x,t)] \Big|_0^T - \int_0^T \int_\Omega \partial_t \lambda^T(x,t) d_\theta u(x,t) dx dt \,,
\tag{F.5}
$$

$$
\begin{aligned}
\int_0^T \int_\Omega \lambda^T(x,t) \partial_{\partial_x u(x,t)} \mathcal{F}_{NN}(\bullet, t) \partial_x d_\theta u(x,t) dx dt \\
= \int_0^T \int_{\partial\Omega} \lambda^T(x,t) \partial_{\partial_x u(x,t)} \mathcal{F}_{NN}(\bullet, t) d_\theta u(x,t) dx dt \\
- \int_0^T \int_\Omega \partial_x \left(\lambda^T(x,t) \partial_{\partial_x u(x,t)} \mathcal{F}_{NN}(\bullet, t) \right) d_\theta u(x,t) dx dt \,,
\end{aligned}
\tag{F.6}
$$

$$
\begin{aligned}
&\int_0^T \int_\Omega \lambda^T(x,t)\partial_{\partial_{xx}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\partial_{xx}d_\theta u(x,t)dxdt \\
&\qquad = \int_0^T \int_{\partial\Omega} \lambda^T(x,t)\partial_{\partial_{xx}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\partial_x d_\theta u(x,t)dxdt \\
&\qquad\quad - \int_0^T \int_\Omega \partial_x \left(\lambda^T(x,t)\partial_{\partial_{xx}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right)\partial_x d_\theta u(x,t)dxdt \\
&\qquad = \int_0^T \int_{\partial\Omega} \lambda^T(x,t)\partial_{\partial_{xx}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\partial_x d_\theta u(x,t)dxdt \\
&\qquad\quad - \left(\int_0^T \int_{\partial\Omega} \partial_x \left(\lambda^T(x,t)\partial_{\partial_{xx}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right) d_\theta u(x,t)dxdt \right. \\
&\qquad\quad \left. - \int_0^T \int_\Omega \partial_{xx} \left(\lambda^T(x,t)\partial_{\partial_{xx}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right) d_\theta u(x,t)dxdt \right) ,
\end{aligned}
\tag{F.7}
$$

$$
\begin{aligned}
&\int_0^T \int_\Omega \mu^T(x,t)\partial_{u(x,t-\tau)}\mathcal{G}_{NN}(\bullet,t-\tau)d_\theta u(x,t-\tau)dxdt \\
&\qquad = \int_{-\tau}^{T-\tau} \int_\Omega \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t)d_\theta u(x,t)dxdt \\
&\qquad = \int_0^T \int_\Omega \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t)d_\theta u(x,t)dxdt \\
&\qquad\quad - \int_{T-\tau}^T \int_\Omega \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t)d_\theta u(x,t)dxdt \\
&\qquad\quad + \int_{-\tau}^0 \int_\Omega \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t)d_\theta u(x,t)dxdt \,.
\end{aligned}
\tag{F.8}
$$

Using the fact that $\mu^T(x,t) = 0, \forall t \geq T$ and $d_\theta u(x,t) = 0, \forall t \leq 0$, we get,

$$
\begin{aligned}
&\int_0^T \int_\Omega \mu^T(x,t)\partial_{u(x,t-\tau)}\mathcal{G}_{NN}(\bullet,t-\tau;\theta)d_\theta u(x,t-\tau)dxdt \\
&\qquad = \int_0^T \int_\Omega \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)d_\theta u(x,t)dxdt \,.
\end{aligned}
\tag{F.9}
$$

Similarly, we also get,

$$\begin{aligned}\int_0^T \int_\Omega \mu^T(x,t)\partial_{\partial_x u(x,t-\tau)}\mathcal{G}_{NN}(\bullet, t-\tau;\theta)\partial_x d_\theta u(x,t-\tau)dxdt \\ = \int_0^T \int_{\partial\Omega} \mu^T(x,t+\tau)\partial_{\partial_x u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)d_\theta u(x,t)dxdt \\ - \int_0^T \int_\Omega \partial_x\left(\mu^T(x,t+\tau)\partial_{\partial_x u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right)d_\theta u(x,t)dxdt\,,\end{aligned} \tag{F.10}$$

$$\begin{aligned}\int_0^T \int_\Omega \mu^T(x,t)\partial_{\partial_{xx} u(x,t-\tau)}\mathcal{G}_{NN}(\bullet, t-\tau;\theta)\partial_{xx} d_\theta u(x,t-\tau)dxdt \\ = \int_0^T \int_{\partial\Omega} \mu^T(x,t+\tau)\partial_{\partial_{xx} u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\partial_x d_\theta u(x,t)dxdt \\ - \left(\int_0^T \int_{\partial\Omega} \partial_x\left(\mu^T(x,t+\tau)\partial_{\partial_{xx} u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right)d_\theta u(x,t)dxdt\right. \\ \left. - \int_0^T \int_\Omega \partial_{xx}\left(\mu^T(x,t+\tau)\partial_{\partial_{xx} u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right)d_\theta u(x,t)dxdt\right)\,.\end{aligned} \tag{F.11}$$

Plugging everything in yields,

$$\begin{aligned}d_\theta\mathbb{L} = &\int_0^T \int_\Omega \frac{1}{M}\frac{1}{|\Omega|}\sum_{i=1}^M \partial_{u(x,t)}\hat{l}(u(x,t))\delta(t-T_i)d_\theta u(x,t)dxdt \\ &- \int_0^T \int_\Omega \partial_t\lambda^T(x,t)d_\theta u(x,t)dxdt \\ &- \int_0^T \int_\Omega \lambda^T(x,t)\partial_{u(x,t)}\mathcal{F}_{NN}(\bullet,t)d_\theta u(x,t)dxdt \\ &- \int_0^T \int_{\partial\Omega} \lambda^T(x,t)\partial_{\partial_x u(x,t)}\mathcal{F}_{NN}(\bullet,t)d_\theta u(x,t)dxdt \\ &+ \int_0^T \int_\Omega \partial_x\left(\lambda^T(x,t)\partial_{\partial_x u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right)d_\theta u(x,t)dxdt \\ &- \int_0^T \int_{\partial\Omega} \lambda^T(x,t)\partial_{\partial_{xx} u(x,t)}\mathcal{F}_{NN}(\bullet,t)\partial_x d_\theta u(x,t)dxdt \\ &+ \int_0^T \int_{\partial\Omega} \partial_x\left(\lambda^T(x,t)\partial_{\partial_{xx} u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right)d_\theta u(x,t)dxdt \\ &- \int_0^T \int_\Omega \partial_{xx}\left(\lambda^T(x,t)\partial_{\partial_{xx} u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right)d_\theta u(x,t)dxdt\end{aligned}$$

$$
\begin{aligned}
&- \int_0^T \int_\Omega \lambda^T(x,t) d_\theta y(x,t) dxdt \\
&- \int_\Omega \mu^T(x,0) d_\theta y(x,0) dx - \int_0^T \int_\Omega \partial_t \mu^T(x,t) d_\theta y(x,t) dxdt \\
&- \int_0^T \int_\Omega \mu^T(x,t) \partial_{u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta u(x,t) dxdt \\
&- \int_0^T \int_{\partial\Omega} \mu^T(x,t) \partial_{\partial_x u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta u(x,t) dxdt \\
&+ \int_0^T \int_\Omega \partial_x \left(\mu^T(x,t) \partial_{\partial_x u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \right) d_\theta u(x,t) dxdt \\
&- \int_0^T \int_{\partial\Omega} \mu^T(x,t) \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \partial_x d_\theta u(x,t) dxdt \\
&+ \int_0^T \int_{\partial\Omega} \partial_x \left(\mu^T(x,t) \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \right) d_\theta u(x,t) dxdt \\
&- \int_0^T \int_\Omega \partial_{xx} \left(\mu^T(x,t) \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \right) d_\theta u(x,t) dxdt \\
&- \int_0^T \int_\Omega \mu^T(x,t) \partial_\theta \mathcal{G}_{NN}(\bullet, t; \theta) dxdt \\
&+ \int_0^T \int_\Omega \mu^T(x,t+\tau) \partial_{u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta u(x,t) dxdt \\
&+ \int_0^T \int_{\partial\Omega} \mu^T(x,t+\tau) \partial_{\partial_x u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) d_\theta u(x,t) dxdt \\
&- \int_0^T \int_\Omega \partial_x \left(\mu^T(x,t+\tau) \partial_{\partial_x u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \right) d_\theta u(x,t) dxdt \\
&+ \int_0^T \int_{\partial\Omega} \mu^T(x,t+\tau) \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \partial_x d_\theta u(x,t) dxdt \\
&- \int_0^T \int_{\partial\Omega} \partial_x \left(\mu^T(x,t+\tau) \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \right) d_\theta u(x,t) dxdt \\
&+ \int_0^T \int_\Omega \partial_{xx} \left(\mu^T(x,t+\tau) \partial_{\partial_{xx} u(x,t)} \mathcal{G}_{NN}(\bullet, t; \theta) \right) d_\theta u(x,t) dxdt \\
&+ \int_0^T \int_\Omega \mu^T(x,t) \partial_\theta \mathcal{G}_{NN}(\bullet, t-\tau; \theta) dxdt \\
&+ \int_\Omega \alpha^T(x) d_\theta y(x,0) \\
&- \int_\Omega \alpha^T(x) \int_{-\tau}^0 \partial_\theta \mathcal{G}_{NN}(h(x,t), \partial_x h(x,t), \partial_{xx} h(x,t), x, t; \theta) dt dx \,.
\end{aligned}
\tag{F.12}
$$

Collecting all the terms with $\int_\Omega$, $d_\theta u(x,t)$, and $d_\theta y(x,t)$, we get the following adjoint PDEs,

$$
\begin{aligned}
0 = &\frac{1}{M}\frac{1}{|\Omega|}\sum_{k=1}^{M}\partial_{u(x,t)}\hat{l}(u(x,t))\delta(t-T_k) \\
&- \partial_t\lambda^T(x,t) - \lambda^T(x,t)\partial_{u(x,t)}\mathcal{F}_{NN}(\bullet,t) + \sum_{i=1}^{d}(-1)^{i+1}\partial_{x^i}\left(\lambda^T(x,t)\partial_{\partial_{x^i}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right) \\
&- \mu^T(x,t)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta) + \sum_{i=1}^{d}(-1)^{i+1}\partial_{x^i}\left(\mu^T(x,t)\partial_{\partial_{x^i}u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right) \\
&+ \mu^T(x,t+\tau)\partial_{u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta) - \sum_{i=1}^{d}(-1)^{i+1}\partial_{x^i}\left(\mu^T(x,t+\tau)\partial_{\partial_{x^i}u(x,t)}\mathcal{G}_{NN}(\bullet,t;\theta)\right)\ , \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x\in\Omega\ ,\ t\in[0,T)\,, \\
0 = &-\lambda^T(x,t) - \partial_t\mu^T(x,t)\ , \quad x\in\Omega\ ,\ t\in[0,T)\,,
\end{aligned}
\tag{F.13}
$$

with initial conditions, $\lambda(x,t) = \mu(x,t) = 0,\ t \geq T$. The boundary conditions are derived based on that of the forward PDE such that they satisfy,

$$
\begin{aligned}
0 = &\sum_{i=0}^{d}\sum_{j=0}^{d-i-1}(-1)^{j+1}\partial_{x^j}\left(\lambda^T(x,t)\partial_{\partial_{x^{j+i+1}}u(x,t)}\mathcal{F}_{NN}(\bullet,t)\right)d_\theta\partial_{x^i}u(x,t) \\
&+ \sum_{i=0}^{d}\sum_{j=0}^{d-i-1}(-1)^{j+1}\partial_{x^j}\left(\mu^T(x,t)\partial_{\partial_{x^{j+i+1}}u(x,t)}\mathcal{G}_{NN}(\bullet,t)\right)d_\theta\partial_{x^i}u(x,t) \\
&- \sum_{i=0}^{d}\sum_{j=0}^{d-i-1}(-1)^{j+1}\partial_{x^j}\left(\mu^T(x,t+\tau)\partial_{\partial_{x^{j+i+1}}u(x,t)}\mathcal{G}_{NN}(\bullet,t)\right)d_\theta\partial_{x^i}u(x,t)\ , \\
&\qquad\qquad\qquad\qquad\qquad\qquad x\in\partial\Omega,\ t\in[t,T)\,.
\end{aligned}
\tag{F.14}
$$

Note that adjoint PDE needs to be solved backward in time, and one would require access to $u(x,t), \forall x \in \Omega,\ 0 \leq t \leq T$. After solving for the Lagrangian variables,

$\lambda(x,t)$ and $\mu(x,t)$, we can compute the required gradients as follows:

$$d_\theta \mathcal{L} = -\int_0^T \int_\Omega \mu^T(x,t)\partial_\theta \mathcal{G}_{NN}(\bullet, t; \theta)dxdt + \int_0^T \int_\Omega \mu^T(x,t)\partial_\theta \mathcal{G}_{NN}(\bullet, t-\tau; \theta)dxdt$$
$$- \int_\Omega \mu^T(x,0) \int_{-\tau}^0 \partial_\theta \mathcal{G}_{NN}(h(x,t), \partial_x h(x,t), \partial_{xx} h(x,t), x, t; \theta)dtdx\,. \tag{F.15}$$

If we restart the above derivation by taking derivative of the Lagrangian (equation F.3) w.r.t. ϕ, we will arrive at the same adjoint PDEs (equations F.13 & F.14), and the required gradient will be given by,

$$d_\phi \mathcal{L} = -\int_0^T \int_\Omega \lambda^T(x,t)\partial_\phi \mathcal{F}_{NN}(\bullet, t; \phi)dxdt\,. \tag{F.16}$$

Finally, using any stochastic gradient descent algorithm, we can find the optimal values of the weights ϕ and θ.

F.2 Experimental Setup

F.2.1 Architectures

The architectures used to generate the results corresponding to different experiments are provided in table F.1. The implementation details of the various biological and carbonate constraints imposed on the neural closure terms in experiments 2a & 2b are also provided.

F.2.2 Hyperparameters

The values of the tuned training hyperparameters corresponding to different experiments are listed next. In all the experiments, the number of iterations per epoch are calculated by dividing the number of time-steps in the training period by the batch-size multiplied by the length of short time-sequences, adding 1, and rounding up to the next integer.

Experiments-1a:

For training, we randomly select short time-sequences spanning 3 time-steps (batch-time) and extract data at every time-step to form batches of size 16; 4 iterations per epoch are used; an exponentially decaying learning rate (LR) schedule is used with initial LR of 0.075, decay rate of 0.97, and 4 decay steps; the RMSprop optimizer is employed; training is for a total of 150 epochs. $\mathcal{L}_1$ and $\mathcal{L}_2$ regularization with factors of 1.5×10^{-3} and 1×10^{-5}, respectively, is used; the weights are pruned if the value drops below 5×10^{-3}.

Experiments-1b:

Only Markovian closure case: We randomly select short time-sequences spanning 30 time-steps (batch-time) and extract data at every other time-step to form batches of size 2. In total 24 iterations per epoch are used, with every 8 of them belonging to one of the (N_x, Re) pairs. An exponentially decaying learning rate (LR) schedule with initial LR of 0.025, decay rate of 0.95, and 24 decay steps is used; the RMSprop optimizer is used; we train for a total of 30 epochs. We also use both $\mathcal{L}_1$ and $\mathcal{L}_2$ regularization for the weights of the neural network with factors of 5×10^{-4} and 5×10^{-4}, respectively, along with pruning of the weights if their value drops below 5×10^{-3}.

Both Markovian and non-Markovian closures case: A batch-time of 30 time-steps is used with data extracted at every other time-step to form batches of size 2; 32 iterations per epoch are used, with every 8 of them belonging to one of the (N_x, Re) pairs; an exponentially decaying learning rate (LR) schedule is employed with an initial LR of 0.01, decay rate of 0.95, and 32 decay steps; the RMSprop optimizer is used; we train for a total of 30 epochs. We also use both $\mathcal{L}_1$ and $\mathcal{L}_2$ regularization for the weights of the neural network with factors of 1.5×10^{-3} and 1×10^{-5}, respectively, along with pruning of the weights if their value drops below 5×10^{-3}.

Experiments-2a:

Parameter values used in the ocean acidification model are (adopted from [77, 4]): $g_{max} = 0.7\ day^{-1}$; $k_W = 0.08\ m^{-1}$; $K_N = 0.5\ mmol\ N\ m^{-3}$; $K_P = 0.25\ mmol\ N\ m^{-3}$; $m_P = 0.08\ day^{-1}(mmol\ N\ m^{-3})^{-1}$; $m_Z = 0.030\ day^{-1}(mmol\ N\ m^{-3})^{-1}$; $\mu_{max} = 2.808\ day^{-1}$; $\alpha = 0.14\ (W\ m^{-2}\ day)^{-1}$; $\beta = 0.0028\ (W\ m^{-2}\ day)^{-1}$; $\epsilon = 0.015\ day^{-1}$; $\lambda = 0.3$; $\gamma = 0.4$; a sinusoidal variation in $I_o(t)$; linear vertical variation in total biomass $T_{bio}(z)$ from 10 $mmol\ N\ m^{-3}$ at the surface to 20 $mmol\ N\ m^{-3}$ at $z = 100\ m$; $K_{z_b} = 0.0864\ (m^2/day)$; $K_{z_0} = 8.64\ (m^2/day)$; $\gamma_t = 0.1\ m^{-1}$; $D_z = -100\ m$; and $\rho_w = 1000\ kg/m^3$. For training, we randomly select short time-sequences spanning 3 time-steps (batch-time) and extract data at every other time-step to form batches of size 4; we use 26 iterations per epoch; an exponentially decaying learning rate (LR) schedule is used with initial LR of 0.075, decay rate of 0.97, and 26 decay steps; the RMSprop optimizer is adopted; training is terminated at 200 epochs. We also use both $\mathcal{L}_1$ and $\mathcal{L}_2$ regularization for the weights of the neural network with factors of 1.5×10^{-3} and 1×10^{-3}, respectively, along with pruning of the weights if their value drops below 5×10^{-3}.

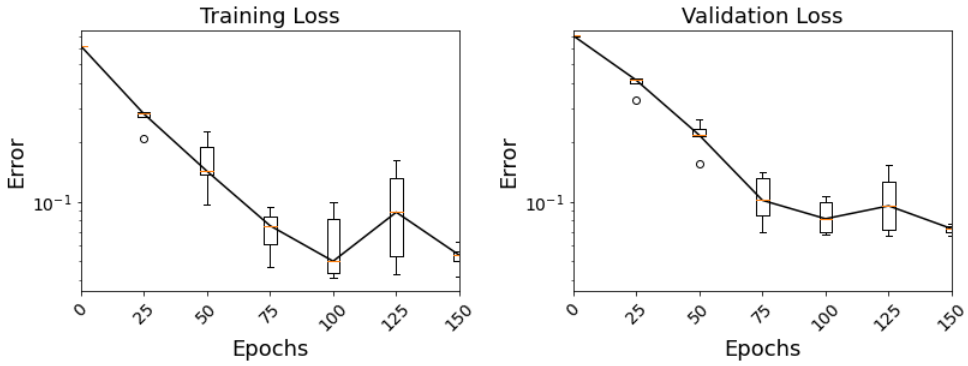
Experiments-2b:

We use a batch-time of 3 time-steps with data extracted at every other time-step to form batches of size 8; we use 26 iterations per epoch; an exponentially decaying learning rate (LR) schedule is applied with initial LR of 0.075, decay rate of 0.97, and 26 decay steps; the RMSprop optimizer is employed; training is terminated at 200 epochs. We also use both $\mathcal{L}_1$ and $\mathcal{L}_2$ regularization for the weights of the Markovian closure with factors of 1.5×10^{-3} and 1×10^{-3}, respectively, along with pruning of the weights if their value drops below 5×10^{-3}. For the neural network in the non-Markovian closure term, only $\mathcal{L}_2$ regularization with a factor of 1×10^{-5} is used.

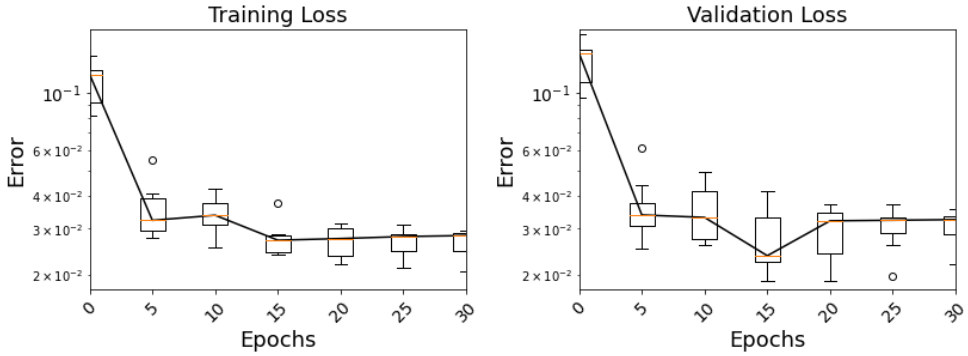
Finally, for all the experiments and their multiple repeats with the exact same tuned hyperparameters, we provide variation of training and validation error as training progresses (figure F-1).

Experiments	Markovian Term			Non-Markovian Term			
	Architecture	**Act.**	**Trainable Weights**	**Architecture**	**Act.**	**Delays**	**Trainable Weights**
1a	$\mathcal{F}_{NN}$		4				
	Input layer with 4 neurons	none					
	Dense output layer with 1 neurons	linear					
1b	$\mathcal{F}_{NN}$		4	$\mathcal{G}_{NN}$		0.075	198
	Input layer with 4 neurons	none		Input layer with 5 neurons	none		
	Dense output layer with 1 neurons	linear		Dense hidden layer with 10 neurons	swish		
				Dense hidden layer with 7 neurons	swish		
				2 Dense hidden layers with 5 neurons	swish		
				Dense hidden layer with 3 neurons	swish		
				Dense output layer with 1 neuron	linear		
				Multiply output with $\lvert u\rvert$			
2a	$\mathcal{F}_{NN}$		18 (effective) $w = \begin{bmatrix} -(w_2+w_3+w_4) \\ w_2 \\ w_3 \\ w_4 \\ -C_P w_2 - C_Z w_3 - C_D w_4 \\ (w_2+w_3+w_4)/\rho_w \end{bmatrix}$				
	Input layer with 6 neurons	none					
	Dense output layer with 6 neurons	linear					
2b	$\mathcal{F}_{NN}$		4 (effective) $w = \begin{bmatrix} -w_3 \\ 0 \\ w_3 \\ -C_Z w_3 \\ w_3/\rho_w \end{bmatrix}$	$\mathcal{G}_{NN}$		2.5	65
	Input layer with 4 neurons	none		Input layer with 4 neurons	none		
	Dense output layer with 5 neurons	linear		2 Dense hidden layer with 5 neurons	swish		
				Dense output layer with 4 neurons	linear		
				$\mathcal{G}_{NN} = \begin{bmatrix} -(\mathcal{G}^1_{NN} + \mathcal{G}^2_{NN}) \\ \mathcal{G}^1_{NN} \\ \mathcal{G}^2_{NN} \\ \mathcal{G}^3_{NN} \\ \mathcal{G}^4_{NN}/\rho_w \end{bmatrix}$			

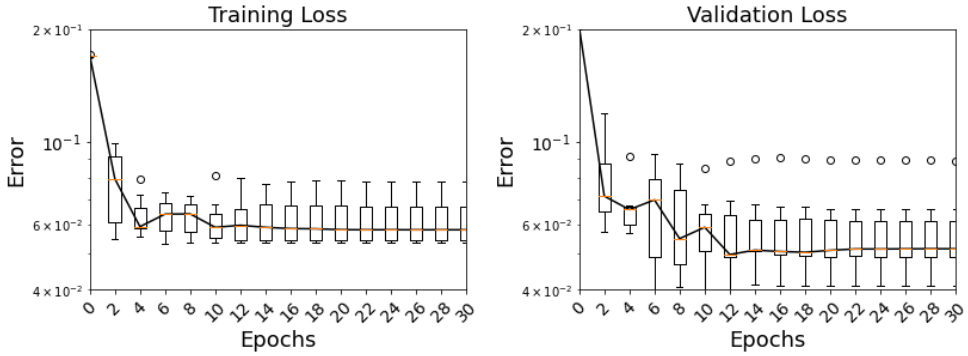
Table F.1: Architectures for different generalized neural closure models used in the four sets of experiments. We explicitly provide the constraints on the weights and output layer of neural networks used in different experiments. $\{w_i\}_{i=1}^4$ are row vectors of the weight matrix. "Effective" number of trainable weights do not count the ones which are not free or are overwritten due to the imposed constraints. C_P, C_Z, and C_D are the carbon-nitrogen ratios for phytoplanktons, zooplanktons, and detritus, respectively. ρ_w is seawater density.



(a) Experiments-1a

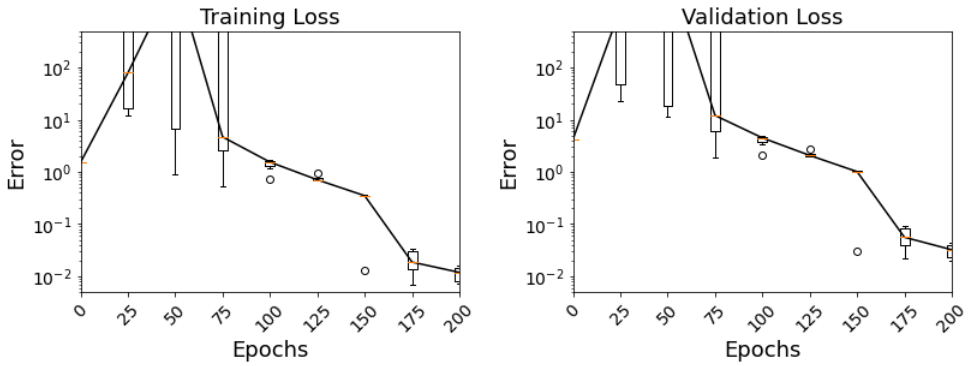


(b) Experiments-1b (gnCM with only Markovian closure term)

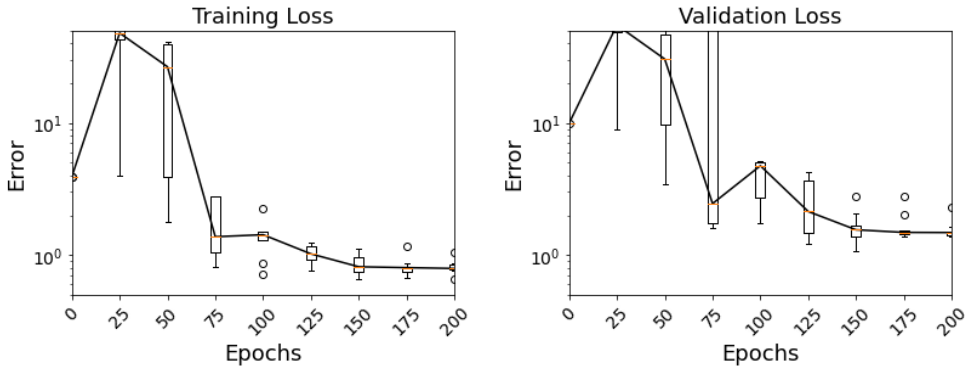


(c) Experiments-1b (gnCM with both Markovian and non-Markovian closure terms)

Figure F-1: Variation of training (left column) and validation (right column) loss with epochs, for each of the experiments-1a, 1b, 2a, and 2b. We use boxplots to provide statistical summaries for multiple training repeats done for each set of experiments. The box and its whiskers provide a five number summary: minimum, first quartile (Q1), median (orange solid line), third quartile (Q3), and maximum, along with outliers (black circles) if any. These results accompany the architectures detailed in table F.1. *(cont.)*



(d) Experiments-2a



(e) Experiments-2b

Figure F-1: Variation of training (left column) and validation (right column) loss with epochs, for each of the experiments-1a, 1b, 2a, and 2b. We use boxplots to provide statistical summaries for multiple training repeats done for each set of experiments. The box and its whiskers provide a five number summary: minimum, first quartile (Q1), median (orange solid line), third quartile (Q3), and maximum, along with outliers (black circles) if any. These results accompany the architectures detailed in table F.1.

Bibliography

[1] P. F. J. Lermusiaux, P. J. Haley, Jr., S. Jana, A. Gupta, C. S. Kulkarni, C. Mirabito, W. H. Ali, D. N. Subramani, A. Dutt, J. Lin, A. Shcherbina, C. Lee, and A. Gangopadhyay. Optimal planning and sampling predictions for autonomous and Lagrangian platforms and sensors in the northern Arabian Sea. *Oceanography*, 30(2):172–185, June 2017. Special issue on Autonomous and Lagrangian Platforms and Sensors (ALPS).

[2] Qian Wang, Nicolò Ripamonti, and Jan S Hesthaven. Recurrent neural network closure of parametric POD-galerkin reduced-order models based on the Mori-Zwanzig formalism. *Journal of Computational Physics*, page 109402, 2020.

[3] Priscilla A. Newberger, John S. Allen, and Y. H. Spitz. Analysis and comparison of three ecosystem models. *Journal of Geophysical Research: Oceans (1978–2012)*, 108(C3), 2003.

[4] Mette Eknes and Geir Evensen. An ensemble kalman filter with a 1-d marine ecosystem model. *Journal of Marine Systems*, 36(1-2):75–100, 2002.

[5] Steven C Chapra and Raymond P Canale. *Numerical methods for engineers*, volume 1221. Mcgraw-hill New York, 2011.

[6] Ching Jen Chen. *Fundamentals of turbulence modelling*. CRC Press, 1997.

[7] Paul A Durbin. Some recent developments in turbulence closure modeling. *Annual Review of Fluid Mechanics*, 50:77–103, 2018.

[8] Peter Lu and Pierre F. J. Lermusiaux. Bayesian learning of stochastic dynamical models. *Physica D: Nonlinear Phenomena*, 427:133003, December 2021.

[9] Jing Lin. *Bayesian Learning for High-Dimensional Nonlinear Systems: Methodologies, Numerics and Applications to Fluid Flows*. PhD thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, September 2020.

[10] P. G. Y. Lu and Pierre F. J. Lermusiaux. Pde-based bayesian inference of high-dimensional dynamical models. MSEAS Report 19, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA, 2014.

[11] Themistoklis P. Sapsis and Pierre F. J. Lermusiaux. Dynamically orthogonal field equations for continuous stochastic dynamical systems. *Physica D: Nonlinear Phenomena*, 238(23–24):2347–2360, December 2009.

[12] Themistoklis P. Sapsis and Pierre F. J. Lermusiaux. Dynamical criteria for the evolution of the stochastic dimensionality in flows with uncertainty. *Physica D: Nonlinear Phenomena*, 241(1):60–76, 2012.

[13] Florian Feppon and Pierre F. J. Lermusiaux. A geometric approach to dynamical model-order reduction. *SIAM Journal on Matrix Analysis and Applications*, 39(1):510–538, 2018.

[14] Florian Feppon and Pierre F. J. Lermusiaux. Dynamically orthogonal numerical schemes for efficient stochastic advection and Lagrangian transport. *SIAM Review*, 60(3):595–625, 2018.

[15] Florian Feppon and Pierre F. J. Lermusiaux. The extrinsic geometry of dynamical systems tracking nonlinear matrix projections. *SIAM Journal on Matrix Analysis and Applications*, 40(2):814–844, 2019.

[16] T. Sondergaard and P. F. J. Lermusiaux. Data assimilation with Gaussian Mixture Models using the Dynamically Orthogonal field equations. Part I: Theory and scheme. *Monthly Weather Review*, 141(6):1737–1760, 2013.

[17] T. Sondergaard and P. F. J. Lermusiaux. Data assimilation with Gaussian Mixture Models using the Dynamically Orthogonal field equations. Part II: Applications. *Monthly Weather Review*, 141(6):1761–1785, 2013.

[18] Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[19] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[20] Ken-ichi Funahashi and Yuichi Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6):801–806, 1993.

[21] Coryn AL Bailer-Jones, David JC MacKay, and Philip J Withers. A recurrent neural network for modelling dynamical systems. *network: computation in neural systems*, 9(4):531, 1998.

[22] Yu Wang. A new concept using lstm neural networks for dynamic system identification. In *2017 American control conference (ACC)*, pages 5324–5329. IEEE, 2017.

[23] Shubhendu Kumar Singh, Ruoyu Yang, Amir Behjat, Rahul Rai, Souma Chowdhury, and Ion Matei. Pi-lstm: Physics-infused long short-term memory network. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 34–41. IEEE, 2019.

[24] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS*, 113(15):3932–3937, 2016.

[25] Samuel Rudy, Alessandro Alla, Steven L Brunton, and J Nathan Kutz. Data-driven identification of parametric partial differential equations. *SIAM Journal on Applied Dynamical Systems*, 18(2):643–660, 2019.

[26] Daniel A Messenger and David M Bortz. Weak sindy for partial differential equations. *Journal of Computational Physics*, page 110525, 2021.

[27] Chinmay S. Kulkarni, Abhinav Gupta, and Pierre F. J. Lermusiaux. Sparse regression and adaptive feature generation for the discovery of dynamical systems. In F. Darema, E. Blasch, S. Ravela, and A. Aved, editors, *Dynamic Data Driven Application Systems. DDDAS 2020.*, volume 12312 of *Lecture Notes in Computer Science*, pages 208–216. Springer, Cham, November 2020.

[28] Robert K Niven, Ali Mohammad-Djafari, Laurent Cordier, Markus Abel, and Markus Quade. Bayesian identification of dynamical systems. In *Multidisciplinary Digital Publishing Institute Proceedings*, volume 33, page 33, 2020.

[29] M Maslyaev, A Hvatov, and A Kalyuzhnaya. Data-driven pde discovery with evolutionary approach.(2019). *arXiv preprint arXiv:1903.08011*.

[30] Maxime Bassenne and Adrián Lozano-Durán. Computational model discovery with reinforcement learning. *arXiv preprint arXiv:2001.00008*, 2019.

[31] Guido Novati, Hugues Lascombes de Laroussilhe, and Petros Koumoutsakos. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1):87–96, 2021.

[32] Yufei Wang, Ziju Shen, Zichao Long, and Bin Dong. Learning to discretize: solving 1d scalar conservation laws via deep reinforcement learning. *arXiv preprint arXiv:1905.11079*, 2019.

[33] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[34] Christopher Rackauckas et al. Universal differential equations for scientific machine learning. *arXiv preprint:2001.04385*, 2020.

[35] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[36] Anima Anandkumar, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Nikola Kovachki, Zongyi Li, Burigede Liu, and Andrew Stuart. Neural operator: Graph kernel network for partial differential equations. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

[37] Christopher Rackauckas et al. SciML Scientific Machine Learning Software. `https://sciml.ai/`.

[38] Carol Lalli and Timothy R. Parsons. *Biological Oceanography: An Introduction*. Elsevier Butterworth-Heinemann, 1997.

[39] Wolfgang Fennel and Thomas Neumann. *Introduction to the Modelling of Marine Ecosystems:(with MATLAB programs on accompanying CD-ROM)*, volume 72 of *Oceanography*. Elsevier, 2014.

[40] Peter J. S. Franks. NPZ models of plankton dynamics: their construction, coupling to physics, and application. *Journal of Oceanography*, 58(2):379–387, 2002.

[41] Ben A. Ward, Marjorie A. M. Friedrichs, Thomas R. Anderson, and Andreas Oschlies. Parameter optimisation techniques and the problem of underdetermination in marine biogeochemical models. *Journal of Marine Systems*, 81(1):34–43, 2010.

[42] P. J. S. Franks, J. S. Wroblewski, and G. R. Flierl. Behavior of a simple plankton model with food-level acclimation by herbivores. *Marine Biology*, 91(1):121–129, 1986.

[43] Glenn Flierl and Dennis J. McGillicuddy. Mesoscale and submesoscale physical-biological interactions. *The sea*, 12:113–185, 2002.

[44] Cabell S. Davis and John H. Steele. Biological/physical modeling of upper ocean processes. Technical report, Woods Hole Oceanographic Institution, 1994.

[45] M. J. R. Fasham, H. W. Ducklow, and S. M. McKelvie. A nitrogen-based model of plankton dynamics in the oceanic mixed layer. *Journal of Marine Research*, 48(3):591–639, 1990.

[46] J. W. Baretta, W. Ebenhöh, and P. Ruardij. The European Regional Seas Ecosystem Model, a complex marine ecosystem model. *Netherlands Journal of Sea Research*, 33(3):233–246, 1995.

[47] J. W. Baretta. Preface to the European Regional Seas Ecosystem Model II. *Journal of Sea Research*, 38(3):169–171, 1997.

[48] J. C. Blackford, J. I. Allen, and F. J. Gilbert. Ecosystem dynamics at six contrasting sites: a generic modelling study. *Journal of Marine Systems*, 52(1):191–215, 2004.

[49] Marjorie A. M. Friedrichs, Jeffrey A. Dusenberry, Laurence A. Anderson, Robert A. Armstrong, Fei Chai, James R. Christian, Scott C. Doney, John Dunne, Masahiko Fujii, Raleigh Hood, et al. Assessment of skill and portability in regional marine biogeochemical models: Role of multiple planktonic groups. *Journal of Geophysical Research: Oceans*, 112(C8), 2007.

[50] Svetlana N. Losa, Gennady A. Kivman, and Vladimir A. Ryabchenko. Weak constraint parameter estimation for a simple ocean ecosystem model: what can we learn about the model and data? *Journal of Marine Systems*, 45(1):1–20, 2004.

[51] Jann Paul Mattern, Mike Dowd, and Katja Fennel. Sequential data assimilation applied to a physical–biological model for the bermuda atlantic time series station. *Journal of Marine Systems*, 79(1):144–156, 2010.

[52] J. I. Allen, M. Eknes, and G. Evensen. An Ensemble Kalman Filter with a complex marine ecosystem model: hindcasting phytoplankton in the Cretan Sea. In *Annales Geophysicae*, volume 21, pages 399–411, 2003.

[53] L. J. Natvik and Geir Evensen. Assimilation of ocean colour data into a biochemical model of the North Atlantic: Part 1. Data assimilation experiments. *Journal of Marine Systems*, 40:127–153, 2003.

[54] Maéva Doron, Pierre Brasseur, and Jean-Michel Brankart. Stochastic estimation of biogeochemical parameters of a 3D ocean coupled physical–biogeochemical model: Twin experiments. *Journal of Marine Systems*, 87(3):194–207, 2011.

[55] Emlyn Jones, John Parslow, and Lawrence Murray. A bayesian approach to state and parameter estimation in a phytoplankton-zooplankton model. *Australian Meteorological and Oceanographic Journal*, 59(SP):7–16, 2010.

[56] Mr. Bayes and Mr. Price. An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S. *Philosophical Transactions (1683-1775)*, 53:370–418, 1763.

[57] Jean-Luc Guermond. Lecture notes for MATH 610 - numerical methods in partial differential equations, May 2016.

[58] Arthur Gelb. *Applied optimal estimation*. MIT Press, 1974.

[59] Peter E Hart, David G Stork, and Richard O Duda. *Pattern classification*. Wiley Hoboken, 2000.

[60] Rucheng C. Tian, Pierre F. J. Lermusiaux, James J. McCarthy, and Allan R. Robinson. A generalized prognostic model of marine biogeochemical-ecosystem dynamics: Structure, parameterization and adaptive modeling. Harvard Reports in Physical/Interdisciplinary Ocean Science 67, Department of Earth and Planetary Sciences, Harvard University, Cambridge, MA, May 2004.

[61] P. F. J Lermusiaux. Adaptive modeling, adaptive data assimilation and adaptive sampling. *Physica D: Nonlinear Phenomena*, 230(1):172–196, 2007.

[62] Ş. T. Beşiktepe, P. F. J. Lermusiaux, and A. R. Robinson. Coupled physical and biogeochemical data-driven simulations of Massachusetts Bay in late summer: Real-time and post-cruise data assimilation. *Journal of Marine Systems*, 40–41:171–212, 2003.

[63] Patrick J. Haley, Jr. and Pierre F. J. Lermusiaux. Multiscale two-way embedding schemes for free-surface primitive equations in the "Multidisciplinary Simulation, Estimation and Assimilation System". *Ocean Dynamics*, 60(6):1497–1537, December 2010.

[64] P. J. Haley, Jr., A. Agarwal, and P. F. J. Lermusiaux. Optimizing velocities and transports for complex coastal regions and archipelagos. *Ocean Modeling*, 89:1–28, 2015.

[65] James C McWilliams. The nature and consequences of oceanic eddies. *Ocean modeling in an eddying regime*, 177:5–15, 2008.

[66] Matthew W Hecht and Hiroyasu Hasumi. *Ocean modeling in an eddying regime*, volume 177. John Wiley & Sons, 2013.

[67] Joel H Ferziger, Milovan Perić, and Robert L Street. *Computational methods for fluid dynamics*, volume 3. Springer, 2002.

[68] D. N. Subramani and P. F. J. Lermusiaux. Energy-optimal path planning by stochastic dynamically orthogonal level-set optimization. *Ocean Modeling*, 100:57–77, 2016.

[69] D. N. Subramani, Q. J. Wei, and P. F. J. Lermusiaux. Stochastic time-optimal path-planning in uncertain, strong, and dynamic flows. *Computer Methods in Applied Mechanics and Engineering*, 333:218–237, 2018.

[70] Michal Branicki and Andrew J Majda. Fundamental limitations of polynomial chaos for uncertainty quantification in systems with intermittent instabilities. *Communications in mathematical sciences*, 11(1):55–103, 2013.

[71] Michael Jesus Humara. Stochastic acoustic ray tracing with dynamically orthogonal equations. Master's thesis, Massachusetts Institute of Technology, Joint Program in Applied Ocean Science and Engineering, Cambridge, Massachusetts, May 2020.

[72] Michael J. Humara, Wael Hajj Ali, Aaron Charous, Manmeet Bhabra, and Pierre F. J. Lermusiaux. Stochastic acoustic ray tracing with dynamically orthogonal differential equations. In *OCEANS 2022 IEEE/MTS*. IEEE, October 2022. Sub-judice.

[73] Aaron Charous, Michael J. Humara, Wael H. Ali, Manmeet S. Bhabra, Abhinav Gupta, and Pierre F. Lermusiaux. Dynamically orthogonal ray equations with adaptive reclustering. *The Journal of the Acoustical Society of America*, 150(4):A209–A209, 2021.

[74] DJ McGillicuddy, DR Lynch, AM Moore, WC Gentleman, CS Davis, and CJ Meise. An adjoint data assimilation approach to diagnosis of physical and biological controls on pseudocalanus spp. in the gulf of maine–georges bank region. *Fisheries Oceanography*, 7(3-4):205–218, 1998.

[75] P. F. J. Lermusiaux. Evolving the subspace of the three-dimensional multiscale ocean variability: Massachusetts Bay. *Journal of Marine Systems*, 29(1):385–422, 2001.

[76] Jesús Pineda, Victoria Starczak, José CB da Silva, Karl Helfrich, Michael Thompson, and David Wiley. Whales and waves: Humpback whale foraging response and the shoaling of internal waves at s tellwagen b ank. *Journal of Geophysical Research: Oceans*, 120(4):2555–2570, 2015.

[77] Rucheng Tian, Changsheng Chen, Jianhua Qi, Rubao Ji, Robert C Beardsley, and Cabell Davis. Model study of nutrient and phytoplankton dynamics in the gulf of maine: patterns and drivers for seasonal and interannual variability. *ICES Journal of Marine Science*, 72(2):388–402, 2015.

[78] AJ Pershing, MA Alexander, D Brickman, J Scott, Enrique Curchitser, D Brady, T Diamond, L McClenachan, KE Mills, O Nichols, et al. Temperature and circulation conditions in the gulf of maine in 2050 and their expected impacts. Scientific scenario paper, Gulf of Maine 2050 International Symposium, 2019.

[79] Tammy L Silva. State of the science report: An addendum to the stellwagen bank national marine sanctuary 2020 condition report. 2021.

[80] M. P. Ueckermann and P. F. J. Lermusiaux. 2.29 Finite Volume MATLAB Framework Documentation. MSEAS Report 14, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 2012.

[81] Bram Van Leer. Towards the ultimate conservative difference scheme. iv. a new approach to numerical convection. *Journal of computational physics*, 23(3):276–299, 1977.

[82] M. P. Ueckermann, P. F. J. Lermusiaux, and T. P. Sapsis. Numerical schemes for dynamically orthogonal equations of stochastic fluid and ocean flows. *Journal of Computational Physics*, 233:272–294, January 2013.

[83] P. F. J. Lermusiaux. On the mapping of multivariate geophysical fields: Sensitivities to size, scales, and dynamics. *Journal of Atmospheric and Oceanic Technology*, 19(10):1602–1637, 2002.

[84] P. F. J. Lermusiaux, P. J. Haley, W. G. Leslie, A. Agarwal, O. Logutov, and L. J. Burton. Multiscale physical and biological dynamics in the Philippine Archipelago: Predictions and processes. *Oceanography*, 24(1):70–89, 2011. Special Issue on the Philippine Straits Dynamics Experiment.

[85] Lennart Bengtsson, Michael Ghil, and Erland Källén. *Dynamic meteorology: data assimilation methods*. Springer, 1981.

[86] Kayo Ide and Michael Ghil. Extended kalman filtering for vortex systems. part 1: Methodology and point vortices. *Dynamics of Atmospheres and Oceans*, 27(1-4):301–332, 1998.

[87] Kayo Ide and Michael Ghil. Extended kalman filtering for vortex systems. part ii: Rankine vortices and observing-system design. *Dynamics of Atmospheres and Oceans*, 27(1-4):333–350, 1998.

[88] P. F. J. Lermusiaux. Data assimilation via Error Subspace Statistical Estimation, part II: Mid-Atlantic Bight shelfbreak front simulations, and ESSE validation. *Monthly Weather Review*, 127(7):1408–1432, July 1999.

[89] T. Lolla, P. J. Haley, Jr., and P. F. J. Lermusiaux. Time-optimal path planning in dynamic flows using level set equations: Realistic applications. *Ocean Dynamics*, 64(10):1399–1417, 2014.

[90] Abhinav Gupta, Patrick J. Haley, Deepak N. Subramani, and Pierre F. J. Lermusiaux. Fish modeling and Bayesian learning for the Lakshadweep Islands. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–10, Seattle, October 2019. IEEE.

[91] G. Wornell. Inference and information. lecture notes for mit course 6.437 in spring 2013, May 2016.

[92] Abhinav Gupta. Bayesian inference of obstacle systems and coupled biogeochemical-physical models. Master's thesis, Indian Institute of Technology Kanpur, Kanpur, India, 2016.

[93] Simo Särkkä. *Bayesian filtering and smoothing*, volume 3. Cambridge University Press, 2013.

[94] T. Lolla and P. F. J. Lermusiaux. A Gaussian mixture model smoother for continuous nonlinear stochastic dynamical systems: Theory and scheme. *Monthly Weather Review*, 145:2743–2761, July 2017.

[95] T. Lolla and P. F. J. Lermusiaux. A Gaussian mixture model smoother for continuous nonlinear stochastic dynamical systems: Applications. *Monthly Weather Review*, 145:2763–2790, July 2017.

[96] P. F. J. Lermusiaux, D. N. Subramani, J. Lin, C. S. Kulkarni, A. Gupta, A. Dutt, T. Lolla, P. J. Haley, Jr., W. H. Ali, C. Mirabito, and S. Jana. A future for intelligent autonomous ocean observing systems. *Journal of Marine Research*, 75(6):765–813, November 2017. The Sea. Volume 17, The Science of Ocean Prediction, Part 2.

[97] Department of Animal Husbandy, Dairing & Fisheries. Handbook on fisheries statistics, 2014.

[98] Kjartan G Magnusson. An overview of the multispecies vpa theory and applications. *Reviews in Fish Biology and Fisheries*, 5(2):195–212, 1995.

[99] Sarah Hinckley. *Biophysical mechanisms underlying the recruitment process in walleye pollock (Theragra chalcogramma)*. PhD thesis, 1999.

[100] Bernard A Megrey, Kenneth A Rose, Robert A Klumb, Douglas E Hay, Francisco E Werner, David L Eslinger, and S Lan Smith. A bioenergetics-based population dynamics model of pacific herring (clupea harengus pallasi) coupled to a lower trophic level nutrient–phytoplankton–zooplankton model: description, calibration, and sensitivity analysis. *Ecological Modelling*, 202(1):144–164, 2007.

[101] Michio J Kishi, Shin-ichi Ito, Bernard A Megrey, Kenneth A Rose, and Francisco E Werner. A review of the nemuro and nemuro. fish models and their application to marine ecosystem investigations. *Journal of oceanography*, 67(1):3–16, 2011.

[102] Alethea Barbaro. Modelling and simulations of the migration of pelagic fish. *ICES Journal of Marine Science*, 66(5):826–838, 2009.

[103] Olivier Maury, Blaise Faugeras, Yunne-Jai Shin, Jean-Christophe Poggiale, Tamara Ben Ari, and Francis Marsac. Modeling environmental effects on the size-structured energy flow through marine ecosystems. part 1: the model. *Progress in Oceanography*, 74(4):479–499, 2007.

[104] Villy Christensen, Marta Coll, Jeroen Steenbeek, Joe Buszowski, Dave Chagaris, and Carl J Walters. Representing variable habitat quality in a spatial food web model. *Ecosystems*, 17(8):1397–1412, 2014.

[105] Rajeev Kumar, Szymon Surma, Tony J Pitcher, Divya Varkey, Mimi E Lam, Cameron H Ainsworth, and Evgeny Pakhomov. An ecosystem model of the ocean around haida gwaii, northern british columbia: Ecopath, ecosim and ecospace. 2016.

[106] Patrick Lehodey, Fei Chai, and John Hampton. Modelling climate-related variability of tuna populations from a coupled ocean–biogeochemical-populations dynamics model. *Fisheries Oceanography*, 12(4-5):483–494, 2003.

[107] Patrick Lehodey, JEAN-MICHEL ANDRE, Michel Bertignac, John Hampton, Anne Stoens, Christophe Menkès, Laurent Mémery, and Nicolas Grima. Predicting skipjack tuna forage distributions in the equatorial pacific using a coupled dynamical bio-geochemical model. *Fisheries Oceanography*, 7(3-4):317–325, 1998.

[108] A. R. Robinson, B. J. Rothschild, W. G. Leslie, J. J. Bisagni, M. F. Borges, W. S. Brown, D. Cai, P. Fortier, A. Gangopadhyay, P. J. Haley, Jr., H. S. Kim, L. Lanerolle, P. F. J. Lermusiaux, C. J. Lozano, M. G. Miller, G. Strout, and M. A. Sundermeyer. The development and demonstration of an advanced fisheries management information system. In *Proc. of the 17th Conference on Interactive Information and Processing Systems for Meteorology, Oceanography and Hydrology*, pages 186–190, Albuquerque, New Mexico, 2002. American Meteorological Society.

[109] Jintao Wang, Wei Yu, Xinjun Chen, Lin Lei, and Yong Chen. Detection of potential fishing zones for neon flying squid based on remote-sensing data in the northwest pacific ocean using an artificial neural network. *International journal of remote sensing*, 36(13):3317–3330, 2015.

[110] Jintao Wang, Xinjun Chen, Kevin W Staples, and Yong Chen. The skipjack tuna fishery in the west-central pacific ocean: applying neural networks to detect habitat preferences. *Fisheries Science*, 84(2):309–321, 2018.

[111] Kuo-Wei Lan, Teruhisa Shimada, Ming-An Lee, Nan-Jay Su, and Yi Chang. Using remote-sensing environmental and fishery data to map potential yellowfin tuna habitats in the tropical pacific ocean. *Remote Sensing*, 9(5):444, 2017.

[112] Iragavarapu Suryanarayana, Antonio Braibanti, Rupenaguntla Sambasiva Rao, Veluri Anantha Ramam, Duvvuri Sudarsan, and Gollapalli Nageswara Rao. Neural networks in fisheries research. *Fisheries Research*, 92(2-3):115–139, 2008.

[113] P. Lehodey, I. Senina, and R. Murtugudde. A spatial ecosystem and populations dynamics model (SEAPODYM)–modeling of tuna and tuna-like populations. *Progress in Oceanography*, 78(4):304–318, 2008.

[114] Peter Guang Yi Lu. Bayesian inference of stochastic dynamical models. Master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, February 2013.

[115] Abhinav Gupta. Bayesian Inference of Obstacle Systems and Coupled Biogeochemical-Physical Models. Master's thesis, IIT Kanpur, Kanpur India, 2016.

[116] James C Orr, Victoria J Fabry, Olivier Aumont, Laurent Bopp, Scott C Doney, Richard A Feely, Anand Gnanadesikan, Nicolas Gruber, Akio Ishida, Fortunat Joos, Robert M. Key, Keith Lindsay, Ernst Maier-Reimer, Richard Matear, Patrick Monfray, Anne Mouchet, Raymond G. Najjar, Gian-Kasper Plattner, Keith B. Rodgers, Christopher L. Sabine, Jorge L. Sarmiento, Reiner Schlitzer, Richard D. Slater, Ian J. Totterdell, Marie-France Weirig, Yasuhiro Yamanaka, and Andrew Yool. Anthropogenic ocean acidification over the twenty-first century and its impact on calcifying organisms. *Nature*, 437(7059):681–686, 2005.

[117] Scott C Doney, William M Balch, Victoria J Fabry, and Richard A Feely. Ocean acidification: a critical emerging problem for the ocean sciences. *Oceanography*, 22(4):16–25, 2009.

[118] Jeremy T Mathis, Sarah R Cooley, Kimberly K Yates, and Phillip Williamson. Introduction to this special issue on ocean acidification: The pathway from science to policy. *Oceanography*, 28(2):10–15, 2015.

[119] Richard A Feely, Christopher L Sabine, J Martin Hernandez-Ayon, Debby Ianson, and Burke Hales. Evidence for upwelling of corrosive "acidified" water onto the continental shelf. *science*, 320(5882):1490–1492, 2008.

[120] Dwight K Gledhill, Meredith M White, Joseph Salisbury, Helmuth Thomas, Ivy Mlsna, Matthew Liebman, Bill Mook, Jason Grear, Allison C Candelmo, R Christopher Chambers, et al. Ocean and coastal acidification off new england and nova scotia. *Oceanography*, 28(2):182–197, 2015.

[121] Stephanie C Talmage and Christopher J Gobler. Effects of past, present, and future ocean carbon dioxide concentrations on the growth and survival of larval shellfish. *Proceedings of the National Academy of Sciences*, 107(40):17246–17251, 2010.

[122] Geoffrey T Evans and VC Garçon. One-dimensional models of water column biogeochemistry. *JGOFS report*, 23(97):85, 1997.

[123] JR Palmer and IJ Totterdell. Production and export in a global ocean ecosystem model. *Deep Sea Research Part I: Oceanographic Research Papers*, 48(5):1169–1198, 2001.

[124] José Pinto Peixoto and Abraham H Oort. *Physics of climate*. New York, NY (United States); American Institute of Physics, 1992.

[125] Wei-Jun Cai, Xinping Hu, Wei-Jen Huang, Li-Qing Jiang, Yongchen Wang, Tsung-Hung Peng, and Xin Zhang. Alkalinity distribution in the western north atlantic ocean margins. *Journal of Geophysical Research: Oceans*, 115(C8), 2010.

[126] Yuri Artioli, Jeremy C Blackford, Momme Butenschön, Jason T Holt, Sarah L Wakelin, Helmuth Thomas, Alberto V Borges, and J Icarus Allen. The carbonate system in the north sea: Sensitivity and model validation. *Journal of Marine Systems*, 102:1–13, 2012.

[127] Rik Wanninkhof, Michelle Wood, and Leticia Barbero. Gulf of Mexico and East Coast Carbon Cruise - 2 (GOMECC-2), 2012.

[128] Charles Sun, A Thresher, R Keeley, N Hall, M Hamilton, P Chinn, A Tran, G Goni, L la VILLEON, T Carval, et al. The data management system for the global temperature and salinity profile programme. *Proceedings of OceanObs*, 9:86, 2010.

[129] Neal Pettigrew. Neracoos buoy b01 - western maine shelf, 2021.

[130] Frank Monaldo. Primer on the estimation of sea surface temperature using Terascan processing of NOAA AVHRR satellite data, January 1996.

[131] National Centers for Environmental Information (NCEI). World ocean database 2018, June 2020.

[132] P. F. J. Lermusiaux. Estimation and study of mesoscale variability in the Strait of Sicily. *Dynamics of Atmospheres and Oceans*, 29(2):255–303, 1999.

[133] Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

[134] Ernst Hairer, Syvert P. Norsett, and Gerhard Wanner. *Solving ordinary differential equations I.* Springer, 1993.

[135] Eleonora Musharbash and Fabio Nobile. Dual dynamically orthogonal approximation of incompressible navier stokes equations with random boundary conditions. *Journal of Computational Physics*, 354:135–162, 2018.

[136] Mulin Cheng, Thomas Y Hou, and Zhiwen Zhang. A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations i: Derivation and algorithms. *Journal of Computational Physics*, 242:843–868, 2013.

[137] Olivier P Le Maître, Matthew T Reagan, Habib N Najm, Roger G Ghanem, and Omar M Knio. A stochastic projection method for fluid flow: Ii. random process. *Journal of computational Physics*, 181(1):9–44, 2002.

[138] Rajat Mittal and Gianluca Iaccarino. Immersed boundary methods. *Annu. Rev. Fluid Mech.*, 37:239–261, 2005.

[139] Yu-Heng Tseng and Joel H Ferziger. A ghost-cell immersed boundary method for flow in complex geometry. *Journal of computational physics*, 192(2):593–623, 2003.

[140] George Casella and Roger L Berger. *Statistical inference*. Cengage Learning, 2021.

[141] Sri Venkata Tapovan Lolla. *Path Planning and Adaptive Sampling in the Coastal Ocean*. PhD thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, February 2016.

[142] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[143] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008.

[144] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions-I. *Mathematical programming*, 14(1):265–294, 1978.

[145] Ralph F Milliff, Jerome Fiechter, William B Leeds, Radu Herbei, Christopher K Wikle, Mevin B Hooten, Andrew M Moore, Thomas M Powell, and Jeremiah Brown. Uncertainty management in coupled physical-biological lower trophic level ocean ecosystem models. *Oceanography*, 26(4):98–115, 2013.

[146] Timothy DelSole. Predictability and information theory. part i: Measures of predictability. *Journal of the atmospheric sciences*, 61(20):2425–2440, 2004.

[147] Timothy DelSole and Michael K Tippett. Average predictability time. part i: theory. *Journal of the atmospheric sciences*, 66(5):1172–1187, 2009.

[148] Lai-Yung Leung and Gerald R North. Information theory and climate prediction. *Journal of Climate*, 3(1):5–14, 1990.

[149] Zhu Wang, Imran Akhtar, Jeff Borggaard, and Traian Iliescu. Proper orthogonal decomposition closure models for turbulent flows: a numerical comparison. *CMAME*, 237:10–26, 2012.

[150] J Nathan Kutz, Steven L Brunton, Bingni W Brunton, and Joshua L Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[151] Marcel Lesieur, Olivier Métais, et al. *Large-eddy simulations of turbulence*. Cambridge U. press, 2005.

[152] Giancarlo Alfonsi. Reynolds-averaged navier–stokes equations for turbulence modeling. *Applied Mechanics Reviews*, 62(4), 2009.

[153] FJ Los and M Blaas. Complexity, accuracy and practical applicability of different biogeochemical model versions. *Journal of Marine Systems*, 81(1-2):44–74, 2010.

[154] Eric P Chassignet and Jacques Verron. *Ocean modeling and parameterization*, volume 516. Springer Science & Business Media, 2012.

[155] Ben A Ward et al. When is a biogeochemical model too complex? objective model reduction and selection for North Atlantic time-series sites. *Progress in Oceanography*, 116:49–65, 2013.

[156] Suraj Pawar, Shady E Ahmed, Omer San, and Adil Rasheed. Data-driven recovery of hidden physics in reduced order modeling of fluid flows. *Physics of Fluids*, 32(3):036602, 2020.

[157] Zhong Yi Wan, Pantelis Vlachas, Petros Koumoutsakos, and Themistoklis Sapsis. Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PloS one*, 13(5), 2018.

[158] Omer San and Romit Maulik. Neural network closures for nonlinear model order reduction. *Advances in Computational Mathematics*, 44(6):1717–1750, 2018.

[159] Shaowu Pan and Karthik Duraisamy. Data-driven discovery of closure models. *SIAM Journal on Applied Dynamical Systems*, 17(4):2381–2413, 2018.

[160] Panos Stinis. Renormalized Mori–Zwanzig-reduced models for systems without scale separation. *Proceedings of the Royal Society A*, 471(2176):20140446, 2015.

[161] Alexandre J Chorin, Ole H Hald, and Raz Kupferman. Optimal prediction and the Mori–Zwanzig representation of irreversible processes. *PNAS*, 97(7):2968–2973, 2000.

[162] Ayoub Gouasmi, Eric J Parish, and Karthik Duraisamy. A priori estimation of memory effects in reduced-order models of nonlinear systems using the Mori–Zwanzig formalism. *Proceedings of the Royal Society A*, 473(2205):20170385, 2017.

[163] Hassler Whitney. Differentiable manifolds. *Annals of Mathematics*, pages 645–680, 1936.

[164] Floris Takens. Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980*, pages 366–381. Springer, 1981.

[165] Romit Maulik et al. Time-series learning of latent-space dynamics for reduced-order model closure. *Physica D: Nonlinear Phenomena*, 405:132368, 2020.

[166] Yibo Yang, Mohamed Aziz Bhouri, and Paris Perdikaris. Bayesian differential programming for robust systems identification under uncertainty. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 476(2243):20200290, 2020.

[167] Gavin D Portwood et al. Turbulence forecasting via neural ODE. *arXiv preprint:1911.05180*, 2019.

[168] A Otto, W Just, and G Radons. Nonlinear dynamics of delay systems: An overview. *Philosophical Transactions of the Royal Society A*, 377(2153):20180389, 2019.

[169] David S Glass, Xiaofan Jin, and Ingmar H Riedel-Kruse. Nonlinear delay differential equations and their application to modeling biological network motifs. *Nature communications*, 12(1):1–19, 2021.

[170] Isao T Tokuda, Ozgur E Akman, and James CW Locke. Reducing the complexity of mathematical models for the plant circadian clock by distributed delays. *J. of theoretical biology*, 463:155–166, 2019.

[171] Fariduddin Behzad et al. On the sensitivity and accuracy of proper-orthogonal-decomposition-based reduced order models for Burgers equation. *Computers & Fluids*, 106:19–32, 2015.

[172] Angel Borja et al. Tales from a thousand and one ways to integrate marine ecosystem components when assessing the environmental status. *Frontiers in Marine Science*, 1:72, 2014.

[173] A. R. Robinson, P. J. Haley, P. F. J. Lermusiaux, and W. G. Leslie. Predictive skill, predictive capability and predictability in ocean forecasting. In *Proceedings of "The OCEANS 2002 MTS/IEEE" conference*, pages 787–794. Holland Publications, September 2002.

[174] Pierre F. J. Lermusiaux, P. Malanotte-Rizzoli, D. Stammer, J. Carton, J. Cummings, and A. M. Moore. Progress and prospects of U.S. data assimilation in ocean research. *Oceanography*, 19(1):172–183, 2006.

[175] P. F. J. Lermusiaux, C.-S. Chiu, G. G. Gawarkiewicz, P. Abbot, A. R. Robinson, R. N. Miller, P. J. Haley, Jr, W. G. Leslie, S. J. Majumdar, A. Pang, and F. Lekien. Quantifying uncertainties in ocean predictions. *Oceanography*, 19(1):92–105, 2006.

[176] Nestor M Robinson et al. A systematic review of marine-based species distribution models (SDMs) with recommendations for best practice. *Frontiers in Marine Science*, 4:421, 2017.

[177] Philip Holmes et al. *Turbulence, coherent structures, dynamical systems and symmetry*. CUP, 2012.

[178] Hermann G Matthies and Marcus Meyer. Nonlinear galerkin methods for the model reduction of nonlinear dynamical systems. *Computers & Structures*, 81(12):1277–1286, 2003.

[179] PK Yeung et al. Effects of finite spatial and temporal resolution in direct numerical simulations of incompressible isotropic turbulence. *Physical Review Fluids*, 3(6):064603, 2018.

[180] Sylvain Laizet, Jovan Nedić, and Christos Vassilicos. Influence of the spatial resolution on fine-scale features in DNS of turbulence generated by a single square grid. *International Journal of Computational Fluid Dynamics*, 29(3-5):286–302, 2015.

[181] James C McWilliams. Submesoscale currents in the ocean. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 472(2189):20160117, 2016.

[182] James C McWilliams. Submesoscale surface fronts and filaments: secondary circulation, buoyancy flux, and frontogenesis. *Journal of Fluid Mechanics*, 823:391, 2017.

[183] Daniel P Dauhajre, James C McWilliams, and Lionel Renault. Nearshore lagrangian connectivity: submesoscale influence and resolution sensitivity. *JGR: Oceans*, 124(7):5180–5204, 2019.

[184] Tapio Schneider et al. Earth system modeling 2.0: A blueprint for models that learn from observations and targeted high-resolution simulations. *Geophy. Res. L.*, 44(24):12–396, 2017.

[185] Martin A Nowak. *Evolutionary dynamics: exploring the equations of life*. Harvard U. press, 2006.

[186] Robert M May. *Stability and complexity in model ecosystems*, volume 1. Princeton U. press, 2019.

[187] Allan R. Robinson and Pierre F. J. Lermusiaux. Data assimilation for modeling and predicting coupled physical–biological interactions in the sea. In Allan R. Robinson, James J. McCarthy, and Brian J. Rothschild, editors, *Biological-Physical Interactions in the Sea*, volume 12 of *The Sea*, chapter 12, pages 475–536. John Wiley and Sons, New York, 2002.

[188] P. F. J. Lermusiaux, C. Evangelinos, R. Tian, P. J. Haley, Jr, J. J. McCarthy, N. M. Patrikalakis, A. R. Robinson, and H. Schmidt. Adaptive coupled physical and biogeochemical ocean predictions: A conceptual basis. In *Computational Science - ICCS 2004*, volume 3038 of *Lecture Notes in Computer Science*, pages 685–692. Springer Berlin Heidelberg, 2004.

[189] Yang Kuang. *Delay differential equations: with applications in population dynamics*. AP, 1993.

[190] Luca Dell'Anna. Solvable delay model for epidemic spreading: the case of covid-19 in italy. *Scientific Reports*, 10(1), September 2020.

[191] Gennadii A Bocharov and Fathalla A Rihan. Numerical modelling in biosciences using delay differential equations. *Journal of Computational and Applied Mathematics*, 125(1-2):183–199, 2000.

[192] Willem Hundsdorfer and Jan G Verwer. *Numerical solution of time-dependent advection-diffusion-reaction equations*, volume 33. Springer Science & Business Media, 2013.

[193] Blaise Faugeras and Olivier Maury. Modeling fish population movements: from an individual-based representation to an advection–diffusion equation. *J. of Theoretical Bio.*, 247(4):837–848, 2007.

[194] Dmitri Kondrashov, Mickaël D Chekroun, and Michael Ghil. Data-driven non-Markovian closure models. *Physica D: Nonlinear Phenomena*, 297:33–55, 2015.

[195] N. Boers, M. D. Chekroun, et al. Inverse stochastic–dynamic models for high-resolution greenland ice core records. *Earth System Dynamics*, 8(4):1171–1190, 2017.

[196] Mickaël D Chekroun, Honghu Liu, and James C McWilliams. Variational approach to closure of nonlinear dynamical systems: Autonomous case. *Journal of Statistical Physics*, pages 1–88, 2019.

[197] Jean-Pierre Richard. Time-delay systems: an overview of some recent advances and open problems. *automatica*, 39(10):1667–1694, 2003.

[198] Odo Diekmann, Stephan A Van Gils, Sjoerd MV Lunel, and Hans-Otto Walther. *Delay equations: functional-, complex-, and nonlinear analysis*, volume 110. Springer Science & Business Media, 2012.

[199] Hal L Smith. *An introduction to delay differential equations with applications to the life sciences*, volume 57. Springer New York, 2011.

[200] Norman MacDonald. *Biological delay systems: linear stability theory*. CUP, 2008.

[201] Gilbert Koch et al. Modeling of delays in PKPD: classical approaches and a tutorial for delay differential equations. *Journal of pharmacokinetics and pharmacodynamics*, 41(4):291–318, 2014.

[202] Marc R Roussel. The use of delay differential equations in chemical kinetics. *The journal of physical chemistry*, 100(20):8323–8330, 1996.

[203] Andre A Keller. Generalized delay differential equations to economic dynamics and control. *American-Math*, 10:278–286, 2010.

[204] Keisuke Matsuya and Masahiro Kanai. Exact solution of a delay difference equation modeling traffic flow and their ultra-discrete limit. *arXiv preprint:1509.07861*, 2015.

[205] Karl Kunisch. Approximation schemes for the linear-quadratic optimal control problem associated with delay equations. *SIAM J. on Control and Optimization*, 20(4):506–540, 1982.

[206] Michael Ghil, Mickaël D Chekroun, and Gábor Stepan. A collection on 'climate dynamics: multiple scales and memory effects', 2015.

[207] K Bhattacharya, M Ghil, and IL Vulis. Internal variability of an energy-balance model with delayed albedo effects. *Journal of the Atmospheric Sciences*, 39(8):1747–1773, 1982.

[208] Chris Rackauckas et al. Diffeqflux.jl - A julia library for neural differential equations. *arXiv preprint:1902.02376*, 2019.

[209] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[210] Adam Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[211] Yulia Rubanova, Ricky TQ Chen, and David Duvenaud. Latent odes for irregularly-sampled time series. *arXiv preprint:1907.03907*, 2019.

[212] Jonathan Calver and Wayne Enright. Numerical methods for computing sensitivities for ODEs and DDEs. *Numerical Algorithms*, 74(4):1101–1117, 2017.

[213] Carl Wunsch. *The ocean circulation inverse problem*. Cambridge University Press, 1996.

[214] A. R. Robinson, P. F. J. Lermusiaux, and N. Q. Sloan III. Data assimilation. In Kenneth H. Brink and Allan R. Robinson, editors, *The Global Coastal Ocean-Processes and Methods*, volume 10 of *The Sea*, chapter 20, pages 541–594. John Wiley and Sons, New York, 1998.

[215] Amir Gholami, Kurt Keutzer, and George Biros. Anode: Unconditionally accurate memory-efficient gradients for neural odes. *arXiv preprint:1902.10298*, 2019.

[216] Andreas Griewank. Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation. *Optimization Methods and software*, 1(1):35–54, 1992.

[217] Talgat Daulbaev et al. Interpolation technique to speed up gradients propagation in neural ODEs. *Advances in Neural Information Processing Systems*, 33, 2020.

[218] H Rasmussen, GC Wake, and J Donaldson. Analysis of a class of distributed delay logistic differential equations. *Mathematical and computer modelling*, 38(1-2):123–132, 2003.

[219] Janni Yuval, Chris N Hill, and Paul A O'Gorman. Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision. *arXiv preprint:2010.09947*, 2020.

[220] Peter N Brown, George D Byrne, and Alan C Hindmarsh. VODE: A variable-coefficient ODE solver. *SIAM journal on scientific and statistical computing*, 10(5):1038–1051, 1989.

[221] Romit Maulik and Omer San. Explicit and implicit les closures for burgers turbulence. *Journal of Computational and Applied Mathematics*, 327:12–40, 2018.

[222] Jing Li and Panagiotis Stinis. Mori-zwanzig reduced models for uncertainty quantification. *Journal of Computational Dynamics*, 6(PNNL-SA-132853), 2019.

[223] Hans Burchard, Eric Deleersnijder, and Andreas Meister. Application of modified patankar schemes to stiff biogeochemical models for the water column. *Ocean Dynamics*, 55(3-4):326–337, 2005.

[224] M. P. Ueckermann and P. F. J. Lermusiaux. High order schemes for 2D unsteady biogeochemical ocean models. *Ocean Dynamics*, 60(6):1415–1445, December 2010.

[225] Eiji Mizutani and Stuart E Dreyfus. On complexity analysis of supervised MLP-learning for algorithmic comparisons. In *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, volume 1, pages 347–352. IEEE, 2001.

[226] Thomas Erneux. *Applied delay differential equations*, volume 3. Springer, 2009.

[227] Justin Sirignano, Jonathan F MacArt, and Jonathan B Freund. Dpm: A deep learning pde augmentation method with application to large-eddy simulation. *Journal of Computational Physics*, 423:109811, 2020.

[228] Priyabrata Saha and Saibal Mukhopadhyay. A deep learning approach for predicting spatiotemporal dynamics from sparsely observed data. *IEEE Access*, 9:64200–64210, 2021.

[229] Abhinav Gupta and Pierre F. J. Lermusiaux. Neural closure models for dynamical systems. *Proceedings of The Royal Society A*, 477(2252):1–29, August 2021.

[230] Jianhong Wu. *Theory and applications of partial functional differential equations*, volume 119. Springer Science & Business Media, 2012.

[231] YuFeng Shi, Biao Xu, and Yan Guo. Numerical solution of korteweg-de vries-burgers equation by the compact-type cip method. *Advances in Difference Equations*, 2015(1):1–9, 2015.

[232] Kumar Rahul and SN Bhattacharyya. One-sided finite-difference approximations suitable for use with richardson extrapolation. *Journal of Computational Physics*, 219(1):13–20, 2006.

[233] Pierre Sagaut. *Large eddy simulation for incompressible flows: an introduction*. Springer Science & Business Media, 2006.

[234] Yuriy Kochura, Yuri Gordienko, Vlad Taran, Nikita Gordienko, Alexandr Rokovyi, Oleg Alienin, and Sergii Stirenko. Batch size influence on performance of graphic and tensor processing units during training and inference phases. In *International Conference on Computer Science, Engineering and Education Applications*, pages 658–668. Springer, 2019.

[235] Deepak Narayanan Subramani. *Probabilistic Regional Ocean Predictions: Stochastic Fields and Optimal Planning*. PhD thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, February 2018.

[236] Deepak Subramani and P. F. J. Lermusiaux. Probabilistic ocean predictions with dynamically-orthogonal primitive equations. 2021. In preparation.

[237] Kyprianos Agioub Gkirgkis. Stochastic ocean forecasting with the dynamically orthogonal primitive equations. Master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, June 2021.

[238] Kyprianos A. Gkirgkis and Pierre F. J. Lermusiaux. Massive probabilistic forecasts for the gulf of mexico: Dynamically-orthogonal primitive equations. 2021. In preparation.

[239] Pierre F. J. Lermusiaux, Chris Mirabito, Patrick J. Haley, Jr., Wael Hajj Ali, Abhinav Gupta, Sudip Jana, Eugene Dorfman, Alison Laferriere, Aaron Kofford, G. Shepard, M. Goldsmith, Kevin Heaney, Emanuel Coelho, J. Boyle, J. Murray, L. Freitag, and A. Morozov. Real-time probabilistic coupled ocean physics-acoustics forecasting and data assimilation for underwater GPS. In *OCEANS 2020 IEEE/MTS*, pages 1–9. IEEE, October 2020.

[240] Pierre F. J. Lermusiaux, Manan Doshi, Chinmay S. Kulkarni, Abhinav Gupta, Patrick J. Haley, Jr., Chris Mirabito, Francesco Trotta, S. J. Levang, G. R. Flierl, J. Marshall, Thomas Peacock, and C. Noble. Plastic pollution in the coastal oceans: Characterization and modeling. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–10, Seattle, October 2019. IEEE.

[241] Wael Hajj Ali, Manmeet S. Bhabra, Pierre F. J. Lermusiaux, Andrew March, Joseph R. Edwards, Katherine Rimpau, and Paul Ryu. Stochastic oceanographic-acoustic prediction and Bayesian inversion for wide area ocean floor mapping. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–10, Seattle, October 2019. IEEE.

[242] Aaron Charous and Pierre F. J. Lermusiaux. Dynamically orthogonal differential equations for stochastic and deterministic reduced-order modeling of ocean acoustic wave propagation. In *OCEANS 2021 IEEE/MTS*, pages 1–7. IEEE, September 2021.

[243] Jacob P. Heuss, Patrick J. Haley, Jr., Chris Mirabito, Emanuel Coelho, Martha C. Schönau, Kevin Heaney, and Pierre F. J. Lermusiaux. Reduced order modeling for stochastic prediction onboard autonomous platforms at sea. In *OCEANS 2020 IEEE/MTS*, pages 1–10. IEEE, October 2020.

[244] Tony Ryu, Jacob P. Heuss, Patrick J. Haley, Jr., Chris Mirabito, Emanuel Coelho, Paul Hursky, Martha C. Schönau, Kevin Heaney, and Pierre F. J. Lermusiaux. Adaptive stochastic reduced order modeling for autonomous ocean platforms. In *OCEANS 2021 IEEE/MTS*, pages 1–9. IEEE, September 2021.

[245] Tony Ryu, Patrick J. Haley, Jr., Chris Mirabito, Aaron Charous, J. Metzger, G. Jacobs, J. Fabre, C. Trott, and Pierre F. J. Lermusiaux. Incremental low-rank dynamic mode decomposition model for efficient dynamic forecast dissemination and onboard forecasting. In *OCEANS 2022 IEEE/MTS*. IEEE, October 2022. Sub-judice.

[246] P. F. J. Lermusiaux, T. Lolla, P. J. Haley, Jr., K. Yigit, M. P. Ueckermann, T. Sondergaard, and W. G. Leslie. Science of autonomy: Time-optimal path planning and adaptive sampling for swarms of ocean vehicles. In Tom Curtin, editor, *Springer Handbook of Ocean Engineering: Autonomous Ocean Vehicles, Subsystems and Control*, chapter 21, pages 481–498. Springer, 2016.

[247] D. N. Subramani, P. F. J. Lermusiaux, P. J. Haley, Jr., C. Mirabito, S. Jana, C. S. Kulkarni, A. Girard, D. Wickman, J. Edwards, and J. Smith. Time-optimal path planning: Real-time sea exercises. In *Oceans '17 MTS/IEEE Conference*, Aberdeen, June 2017.

[248] Chinmay S. Kulkarni and Pierre F. J. Lermusiaux. Three-dimensional time-optimal path planning in the ocean. *Ocean Modelling*, 152, August 2020.

[249] Manan M. Doshi, Manmeet S. Bhabra, and Pierre F. J. Lermusiaux. Energy-time optimal path planning in dynamic flows: Theory and schemes. *Computer Methods in Applied Mechanics and Engineering*, 2022. Sub-judice.

[250] Manan Doshi, Manmeet Bhabra, Marius Wiggert, Claire J. Tomlin, and Pierre F. J. Lermusiaux. Hamilton–Jacobi multi-time reachability. In *IEEE CDC 2022 Cancún*, pages 1–8, December 2022. In press.

[251] Marius Wiggert, Manan Doshi, Pierre F. J. Lermusiaux, and Claire J. Tomlin. Navigating underactuated agents by hitchhiking forecast flows. In *IEEE CDC 2022 Cancún*, pages 1–8, December 2022. In press.

[252] Abhinav Gupta. *Scientific Machine Learning for Dynamical Systems: Theory and Applications to Fluid Flow and Ocean Ecosystem Modeling*. PhD thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, September 2022.

[253] Jeff A. Bilmes et al. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. *International Computer Science Institute*, 4(510):126, 1998.

[254] Petre Stoica and Yngve Selen. Model-order selection: a review of information criterion rules. *Signal Processing Magazine, IEEE*, 21(4):36–47, 2004.

[255] Andrew M Bradley. PDE-constrained optimization and the adjoint method. Technical report, Technical Report. Stanford University. https://cs.stanford.edu/∼ambrad, 2013.

[256] Svante Björklund. *A survey and comparison of time-delay estimation methods in linear systems*. Citeseer, 2003.

[257] Shengtai Li and Linda Petzold. Adjoint sensitivity analysis for time-dependent partial differential equations with adaptive mesh refinement. *Journal of Computational Physics*, 198(1):310–325, 2004.

[258] Yang Cao, Shengtai Li, and Linda Petzold. Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software. *Journal of computational and applied mathematics*, 149(1):171–191, 2002.