Deep learning based physical-statistics modeling of ocean dynamics

Marie Déchelle-Marquet

Sorbonne Université — LIP6 & ISIR — EDITE de Paris

THÈSE

MARIE DÉCHELLE-MARQUET

Deep Learning based Physical-statistics modeling of ocean dynamics

---

soutenue publiquement le

10 mai 2023

Devant le jury composé de

Cécile Mallet Université Versailles Saint-Quentin	Rapportrice
Marc Sebban Université Jean Monnet St-Étienne	Rapporteur
Claire Monteleoni INRIA	Examinatrice
Julien Le Sommer Institut des Géosciences de l'Environnement	Examinateur
Marina Levy Sorbonne Université	Directrice de thèse
Patrick Gallinari Sorbonne Université	Directeur de thèse

ABSTRACT

The modeling of dynamical phenomena in geophysics and climate is based on a deep understanding of the underlying physics, described in the form of PDEs, and on their resolution by numerical models. The ever-increasing number of observations of physical systems, the recent rise of deep learning and the huge computational power required by numerical solvers, which hinders the resolution of existing models, suggest that the future of physical models could be data-driven. But for this prognosis to come true, deep learning must tackle several challenges, such as the interpretability and physical consistency of deep models, still largely under-addressed by the deep learning community.

In this thesis, we address both challenges: we study the prediction of sea surface temperature (SST) using hybrid models combining a data-driven and a physical model. Ensuring the physical plausibility of hybrid models necessitates well-posing their learning: otherwise, the high versatility of neural networks may lead the data-driven part to bypass the physical part.

Our study is divided into two parts: a theoretical study on hybrid models, and a practical confrontation of our model on simulations of real data. First, we propose a new generic well-posed learning framework based on the optimization of an upper-bound of a prediction error. Second, we study real-like ocean observations of SST and velocity fields from the Gulf Stream current in the North Atlantic (from the NATL60 model). This application highlights the challenges raised by confronting physics aware learning to the complexity of real-world physics. It also raises issues such as model generalization, which we discuss as a possible perspective.

ACKNOWLEDGEMENTS

Je tiens tout d'abord à remercier mes encadrants de thèse, Patrick Gallinari et Marina Levy, pour m'avoir donné l'opportunité de réaliser cette thèse, puis pour leur accompagnement et leur soutien durant ces trois années.

Merci aux membres de mon jury, Marc Sebban, Cécile Mallet, Claire Monteleoni et Julien Le Sommer pour leur lecture, leurs remarques, leurs questions et l'intérêt qu'ils ont porté à mon travail.

Ces quelques années de thèse n'auraient pas été aussi joyeuses sans les doctorants de l'équipe MLIA. Merci à mon co-auteur Jérémie et à Étienne pour l'aide précieuse avec LaTeX. Merci infiniment à Agnès et Perrine pour leur soutien et le partage d'expérience. Merci à Tristan pour sa gentillesse infinie. Merci à Christophe pour les leçons de babyfoot.

Pour leur amitié sans faille, merci à Albane, Quitterie, Théis, Marine, Marion, Vincent, Estelle, Kamelia, Gauthier, Christelle, Suzanne et Athina. Merci à mes amis co-thésards Cyprien, Flavien et Arthur pour les nombreux déjeuners salvateurs.

Cette thèse aura été marquée par la pandémie, merci à mes colocataires et amis pour avoir rendu les confinements si agréables.

Merci à mes parents pour leur présence et leur écoute. Merci également pour leur aide incommensurable pour la préparation du pot, et pour leur aide en général.

Merci à mon frère pour sa présence, sa compréhension et pour la motivation à courir.

Je suis très reconnaissante à Marie-Jeanne Lesot pour le temps qu'elle m'a consacré dans les derniers mois de thèse, ses retours sur mon manuscrit, son aide dans la préparation de soutenance, et le temps passé à bavarder.

Merci à toutes les personnes que j'ai pu croiser au sein de l'altertour pour l'espoir et le courage qu'ils ont fait naître en moi.

Merci aux coordinatrices des Petits Débrouillards IDF. Vous croiser lors de préparations d'ateliers a grandement facilité mes derniers mois de thèse.

Merci également à tous les encadrants et participants du DU MSI de l'Université Paris Cité. Terminer ma thèse en formation n'a pas été facile, mais a été un immense bol d'air et une ouverture prometteuse vers le futur.

Ces années de thèse ont eu leur lot de lectures et de podcasts. Certains ont révolutionné ma vision du monde. Merci à Mona Chollet, Nicolas Bouvier, Richard Powers, Victoire Tuaillon et Claire Richard.

Enfin, merci à Gaël Faye, Idir, FIP et Radio Nova pour leur accompagnement quasiment quotidien durant ces trois années.

CONTENTS

LIST OF ABBREVIATIONS

BVP	Boundary Value Problem
CFL	Courant-Friedrichs-Lewy
CNN	Convolutional Neural Network
DE	Differential Equation
DL	Deep Learning
GAN	Generative Adversarial Network
GHG	Greenhouse Gas
GNN	Graph Neural Network
GPU	Graphics Processing Unit
IPCC	Intergovernmental Panel on Climate Change
INR	Implicit Neural Representations
IVP	Initial Value Problem
MB	Model Based
ML	Machine Learning
MSE	Mean Squared Error
NATL	North Atlantic (model)
NEMO	Nucleus for European Modelling of the Ocean
NN	Neural Network
ODE	Ordinary Differential Equation
OGCM	Ocean General Circulation Model
OOD	Out Of Distribution
PDE	Partial Differential Equation
PINN	Physics Informed Neural Network
RK	Runge Kutta
RNN	Recurrent Neural Network
SSH	Sea Surface Height
SST	Sea Surface Temperature

Part I

MOTIVATION

In this part, we motivate and contextualize our work. First, in chapter 1, we introduce the scientific context of this thesis and summarize our key contributions. Second, in chapter 2, we present the technical background required for the thesis work and we summarize the current state of the literature.

1

INTRODUCTION

1.1 CONTEXT

Machine learning (ML), which deals with building computer systems that automatically improve through experience, is widely regarded as one of the most important disruptive technologies of our time (Jordan and Mitchell, 2015). Today, the adoption of data-intensive machine-learning methods can be found throughout science, technology and commerce. ML development is currently extended and amplified with the rise of deep learning (DL) and the use of deep artificial neural networks usually optimized via gradient descent techniques (Goodfellow, Bengio, and Courville, 2016).
Deep learning methods rely on multiple levels of representation to recognize patterns in data: starting from the raw input data, each level transforms its input into a representation at a higher, slightly more abstract level (LeCun, Bengio, and G. Hinton, 2015). The composition of multiple transformations makes it possible to learn very complex functions, which are, most of the time, difficult to translate into an an intuitive analytic form. The use of hardware accelerators such as Graphics Processing Units (GPU) and the creation of efficient frameworks (Paszke et al., 2019) made possible the development of deep learning systems that contain billions of parameters and that can be trained on very large collections of data such as images, videos, and speech samples. This way, the last decade has been marked by numerous scientific and technologic breakthroughs in many domains such as computer vision (Krizhevsky, Sutskever, and G. E. Hinton, 2012; Szegedy et al., 2017), natural language processing (Collobert and Weston, 2008; Y. Wu et al., 2016) or even health science (Leung et al., 2014).
Following major advances tackling longstanding problems in the artificial intelligence community (Silver et al., 2016), one now questions whether these advancements may be applied to other fields, such as the natural sciences and the study of physical processes and systems.

1.1.1 *ML for physics*

The understanding and prediction of the world has long stirred human heart and thus guided scientific research. Thus, fields such as geophysics, astronomy, epidemiology or chemical kinetics have been studied for centuries and are now dominated by mechanistic models (e.g. first principles) based on a deep understanding of the underlying phenomena, mathematically translated as statistical and/or physical relationships, i.e. laws. The increasing availability of supercomputing power in the 1970s made feasible the development of numerical simulations (Lynch, 2008), relying on the assimilation of large amounts of data to model the physical system evolution through time (Bauer, Thorpe, and Brunet, 2015). Such tools have shown great outcomes: weather prediction models have achieved unprecedented performances over the past 40 years, and are at the core of climate models used by the Intergovernmental Panel on Climate Change (IPCC) for climate change monitoring (Stockhause and Lautenschlager, 2017).
However such approaches are now reaching their limits: on the one hand in terms of tools and technologies, traditional models are limited due to their computational cost; on the other hand in

terms of physical knowledge, due to a weak understanding of certain processes, models are loosy approximations of the reality. Besides, in the era of satellite imagery, a deluge of Earth system data has become available, with storage volumes already well beyond dozens of petabytes (Agapiou, 2017), and extracting interpretable information and knowledge from such data would help in advancing scientific discovery. More than just an extension of computing power, the efficiency of the algorithms used to translate dynamical laws into practical computation will have a direct impact on the efficiency of existing models.
Having already found tremendous success in numerous applications, ML could help and play an important role in the future of physical modeling. Indeed, AI-based method have recently shown great success in weather prediction: outperforming state-of-the-art numerical weather prediction (Barros et al., 1995) for the first time (Bi et al., 2022; Lam et al., 2022).

1.1.2 *Physics-guided ML and Hybrid Modeling*

Even though the use of data-driven methods in Earth geosciences is rapidly growing, it is still in its early stages and makes uneven progresses (Bergen et al., 2019). Whereas data-driven approaches are model-agnostic interpretation methods (black-box models), representing the physical world requires interpretable models. To derive models which learn from data while still respecting our evolving understanding of nature's laws, the ML community is confronted with new challenges including : 1. to make use of available physical knowledge, 2. to produce physically consistent models. In this regard, a new paradigm arises, making use of domain-specific knowledge and integrating scientific knowledge directly into the ML framework (Bergen et al., 2019). Those new approaches, referred to as physics-guided or physics-based ML (Willard et al., 2020), are fundamentally different from mainstream purely data-driven practices and it has expanded in recent years.
Within this flourishing literature, an emerging field of research is the coupling of physical process models with data-driven ML (Reichstein et al., 2019). In this perspective, ML is seen as a complementary approach to traditional physics based models (Dueben and Bauer, 2018). Both offer advantages: whereas traditional approaches generalize and extrapolate better, high expressive ML approaches benefit from the ongoing growth of available data such as satellite observations, with reduced costs. This is referred to as hybrid modeling, and this is the approach we will focus on throughout this thesis.

1.1.3 *The "reality gap"*

Physical phenomena are based on processes involving multiple scales and variables : for instance, processes relevant to understanding Earth's geosystem behavior range in spatial scale from the atomic to the planetary, in temporal scale from milliseconds to billions of years and accounts for dynamic, thermodynamic, radiative and chemical processes (Bergen et al., 2019). However, most physics-guided ML publications experiment on toy problems in low-dimensional settings and develop models which are not directly applicable on complex settings and real-case scenario. Despite the promise shown by early proof-of-concept studies, the community has been slow to adopt ML more broadly. We inscribe our thesis within this burgeoning research challenge by studying how deep learning models developped for academic cases may apply to real settings. This work is at the frontier of real-world research. We conduct an incremental study and explore the potential of deep learning to complete physical models of the oceanic system. The next

section is devoted to a brief overview of the ocean modeling challenges. Then, we introduce more thoroughly this thesis subject and contributions.

1.2 ML FOR EARTH SYSTEM MODELING

This thesis aims at advancing towards bridging the gap between climate sciences and deep learning. To assess the ability of our models to capture real-world phenomena, we tackle problems in the field of physical oceanography, that is the study of the physics of marine systems, which is necessary to sustain life on our planet and plays a critical role in Earth's climate.

1.2.1 *The Ocean within the Earth System*

The climate is driven by the energy that we receive from the sun. This energy is unevenly distributed over the globe, the tropics are favored over the poles. In this context, the ocean plays two major roles: 1. thanks to its high heat capacity, the ocean's tropical waters absorb the excess of radiation from the sun. Around the equator, the ocean thus acts as a massive heat reservoir. 2. the non-uniform heating leads to temperature differences throughout the globe, which the ocean acts to reduce by transporting heat from the warm tropics to the cool poles. In other words, the ocean moderates the climate by taking in heat when the overlying atmosphere is hot, storing that energy and releasing heat when the atmosphere is cold (Vallis, 2011). It acts both diurnally (i.e., the day–night contrast) and annually (the seasonal cycle) (Edenhofer et al., 2011). The higher heat capacity of oceans explains why maritime climates tend to be less extreme than continental ones, with smaller day–night and winter–summer differences.
The heat is redistributed thanks to the large-scale ocean circulation : weather patterns are thus mainly driven by ocean currents (Vallis, 2011), i.e. patterns of water movement created by surface winds but also partly by temperature and salinity gradients, Earth's rotation, and tides. For instance, the famous Gulf Stream moves from the tropics northward through the Atlantic. There it bathes the shores of Western Europe, where the climate is surprisingly mild for that latitude. The ocean circulation thus plays a critical part of modeling the overall earth system (Chassignet, Le Sommer, and Wallcraft, 2019) and has been a major field of study for years.

1.2.2 *Large Scale Ocean Circulation*

The motion of the ocean is described by ocean currents. Those are driven by three main factors: 1. tides 2. wind 3. heat and salinity. Whereas tidal currents are strongest near the shore, in bays and estuaries along the coast, wind is responsible for surface motions on global scale, e.g. the Gulf Stream in the North Atlantic ocean or the Kuroshio current off the coast of Japan. Taken together, larger and more permanent currents make up the systems of currents known as gyres. Figure 1 shows that major gyres flow clockwise in the northern hemisphere and counterclockwise in the southern hemisphere. This is due to the Earth's rotation: because the Earth rotates on its axis, circulating air (i.e. wind) is deflected toward the right in the Northern Hemisphere and toward the left in the Southern Hemisphere. This deflection is called the Coriolis effect(Figure 2). Finally, changes in heat and

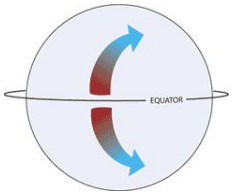

Figure 2: The Coriolis effect. From NOAA (2022).

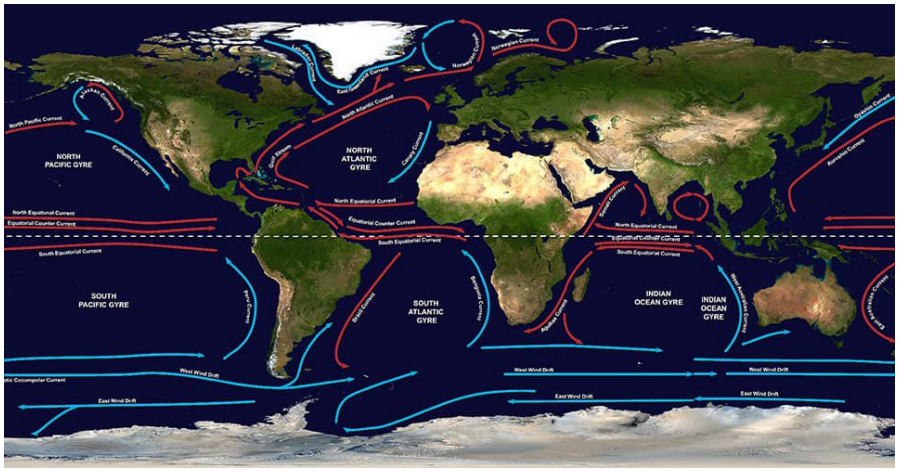

Figure 1: There are five major gyres: the North and South Pacific Subtropical Gyres, the North and South Atlantic Subtropical Gyres, and the Indian Ocean Subtropical Gyre. From (NOAA, 2021).

salt content constantly change the density of ocean water: while cold and salty water is dense and sinks to the bottom of the ocean, warm and fresh water surfaces. Heat and salinity are thus responsible for the deep circulation, known as the thermohaline circulation ("thermo" referring to temperature, "haline" to saltiness). The sinking and rising of ocean waters up and down on a global scale creates what is known as the "great ocean conveyor belt": it takes about 1500 years for water particles to travel across the entire planet.

1.2.3 *Brief overview of ocean modeling*

Ocean modeling has a long history. First studies were rather experimental: the first map of the Gulf Stream was published in 1769 (Figure 3), and the first wind and currents map for the northern Atlantic Ocean was established in 1848 (Figure 4). More theoretically, Laplace (1799) described the ocean tides in 1799. Some years later, Navier (1823) and Stokes (1880) express the conservation of mass, momentum and energy for Newtonian fluids. Those laws are famously referred to as Navier–Stokes equations. A major breakthrough was achieved in 1901, when Abbe (1901) proposed to use the laws of physics to forecast the weather. Predicting the state of the ocean could be treated as an initial value problem of mathematical physics, wherein future is determined by integrating the governing laws, starting from the observed current state. However, because of the mathematical intractability of obtaining analytical solutions to Navier-Stokes equations, it wasn't until the second half of the 20th century that Bryan (1969) proposed numerical solutions, using spatial and temporal discretization. This lays out the basis for computational fluid dynamics and the underlying principles of the algorithmic formulation of ocean circulation models, i.e. a numerical model that represents the movement of water in the ocean (Chassignet, Le Sommer, and Wallcraft, 2019). Nowadays, ocean general circulation models describe physical and thermodynamical processes in the ocean based on a three-dimensional grid. NEMO ("Nucleus for European Modeling of the Ocean") is a state-of-the-art modeling framework

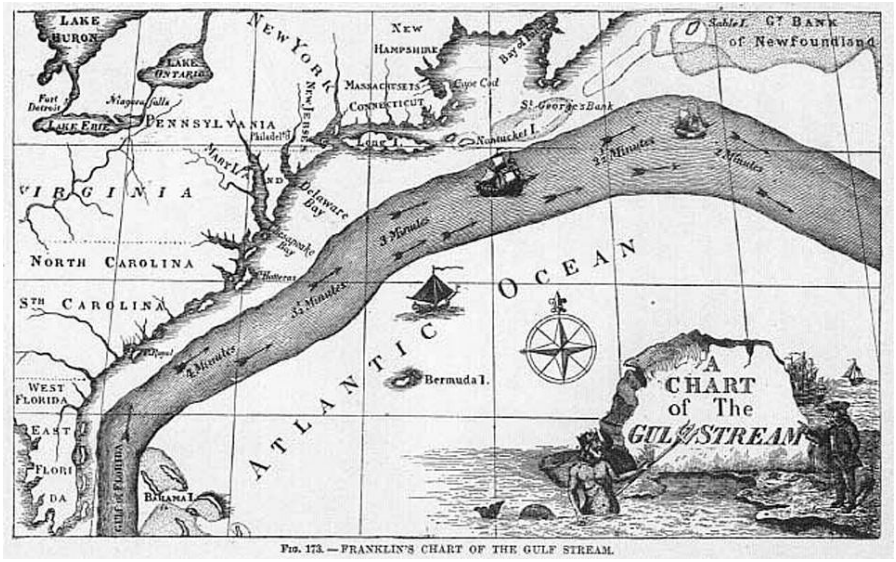

FIG. 173.—FRANKLIN'S CHART OF THE GULF STREAM.

Figure 3: While searching for the legendary Fountain of Youth, the Spanish conquistador Juan Ponce de León landed in Florida in 1513. On his way, he encountered a very strong sea current off the coast that prevented his ships from moving forward. He had just come across the Gulf Stream. This current was then reported by many other sailors, until Benjamin Franklin (1706–1790) made the first scientific studies: he measured its temperature on several occasions during his Atlantic crossings, and published the first map of the Gulf Stream in 1769.

for research activities and forecasting services in ocean and climate sciences, developed in a sustainable way by a European consortium (Madec et al., 2017).

1.2.4 *Case Study*

Because ocean currents are moving bodies of water, they result in changes in the ocean surface. In turn, this alters ocean surface topography by a few tens of centimeters to more than a meter. For instance, in the northern hemisphere, the gyre rotates clockwise, bringing water into the sea level, which thus rises. Since the early 1990s, global surface currents velocity fields are inferred from satellite observations of sea surface heights (SSH) (Dohan and Maximenko, 2010), i.e. altimetry data(see Figure 5). Such approximation is used to estimate the dynamics of slow, large-scale currents (Sinha and Abernathey, 2021), i.e. spatial and temporal resolution of about respectively 50km and a week. However, the dynamics of currents is influenced by phenomena operating at much finer scales of the order of 1km $\times$ 1day (Lévy, P. Klein, and Treguier, 2001). Whereas altimetry does not provide any information on those thin structures, high resolution satellite imagery could be used to infer fine scale spatio-temporal evolution. Indeed, SST and surface chlorophyll satellite images are now available daily at kilometric resolution. In this thesis, we consider ML algorithms as an alternate route to infer surface currents from high resolution satellite observables quantities such as the Sea Surface Temperature (SST). More specifically, we explore the dependencies between the SST and the currents velocity fields and investigate hybrid physico-statistical models to represent the evolution of both the observable SST and unobserved currents velocity.

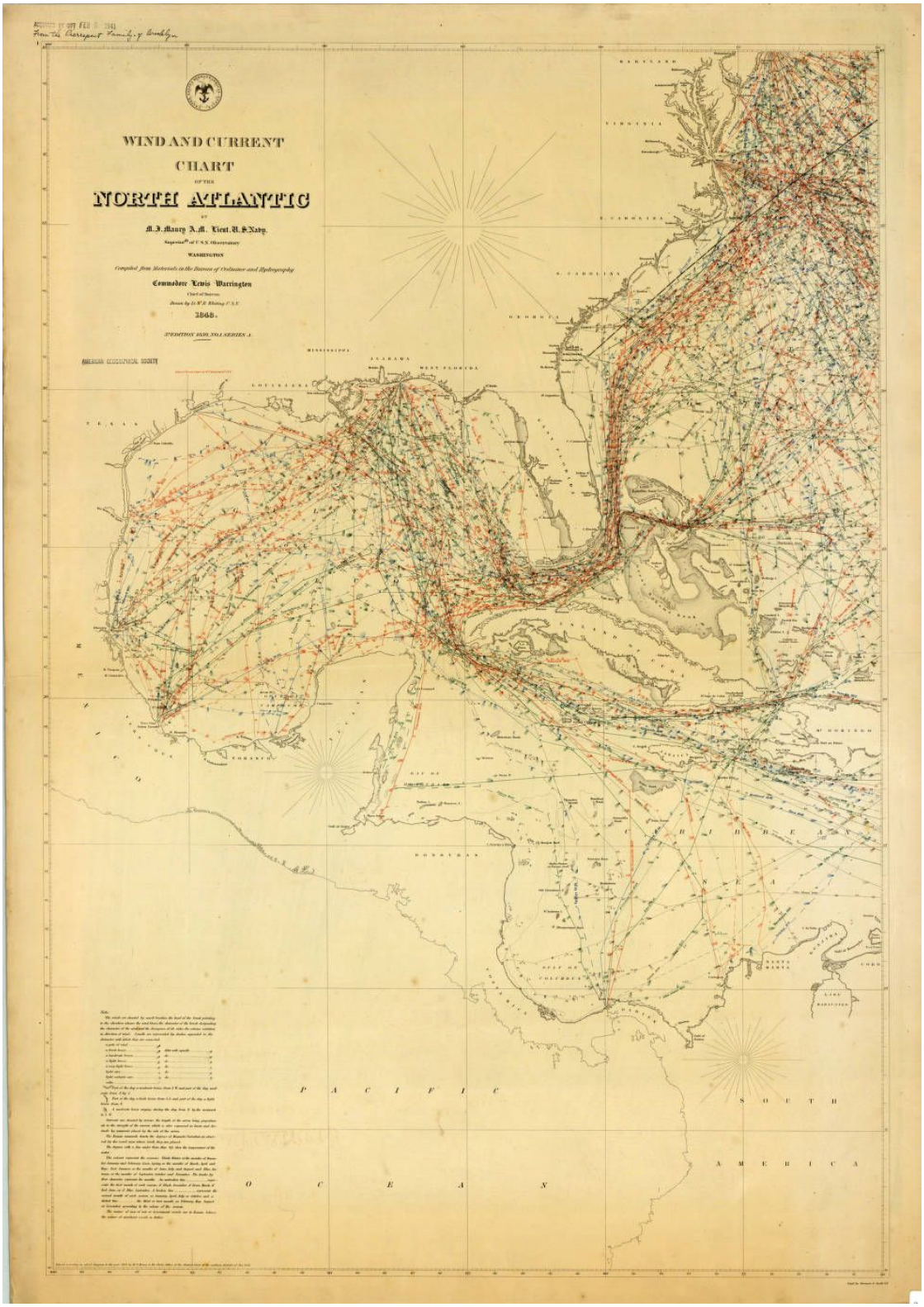

Figure 4: In 1848, Matthew Fontaine Maury, an officer from the US navy considered as the father of modern oceanography, systematically evaluated recorded information to create the first wind and current map for the northern Atlantic Ocean.

.

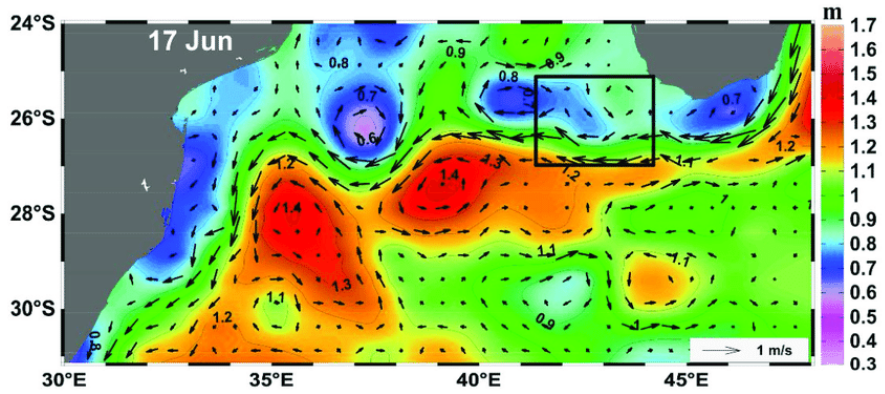

Figure 5: Sea Surface Height (colour contours) and geostrophic velocity (black arrows) over the Mozambique Basin on 17 June 2013. Surface currents travel between the troughs and hills of the sea surface height. From Lamont, Barlow, and Brewin (2018).

Note that, even if we rely on this concrete problem, this thesis is above all methodological. Indeed, we propose very general models applicable to many physical problems. Nonetheless, to understand the intuition of the presented models, we will refer to the prototype problem, the modeling of marine currents from SST data with neural networks, throughout our work.

1.3 CONTRIBUTIONS

We aim at modelling ocean currents velocity fields from sea surface temperature observations using deep learning. As we focus on a real problem, ensuring the physical soundness of our results is essential. However, Kirchmeyer et al. (2022) and Willard et al. (2020) underlines the lack of generalizability of black-box ML models and their inability to produce physically sound results. As ML alone ignores the fundamental laws of physics and can result in ill-posed problems or non-physical solutions (Alber et al., 2019), we will rely on hybrid models, using both prior physical knowledge and machine learning. In this field of research, one of the main challenges is to solve the ill-posedness inherent to the decomposition between physical and data-based models. For instance, this can be done by incorporating physically motivated constraints in the learning of hybrid models, e.g. through regularization penalties. To complete prior physical knowledge with a data-driven component and ensure interpretability of the decomposition, we first consider a simplified model of the ocean dynamics. Within this context, we introduce two contributions.

REGULARIZING HYBRID DYNAMICAL MODELS Incorporating physical knowledge, our framework considers the dynamics of the observed SST and its known dependency to the unknown velocity. Looking at real ocean dynamics equations, the current velocity dynamics should follow an ordinary differential equation (ODE) (defined in chapter 2). To cope with the ill-posedness inherent to hybrid modeling, we introduce a dynamical regularization on the estimated velocity, enforcing it to follow an ODE. This contribution, detailed in chapter 3 of this thesis, led to the following publication in an international conference workshop.

Marie Déchelle et al. (2020). "Bridging Dynamical Models and Deep Networks to Solve Forward and Inverse Problems". In: *NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning*.

FRAMEWORK FOR THE LEARNING OF HYBRID MODELS Even though it enables sound experimental results, the proposed regularization has no theoretical basis and shall in no case be sufficient to ensure well-posedness in the hybrid approach. To recover well-posedness and interpretability in the learning of hybrid models, we propose to control an upper bound of the prediction error and introduce a novel alternate-optimization algorithm for minimizing this bound. This contribution, detailed in chapter 4 of this document, led to the following publication in an international conference.

Jérémie Dona*, Marie Déchelle*, Marina Levy, et al. (2021). "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction". In: *International Conference on Learning Representations*.

REAL CASE STUDY Finally, we confront the proposed framework to the modeling of real ocean dynamics. We study the limitations of the previous approach and propose an adaptation of our model, incorporating new additional sources of physical knowledge. This improves the performances compared to the vanilla theoretical model. This contribution is detailed in the chapter 5 of this document.

This thesis is divided into three parts. Part i motivates our work: it encompasses chapter 1 and chapter 2. Whereas the former was a global introduction, the latter explains the current state of the literature and the necessary background for the description of our contributions. We then introduce our theoretical approaches, proposed in part ii, divided into chapters 3 and 4, and then a confrontation to real world data, proposed in part iii, in which chapter 5 explores the application of the derived models to real-world like simulations. Finally, supplementary material for chapters 2 to 5 is given in appendices A to C.

2

BACKGROUND AND RELATED WORK

In this chapter, we expose and contextualize the principal notions that are employed and explored in the rest of this work. We first address in section 2.1 the relationship between physics and deep learning: physical modeling is based on the numerical resolution of differential equations, on which we give a brief overview; tackling physical modeling with deep learning encompasses many issues, which we present with the associated architectures. Then, in section 2.2, we give an overview on the variables and equations used to describe ocean surface dynamics. We also introduce our datasets and the simplifying assumptions on which they rely. Finally, in section 2.3, we introduce the formalism, notations and objectives of this thesis.

2.1 DEEP LEARNING AND PHYSICS

We give in section 2.1.1 an overview of differential equations used together with neural networks in the context of temporal modeling, a topic that has been increasingly studied throughout recent years. From such a perspective, we review the various neural networks architectures designed to learn dynamics associated to ODEs. We also briefly outline the solving of differential equations with neural networks. Then, in section 2.1.2, we address the incorporation of physics into the learning framework with a focus on hybrid modeling. In section 2.1.3, we review the main issues ML faces when handling real data. Finally, we give insights on climate neural models.

2.1.1 *Neural Differential Equations*

Differential equations (DEs) characterize the evolution of systems in many fields such as physics, chemistry and biology (Alber et al., 2019). Physical modeling traditionnally relies on understanding of the physical phenomena, which are mathematically formalized with Ordinary or Partial Differential Equations (ODE/PDE). While ordinary differential equations are functions of a single variable, partial differential equations depend on several dimensions. As methods exist to reduce PDEs to a single continuous dimension and thus assimilate them to ODEs, we will only give a theoretical introduction to generic ODEs. We will then present the link established between ODEs and Neural Networks.

Ordinary Differential Equations (ODEs)

An ordinary differential equation (ODE) is a relation that contains functions of only one independent variable, in this thesis that is the time t, and one or more of its derivatives with respect to that variable, i.e. $\frac{\mathrm{d}^k y}{\mathrm{d}t^k}(t)$. With $y : [0, T] \to \mathbb{R}^d$ an unknown function and f a function of t, y, and derivatives of y, an ODE is expressed as

$$\frac{\mathrm{d}^n y}{\mathrm{d}t^n}(t) = f\left(t, y(t), \frac{\mathrm{d}y}{\mathrm{d}t}, \frac{\mathrm{d}^2 y}{\mathrm{d}t^2}, \dots, \frac{\mathrm{d}^{n-1} y}{\mathrm{d}t^{n-1}}\right) \tag{1}$$

Considering

$$\begin{aligned} \mathbf{y}: \quad & \mathbb{R}^d \to \mathbb{R}^n \\ & y \quad \mapsto \left(y(t), \frac{\mathrm{d}y}{\mathrm{d}t}, \frac{\mathrm{d}^2y}{\mathrm{d}t^2}, \dots, \frac{\mathrm{d}^{n-1}y}{\mathrm{d}t^{n-1}}\right) \end{aligned} \tag{2}$$

we can rewrite eq. (1) into

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{f}(t, \mathbf{y}) \tag{3}$$

In the rest of this document, to simplify notations, we write

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y) \tag{4}$$

Equation (4) associated with an initial condition $y_0 \in \mathbb{R}^n$ form an initial value problem (IVP):

$$\begin{aligned} \frac{\mathrm{d}y}{\mathrm{d}t} &= f(t, y) \\ y(t_0) &= y_0 \end{aligned} \tag{5}$$

As soon as f is continuous in t and Lipschitz continuous in y, the Picard-Lindelöf theorem guarantees that eq. (5) admits a unique solution Let $y^\star$ be this solution. We can then define the flow ϕ_f (illustrated in Figure 6) associated to eq. (5) such that:

$$\begin{aligned} \phi_f : [0,T]\times\mathbb{R}^n &\to \mathbb{R}^n \\ t, \quad y_0 &\mapsto y^\star(t) \end{aligned} \tag{6}$$

Usually, solutions of IVPs does not have a closed-form and are approximated using numerical schemes, the simplest one being the Euler Method (Butcher, 2016). Given an initial condition $(t_0, y_0) \in [0, T] \times \mathbb{R}^n$, this method provides a sequence $(y_i)_{i\in\mathbb{N}}$ of approximations of the value $y(t_i)$ that the solution of the equation corresponding to this initial condition takes. The approximations y_i are given by the recurrence relation

$$y_{i+1} = y_i + (t_{i+1} - t_i) f(t_i, y_i)$$

Reindexing y in the time domain, this can be translated into:

$$y_{t+\Delta t} = y_t + \Delta t f(t, y_t) \tag{7}$$

When solution and derivative values are specified at more than one point, we talk about boundary value problem (BVPs). However, there is no general theory for the existence and uniqueness of BVPs solutions.

Including the dependance of y to other variables such as space, eq. (4) becomes a partial differential equation (PDE), and depends on partial derivatives $\frac{\partial}{\partial x}$. In $\mathbb{R}^n$ with coordinates $(x_1, x_2, \dots, x_n)$, we write the first order derivatives as

$$\nabla = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \dots, \frac{\partial}{\partial x_n}\right) \tag{8}$$

As is the case for ODEs, analytic solutions can usually not be obtained. PDEs resolution also rely on numerical methods, such as finite differences, finite volums, finite elements or spectral

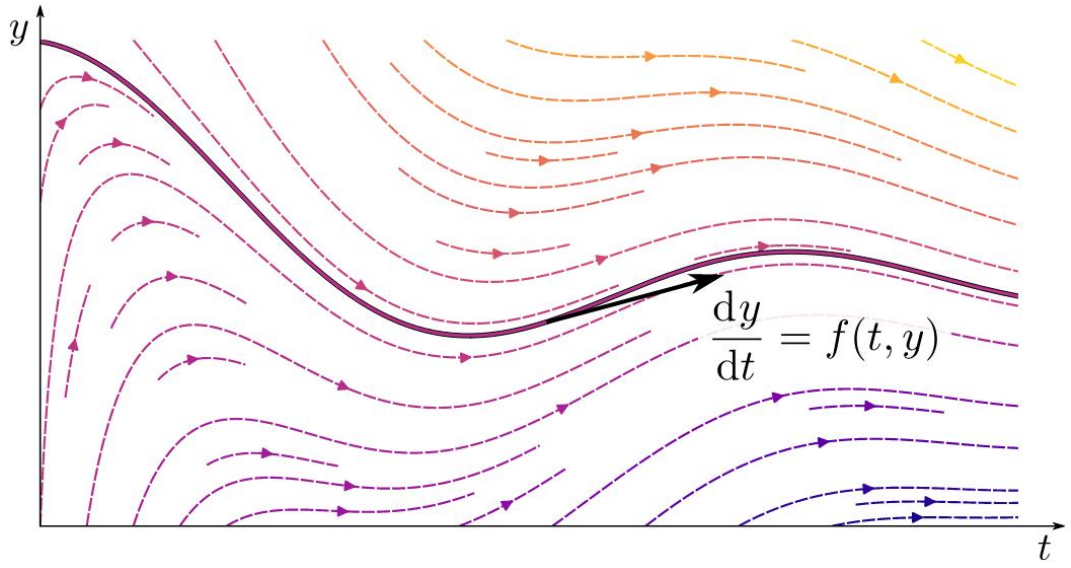

Figure 6: Illustration (in dashed lines) of the continuous flow of an ODE, with a particular solution that is plotted with a solid thicker line. Each dashed lines correspond to a different initial condition y_0. The black arrow represents the tangent to the highlighted solution, which is fully determined by its derivative and initial condition according to variants of the Picard-Lindelöf theorem (Demailly, 2016). In this example, f is defined as $f\colon (t, y) \mapsto \frac{1}{t}(\cos t - y)$, the ODE admitting as solutions functions $y_C\colon t \mapsto \frac{C}{t} + \operatorname{sinc} t$ for all $C \in \mathbb{R}$ over the time domain $[0, +\infty[$.

methods. A necessary condition for the convergence of the explicit time integration methods is given by the Courant-Friedrichs-Lewy (CFL) condition, involving the time-step Δt and length interval Δx. However, some numerical schemes, such as the semi Lagrangian scheme, make it possible to avoid such conditions. Unlike Eulerian scheme, relying on time discretization of the time derivative, the semi Lagrangian scheme relies on the constancy of the solution of a PDE along a characteristic curve. Further insights on such a scheme are given in appendix A.1. In our work, we consider PDEs with boundary value conditions, which we develop in section 2.2 and aim to learn their associated dynamics. This is a longstanding task in the ML community, on which we give an overview below.

Learning Dynamics with Neural Networks

Within this section, we consider dynamical systems obeying an ODE or a PDE. In the case of a PDE, the PDE is considered to be transformed into an ODE, for example with the method of lines (Hamdi, Schiesser, and Griffiths, 2007). This consists in discretizing the space and considering only the time variable. This allows the PDE to be replaced by an IVP or BVP described by a set of ODEs. We thus assume that the physical process can be accurately modeled by a differential equation of the form of eq. (4), i.e. it follows a dynamics f:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y).$$

In this context, our objective is to forecast the system state, i.e. to make prediction of the system state y: from a sequence of observations $y_{t-\tau:t}$, one aims at predicting $y_{t+1:t+\tau'}$. To that purpose, one can learn the dynamics of the studied system by parameterizing f with a neural network f_θ. Learning f_θ amounts to approximating f from eq. (4) through a parametric class of functions $\mathcal{F}(\theta)$, large enough to represent a wide variety of dynamics f. Note that, as the literature in this area is very extensive, this section is not intended to be exhaustive.

To learn dynamics from data is a longstanding problem within the ML community (Alvarez, Luengo, and Lawrence, 2013; Crutchfield and McNamara, 1987) and this has first been addressed with Recurrent Neural Networks (RNN). Indeed, Funahashi and Nakamura (1993) showed that any finite trajectory associated to a dynamical system can be learned with RNN. These models have shown impressive results on dynamical systems learning such as in Jia, Zwart, et al. (2021) and Wan et al. (2018).

A number of works have noticed the links between spatial derivatives in PDEs and convolutional networks (Z. Long, Lu, and Bin Dong, 2019; Z. Long, Lu, Ma, et al., 2018b; Ruthotto and Haber, 2020) due to the relationships between convolutions and finite difference approximation methods, thereby explaining the utility of convolutions even in latent spaces when it comes to predicting complex spatiotemporal phenomena (Ayed et al., 2020; X. Shi et al., 2015).

While all the aforementioned papers rely on a regular grid sampling, other approaches rely on Graph Neural Networks (GNN) (Zhou et al., 2020) to address non uniform sampling. For example, Brandstetter, Worrall, and Welling (2022) and Pfaff et al. (2020) modeled high-dimensional dynamics with GNN.

Recently, there has been a regain of interest for neural prediction of dynamical systems with the interpretation of ResNets as dynamical systems. This opened the door to several studies on training stability and the understanding of deep learning through the lens of physics. We introduce below this topic.

RESIDUAL NEURAL NETWORKS To overcome vanishing gradient in deep networks, He et al. (2016) proposed to introduce identity mappings, so that each layer only fits a residual mapping. The activation of the kth layer of ResNet is thus defined as

$$y_{k+1} = y_k + f(y_k, \theta_k) \tag{9}$$

where k is the kth layer of the network. With $\Delta t = 1$ and k integration step, this equation is similar to the Euler scheme defined in eq. (7), even though the use of ResNet for ML tasks was not initiated by the analogy with numerical schemes. As previously stated, it was rather considered for its stability during training: solving vanishing gradient problems enabled very deep architectures (Zaeemzadeh, Rahnavard, and Shah, 2020). It was later that Weinan (2017) established that residual neural network appeared to follow the modelling pattern of an ODE. Finally, the ODE eq. (4) can be formulated as the continuum limit of the ResNet associated with eq. (9).

NEURAL ODE This idea was exploited by R. T. Q. Chen et al. (2018), who leveraged ODEs to model the continuous temporal evolution of NN hidden states, y, given as initial condition the input of the NN f_θ, y_0:

$$\begin{aligned} \frac{\mathrm{d}y}{\mathrm{d}t} &= f_\theta(t, y) \\ y(t=0) &= y_0 \end{aligned} \tag{10}$$

This is possible as standard architectures of neural networks are Lipschitz-continuous (Gouk et al., 2021) and we are thus in the conditions of the Picard-Lindelöf theorem (see section 2.1.1). The idea behind Neural-ODE is to view the transformation made by each layer of the neural network f_θ as an evolution of the hidden state through time. In other words, R. T. Q. Chen et al. (2018) consider the dynamics of the hidden state with respect to the network depth, and take the analogy between ResNet and numerical schemes a step further, involving the continuous limit of a ResNet. From such perspective and considering the depth of a neural network as continuous, learning f_θ can be reformulated as modeling hidden states trajectories, i.e. parameterizing the hidden state dynamics.

First, eq. (10) is solved forward in time with any solver, for instance Runge-Kutta. Then, relying on the adjoint method (Pontryagin, 1987), i.e. a method to compute gradients, the parameters can be optimized in two ways: optimize-discretize (opt-disc), the original approach proposed in R. T. Q. Chen et al. (2018), or discretize-optimize (disc-opt). With the opt-disc method, the gradients are computed analytically. Although this has the advantage of not requiring a differentiable solver and allows an efficient use of memory, it can lead to numerical errors in the evaluation of the gradients, when discretizing backward (Gholami, Keutzer, and Biros, 2019). Using the disc-opt method avoids the above problem: it computes the gradients based on the explicit discretization of the ODE (Gholami, Keutzer, and Biros, 2019; Onken and Ruthotto, 2020). This can be done using automatic differentiation and amounts to backpropagation. However, as backpropagation is done through the ODE solver, this one should be differentiable.

Whereas the new paradigm of Neural ODE is memory-efficient and allows us to learn models that are as close as continuous-time as possible, it is prone to numerical errors inherent to ODE solvers (Zhuang et al., 2020). Note that Neural-ODE has been used not only for modeling temporal data, but also for static data, such as in image classification and generation (R. T. Q. Chen et al., 2018; Dupont, Doucet, and Teh, 2019). In this case, time t is a continuous abstraction

of the depth of a neural network: inputs are given at time $t = 0$ and outputs are returned at some time $t = T$.

NEURAL NETWORKS MOTIVATED BY NUMERICAL ANALYSIS R. T. Q. Chen et al. (2018), He et al. (2016), and Weinan (2017) paved the way to the exploration of the continuum between mechanistic and ML models. From the established links between ODE/PDE and neural networks, a flourishing literature has thrived, ranging from the development of stable architectures for deep neural networks (Haber and Ruthotto, 2017) to the reinterpretation of networks such as ResNet, PolyNet (X. Zhang et al., 2017), FractalNet (Larsson, Maire, and Shakhnarovich, 2016) and RevNet (Gomez et al., 2017) as different numerical discretizations of differential equations (Lu et al., 2018). Such considerations finally led to the neural implementation of sophisticated numerical schemes (Ruthotto and Haber, 2020).
Even though our work derives from the neural representation of ODE, it is not part of this research theme and the interpretation of neural networks training with numerical analysis in the prism of classical ODE resolution is out the field of this thesis. However, we will use the networks presented above for their connection with the physical world and their interpretation in dynamical terms. We use such insights to model physical phenomena with neural networks.

While the approach just developed is prior-free and relies only on ML, another approach includes the form of the differential equation as prior knowledge. The introduction of physical knowledge into models has indeed recently become a hot topic.

Solving differential equations with neural networks

Solving differential equations with neural networks is a long-standing task (Lagaris, Likas, and Fotiadis, 1998; H. Lee and Kang, 1990) that has recently been reintroduced with Physics-Informed Neural Networks (PINNs) (Raissi, 2018; Raissi, Perdikaris, and G. E. Karniadakis, 2019). This approach assumes that the PDE is known, i.e. the dynamics f as well as initial or border conditions in eq. (5) are known. The solution is parameterized with a neural network y_θ, i.e. y_θ is learned to approximate y obeying to eq. (5). In fact, this approach builds a surrogate model to solve the PDE and replace traditional numerical simulations. Such methods present the advantage to be mesh-free approaches. However, some drawbacks come from their difficulty to represent high spatial and temporal frequencies and to be grid-dependent, so that they cannot generalize over new geometry. To overcome the first issue, Sitzmann et al. (2020) uses activation functions to recover higher-order derivatives and thus high-frequency information. Brandstetter, Worrall, and Welling (2022) proposes a solver based on graph neural networks, which enables to generalize to new sampling.

Note the shift of paradigm with section 2.1.1: whereas Ayed et al. (2020) and Yin, Le Guen, et al. (2021) use Resnet and Neural-ODE to learn the system trajectories, Raissi, Perdikaris, and G. E. Karniadakis (2019) consider f known and take a physicist standpoint to solve the ODE. Whereas Physics-informed machine learning has been studied in specific cases for weather and climate modelling (Kashinath et al., 2021), we rely on the approach developed previously i.e. we do not focus on approximating the solution to eq. (4) but rather learn its dynamics f.

2.1.2 *Physics-guided Deep Learning and Hybrid Modeling*

While Neural ODE and ResNet provide an interesting learning setting, the learned function remains nonetheless difficult to interpret. Indeed, recent works show that a partially observed physical system can be subject to accurate prediction by a ResNet-like integration method but the learned hidden state violates physical principles (Ayed et al., 2020). To palliate such observations, one can enforce physical properties by considering additional penalty besides the sole prediction loss. Since the two pioneering works Bezenac, Pajot, and Gallinari (2019) and Karpatne et al. (2017) constraining neural networks to be consistent with physics using prior physical knowledge for a prediction task, approaches to integrate physical knowledge have become commonplace. These are thoroughly reviewed in Willard et al. (2020).

To integrate physical principles into ML models, various methods exist. Increased interpretability can be obtained via physics-guided loss functions. For instance, Z. Long, Lu, and Bi Dong (2019) and Z. Long, Lu, Ma, et al. (2018a) impose constraints on the forward model by enforcing convolutional filters to approximate euclidean differential operators while Geneva and Zabaras (2020) impose constraints on learned residual.
Physical knowledge may also be incorporated within the design of architecture. Recently, some works leveraged data specific knowledge to shape the prediction function, for example imposing specific fluid dynamic (Raissi, Babaee, and Givi, 2019) or Hamiltonian constraints (Toth et al., 2020). Greydanus, Dzamba, and Yosinski (2019) proposed structural constraints on NN forcing them to obey the Hamiltonian formalism and integration schemes, thus learning to simulate mechanical systems obeying a conservative principle such as the coordinate and momentum variables of an ideal pendulum from pixel observations. This approach to regularize NN has been generalized to Lagrangian motions (Cranmer et al., 2020).

The originality of our approach is to leverage model-based prior knowledge by augmenting it with NN-parameterized dynamics, which belongs to the broader class of hybrid ML/MB models (where MB stands for physical model-based).

Hybrid Modeling

Grey-box or hybrid modeling was first explored in the 1990's(Psichogios and Ungar, 1992; Rico-Martinez, J. Anderson, and Kevrekidis, 1994; Thompson and Kramer, 1994). Such models take into account strong inductive biases to model the evolution of a system state over time, combining physical models relying on ODE/PDE (hereafter referred to as the MB part) and data based models (hereafter referred to as the ML part). Hybrid modeling rely on two tasks: identification of the parameters of the physics-based part, and prediction of the system state. In the last few years, they have received a regain of interest in the machine learning community. Linial et al. (2021), Saemundsson et al. (2020), and Tait and Damoulas (2020) use variational encoding (Kingma and Welling, 2013) to sample the space of initial conditions and parameters to solve both identification and prediction. Mehta et al. (2020) use Neural ODE combined with prior knowledge in the forward model and fully observed state to recover unknown parameters.

Hybrid approaches offer several benefits. They allow for alleviated computational costs for fluid simulation (Kochkov et al., 2021; Tompson et al., 2017; Wandel, Weinmann, and R. Klein, 2021). They can also be used to complete a physical model: Y. Long, She, and Mukhopadhyay (2018) and Saha, Dash, and Mukhopadhyay (2020) both use data-driven networks to learn additive perturbations to a given PDE. Mehta et al. (2020), Saha, Dash, and Mukhopadhyay (2020), San

and Maulik (2018), and Young, Liu, and M.-C. Wu (2017) also study the learning of a physical model augmented with a statistical component.

Since inverse problems are inherently ill-posed (Sabatier, 2000), the imposition of prior knowledge or regularisation is necessary to ensure sound parameter identification (Stewart and Ermon, 2017). For example, Bezenac, Pajot, and Gallinari (2019) estimates a velocity field from data under the constraint of an advection-diffusion PDE to derive a forecast scheme. However, most of the above approaches do not address either the uniqueness of the decomposition between MB and ML or the plausibility of the parameter identification. The challenge of proper MB/ML cooperation has been raised as a limitation of grey-box approaches, but has only recently been specifically addressed. For instance, to ensure uniqueness of the solution, Ardizzone et al. (2019) use invertible neural networks.

APHYNITY In our work, we take from the formalism introduced in Yin, Le Guen, et al. (2021), which specifically addresses the uniqueness of the MB/ML decomposition. Yin, Le Guen, et al. (2021) propose to decompose the dynamics into two components, adding both ML and MB models, such that

$$f = f_p + f_d$$

where f is the overall unknown dynamics, f_p is the physical MB dynamics and f_d accounts for the data-driven component. In this context, the Aphynity framework enables physically plausible estimates of the unobserved states, opening the way to well-built combinations between neural estimation and numerical scheme of partial differential equations. To ensure that f_d only plays a complementary role, i.e. that it models only the information that cannot be captured by f_p, Yin, Le Guen, et al. (2021) proposes the following optimization problem:

$$\min_{f_p \in \mathcal{F}_p, f_d \in \mathcal{F}_d} \|f_d\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \quad \forall t, \quad \frac{\mathrm{d}X_t}{\mathrm{d}t} = (f_p + f_d)(X_t).$$

Yin, Le Guen, et al. (2021) show that if $\mathcal{F}_p$ is a proximinal set, there exists a unique decomposition minimizing such optimization problem, as well as the existence and uniqueness of the decomposition for looser conditions.

In our work, we make two assumptions: 1. the MB component, i.e. the PDE only provides partial incomplete information about the physical process, 2. the observations are partial meaning that we do not observe the whole state of the system. This is a realistic setting for many applications invloving physical priors and ML. Up to our knowledge, this setting has not been handled yet. Note that most existing work consider that the whole system state is observed (with the excpetion of Ayed et al. (2020)). We generalize latter approaches and, as (Yin, Le Guen, et al., 2021), we address the well-posedness in the learning of hybrid ML/MB models through additional regularization on the estimated parameters of the physical part. Finally, contrary to most papers listed in this section, we test our proposed models on complex data, coming from ocean modeling and introduced in section 2.2.

2.1.3 *Machine Learning for Complex Data*

The last few years have seen an exponentially increasing number of deep learning applications to geophysics through the use of earth observation data (Bergen et al., 2019). They raise new concerns, such as the high dimensionality of the data, the complexity of the phenomena...Most papers experiment on so-called toy dataset, or on low-dimensional phenomena. Even though this

appears as a necessary step in the confrontation between data-driven models and the physical world, it now seems important to reach the next step, and to face real-world problems.

ML AND SCIENCE R. Stevens et al. (2020) give an overview of the scientific fields where ML could be of help and summarizes an AI roadmap for the coming decade, proposing opportunities for accelerating progress. The challenges concern various fields, from chemistry and materials to biology and life sciences, from engineering and manufacturing to earth and environmental sciences. Some works explore the direct application of off-the-shelf models to real-world data. For instance, while Jia, Willard, Karpatne, Read, Zwart, Michael Steinbach, et al. (2019) consider lake temperature, PINNs (Raissi, Perdikaris, and G. E. Karniadakis, 2019) have been investigated to simulate incompressible laminar and turbulent flows (Cai et al., 2022; Jin et al., 2021). These studies could pave the way for a massive reduction in the costs of computational fluid dynamics (Kochkov et al., 2021), widely used in weather and climate modeling.

ML FOR WEATHER AND CLIMATE MODELING The difference between weather and climate models lies in the scale and complexity of prediction: while weather prediction focuses on specific areas and short periods of time, climate models are broader and aim to predict over longer periods of time. Both rely on data assimilation, and researchers are now asking whether ML techniques could help with such models. As ML emerges as a viable alternative to physical models in weather forecasting, many papers have asked whether artificial intelligence methods will eventually replace current numerical weather models and data assimilation systems (Chantry et al., 2021; Schultz et al., 2021; Watson-Parris, 2021). Today, this challenge seems to be met: purely ML methods Bi et al. (2022) and Lam et al. (2022) outperform state-of-the-art numerical weather prediction (Barros et al., 1995) for the first time. For climate modeling, Monteleoni et al. (2013) provide a proof of concept for the use of machine learning for climate science, with the expectation that it will greatly accelerate discoveries in this area of research. For example, ML could help with climate change attribution (A. Ganguly et al., 2014). The exploitation of massive climate data could also lead to significant advances in the prediction of climate extremes. Reliable projections at shorter, more localised time scales would be more easily exploited by policy makers and thus advocate for adaptation and action. In this regard, downscaling (Groenke, Madaus, and Monteleoni, 2020) and subgrid parameterization (Bolton and Zanna, 2019; Frezat et al., 2022) are commonly tackled problems in climate modeling with ML. Ling, Kurzawski, and Templeton (2016) studies the modeling of turbulent processes, which also has great applications in ocean processes in climate models (Zanna and Bolton, 2021).

One major challenge for ML is the tricky issue of generalization, i.e. could neural networks infer system behavior in regions of the phase space not included in the training dataset.

DEALING WITH GENERALIZATION When handling real-world data, one main problem is overfitting. Indeed, data-driven approaches to modeling physical systems fail to generalize to unseen systems that share the same general dynamics, but correspond to different physical contexts (Kirchmeyer et al., 2022). In other words, deep neural networks are usually trained with closed-world assumption: the test data distribution is assumed to be similar to the training data distribution. However, when employed in real-world tasks, this assumption does not hold. For instance, the model could face new initial conditions or new parameters in the leading ODE. To generalize, the model should both interpolate, e.g. generalize to new initial conditions, and

extrapolate, e.g. generalize to new ODE parameters (what we hereafter refer to as new zones). Whereas ODE are not data specific, a major challenge in ML applied to real-world data is thus the generalization to Out-of-Distribution (OOD) data: in order to make models trained on synthetic datasets robust enough to function in real world applications it is necessary to ensure that they capture the variability of real world data.

In that perspective, meta-learning methods have recently been considered for dynamical systems as in (Finn, Abbeel, and Levine, 2017; S. Lee, Yang, and Seong, 2021). Their objective is to train a single model that can be quickly adapted to a novel environment with a few data-points in limited training steps. Kirchmeyer et al. (2022) and Yin, Ayed, et al. (2021) also consider the generalization across environments when modeling dynamical systems from real-world data samples. To be able to adapt to new dynamics, they take from multi task learning (C. Zhang et al., 2021) by learning to condition the learned model on domain information. Such techniques are outside the scope of our research, but we will draw on them in part iii.

2.2 OCEAN SURFACE DYNAMICS DATA

Throughout this thesis, we work on data representative of the ocean surface dynamics. In this section, we first introduce some basic notions for the description of such dynamics (sections 2.2.1 and 2.2.2). We then introduce the data we have been working with. As a first step, we initially focus on synthetic data, to establish a methodology and derive theoretical guarantees for the proposed models. We then confront our models to an ocean general circulation model (OGCM) of the dynamics in the North Atlantic Ocean (NATL60). We present those two datasets respectively in sections 2.2.3 and 2.2.4.

2.2.1 *Ocean variables*

Oceanography aims at understanding ocean processes. To that end, one studies the evolution of ocean physical properties, such as its temperature $\boldsymbol{T}$, its salinity $\boldsymbol{S}$, its density ρ, the chlorophyll concentration, the velocity $\boldsymbol{U} = (u, v, w)$ (Chassignet, Le Sommer, and Wallcraft, 2019). Such fields are four-dimensional, i.e. they evolve through time t and tri-dimensional space x, y, z. To access a discrete sub-sampling of such quantities, past decades have seen an increase in ocean observation systems, either from in-situ measurements, e.g. thanks to floats or buoys (Roemmich et al., 2009), or through remote sensing (Esaias et al., 1998). Nowadays, $\boldsymbol{T}$, $\boldsymbol{S}$ and the chlorophyll concentration can be observed with satellites, at a 1km $\times$ 1day resolution at the sea surface. Note that the velocity horizontal components u and v are not directly accessible. They are instead inferred from remote observations of the sea surface height (SSH), at a 50km $\times$ 1week resolution. Despite great advances, observational systems provide limited and incomplete information. Especially for the deep ocean, i.e. below a water depth of over 200m, only scarce in situ observations are available (Levin et al., 2019).

2.2.2 *Ocean equations*

The ocean exchanges fluxes of heat, fresh water, salt, and momentum, for example through wind stress in the case of atmosphere-ocean interface, with the solid earth, the continental margins, the sea ice and the atmosphere. Such processes can be described to a good approximation by the primitive equations, i.e. a set of nonlinear partial differential equations. Under the assumptions

described in Madec et al. (2017), those include the momentum balance, the hydrostatic equilibrium, the incompressibility equation, the heat and salt conservation equations. In this order, they write as:

$$\frac{\partial \boldsymbol{U}}{\partial t} + (\boldsymbol{U} \cdot \nabla)\boldsymbol{U} + g'\nabla h = \gamma \wedge \boldsymbol{U} + D^{\boldsymbol{U}} + F^{\boldsymbol{U}} \tag{11a}$$

$$\frac{\partial p}{\partial z} + \rho g = 0$$

$$\nabla \cdot \boldsymbol{U} = 0$$

$$\frac{\partial \boldsymbol{T}}{\partial t} = -\nabla \cdot (\boldsymbol{T}\boldsymbol{U}) + D^{\boldsymbol{T}} + F^{\boldsymbol{T}} \tag{11b}$$

$$\frac{\partial \boldsymbol{S}}{\partial t} = -\nabla \cdot (\boldsymbol{S}\boldsymbol{U}) + D^{\boldsymbol{S}} + F^{\boldsymbol{S}}$$

where γ is the Coriolis parameter, h the depth of the surface layer obtained from sea surface height (SSH) observations, g' the reduced gravity which takes the stratification in density of the ocean into account such that $g' \approx g.10^{-3}$, p is the pressure, ρ is the density, $D^{\boldsymbol{T}/\boldsymbol{U}/\boldsymbol{S}}$ refer to small scale processes such as diffusion and $F^{\boldsymbol{T}/\boldsymbol{U}/\boldsymbol{S}}$ balance the surface forcings, i.e. the exchanges at the surface of kinetic energy, heat and salinity. In all equations, the scalar product of the velocity U and the ∇ operator represents the advection phenomenon: for instance, $\nabla \cdot (\boldsymbol{T}\boldsymbol{U})$ depicts the advection of $\boldsymbol{T}$ by $\boldsymbol{U}$. $\gamma \wedge \boldsymbol{U}$ is the cross product of γ and $\boldsymbol{U}$, depicting for the Coriolis effect.

The ocean circulation, that is the current velocity fields dynamics, are now realistically modeled in tri-dimensional structured models such as NEMO (Madec et al., 2017), relying on numerical solutions to the above primitive equations. Within this study, we work on data from such models. This frees us from considerations inherent to observations, and limitations such as the cloud cover while providing us a realistic setting. Besides, we work on surface data, i.e. we only consider the two-dimensional surface of the ocean generated by (x, y), hereafter denoted T, $U = (u, v)$ and F for respectively the temperature, the velocity and the forcings. We consider as variables of interest the temperature T and the surface currents velocity fields $U = (u, v)$, and thus only dynamics on T and U from eqs. (11a) and (11b) i.e. we consider the advection of the temperature by the surface currents velocity fields. In particular, we do not represent the vertical velocities, responsible for the movements between the ocean surface and the lower strata. In a two-dimensional setting, $\nabla \cdot (TU)$ refers to the advection of a scalar quantity T by a velocity field $U = (u, v)$ and writes as : $\nabla \cdot (TU) = \frac{\partial T}{\partial x} u + \frac{\partial T}{\partial y} v$. We first work on a simplified representation of ocean dynamics, relying on simplifying assumptions, which we review in the following.

2.2.3 *Synthetic dataset*

For slow movements (that is of characteristic time superior to a day and of spatial dimension superior to 20km) incompressibility is assumed, i.e the turbulent terms are null: $(U \cdot \nabla)U = 0$. Besides, diffusion is omitted. Whereas T is observed by satellites, U is not known. However, the Sea Surface Height (SSH) could be used to compute coarse estimates of U. Indeed, under hypotheses such as stationarity ($\frac{\partial U}{\partial t} = 0$), incompressibility ($(U \cdot \nabla)U = 0$), forcings can be omited. In this case, eq. (11a) can be rewritten into:

$$\gamma \wedge U = g'\nabla h \tag{12}$$

In this case, the SSH h can be regarded as a stream function i.e. a function whose derivatives express the velocity components. When projected onto x and y axis, eq. (12) becomes

$$\gamma v = g' \frac{\partial h}{\partial x}, \qquad \gamma u = -g' \frac{\partial h}{\partial y} \tag{13}$$

Note that eq. (12) and eq. (13) do not hold at fine scales as the stationarity and incompressibility assumptions only hold at large scale. We first investigate a dataset generated following simplifying assumptions, which we refer to as Adv+F (for advection + forcing). We don't rely on the true U and F, we instead build them. Their computation is described below. We generate data following the tracer equation inspired by eq. (11b) (where F accounts for F^T):

$$\frac{\partial T}{\partial t} = -\nabla \cdot (TU) + F \tag{14}$$

where U and F computations are derived hereafter. Note that transport equations describe a wide range of physical phenomena such as chemical concentration, fluid dynamics or material properties. In this thesis, we interpret eq. (14) as the evolution of the temperature T advected by a time-dependent velocity field U and subject to forcing F.

BUILDING A VELOCITY FIELD U To simulate a transport setting represented by eq. (14), we first build a velocity field U. Under stationarity and incompressibility hypotheses, U can be approximated from a stream function $\mathcal{H}$. Note that, in this dataset, $\mathcal{H}$ is not equal to the SSH h, it is simulated following Boffetta et al. (2001):

$$\mathcal{H}(x, y, t) = -\tanh\left(\frac{y - B(t) \times \cos kx}{\sqrt{1 + k^2 B(t)^2 \times sin^2 kx}}\right) + cy \tag{15}$$

where B varies periodically with time according to $B = B_0 + \epsilon \cos(\omega t + \phi)$. We compute 10 different velocity fields sampling random parameters $B_0, k, c, \omega, \epsilon, \phi$. As introduced precedently (see eq. (13)), eq. (11a) can be simplified and we compute $U = (u, v)$ so that it follows:

$$u = -\frac{\partial \mathcal{H}}{\partial y}, \quad v = \frac{\partial \mathcal{H}}{\partial x} \tag{16}$$

BUILDING A SOURCE TERM F In eq. (11b), the diffusion term D^T is omitted. We generate the forcing term F so that it represents the forcing term F^T in eq. (11b). To illustrate heat exchanges, we draw from Frankignoul (1985). This source term is a non linear transformation of $U = (u, v)$ multiplied by the difference between the ocean temperature and a reference temperature:

$$F(U, T) = w_e \times (T - T_e) \quad \text{where} \quad w_e = \begin{cases} 0 & \text{if } \dfrac{\partial \mathcal{H}}{\partial t} < 10^{-4} \\ 1 & \text{otherwise.} \end{cases}$$

where T_e is the sequence mean image (computed without forcing).

DATASET GENERATION Using generated U and F, we integrate eq. (14) with a timestep $\Delta t = 8640s$ (about 2 hours) over 30 days. We integrate with a Semi-Lagrangian scheme, as it is mainly used to describe physical systems with advective behaviour, such as fluid flows. Details on the semi-Lagrangian advection can be found in appendix A.1. The semi-Lagrangian is implemented

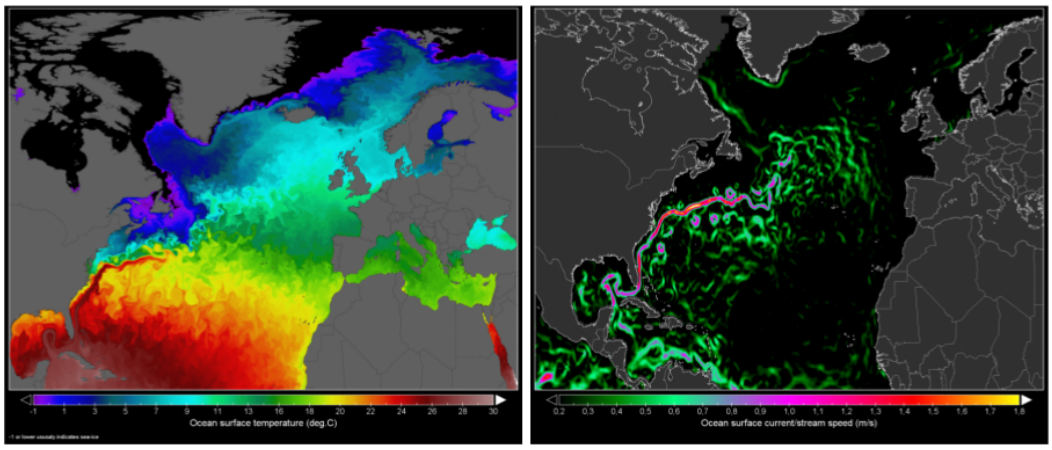

Figure 7: Sea Surface Temperature (on the left) and surface current velocity (on the right) data in the North Atlantic. Mercator Ocean analysis 23/04/2019 (for the SST T) and 22/04/2019 (for the surface currents U) provided by Andrej Flis.

using Pytorch function `gridsample`, following Jaderberg et al. (2015). To generate our data, we sample randomly 800 images of size 64×64 in NATL60 dataset (see section 2.2.4), each image representing an initial condition. For integration, we impose as border conditions East-West periodic conditions, implying that what comes out the left part re-enters at the right, and reciprocally. We impose velocity to be null on both top and bottom parts of the image. Among these 800 images, 80 are reserved for validation. 200 other images are sampled for test. These data are used in part ii.

2.2.4 *OGCM realistic data*

Having worked on ideal data, we want to test the developed models on data closer to reality. To investigate the ability of deep learning to reconstruct fine-scale surface current velocity fields from SST, we use the data from the NATL60 simulation, based on the NEMO code (Ajayi et al., 2020). This is the first kilometer-scale ($1/60°$resolution) simulation of ocean circulation in the North Atlantic to take into account the complexity of the coasts and submarine landforms as well as the large variability of surface atmospheric conditions. Figure 7 gives an example of SST and surface currents in the North Atlantic Ocean. The data were provided by MEOM research team, from the IGE laboratory from the Université Grenoble Alpes.

DATASET GENERATION From the simulation, we select a 1096km $\times$ 1352km zone over one year, representative of the Gulf Stream off the coast of Canada. We use the SST data, as well as the zonal (west–east direction) and meridional (north–south direction) components of the current

velocity. We split each image into 320 patches, i.e. regions, of size 64×64. We thus have 320 sequences of length one year. To select the test set, we have two choices: 1. to use zones never seen during training, i.e. to test the ability of our model to extrapolate spatially, 2. to use time sequences never seen during training, but belonging to training areas, i.e. to test the ability of our model to extrapolate in the time domain. In this thesis, we focus on the time extrapolation capacity of our model, and select our test set accordingly. Over one sequence of length one year, i.e. 365 images, we keep 75 for test and validation, divided into 5 sequences of length 15 days. Those are regularly sampled over the year, so that all seasons are seen during training, and tested. The spatial generalizability is out the scope of our study. Note that those data are only used in part iii, in which we explore the possibility to model real-like ocean dynamics with ML.

2.3 OBJECTIVES OF THE THESIS

We now give an overview of the core hypotheses used through this thesis. Recall that we are only interested in surface modeling, so we are in a 2D spatial environment. We work on two datasets, both depicting flow dynamics: we refer to the SST, the velocity and the forcings. In section 2.3.1, we introduce the formalism of the notations. Then, in section 2.3.2, we present the assumptions made throughout the document, i.e. we take them into account in all the models we develop. In section 2.3.3, we introduce our objectives.

2.3.1 *Notations*

At a time t, we observe the SST $T_t \in \mathcal{T} \subseteq \mathbb{R}^p$. This temperature is influenced by the unobserved surface current velocity, i.e. a vector field, $U_t \in \mathcal{U} \subseteq \mathbb{R}^{2q}$. To ease notations, we refer to both observed and unobserved variables as $X_t = (T_t, U_t) \in \mathcal{T} \times \mathcal{U} \subseteq \mathbb{R}^{p+2q}$. X_t follows a dynamics f such that:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = f(X_t, t).$$

Every quantities, observed or to estimate, are regularly sampled on a spatiotemporal grid: at each timestep t, the time varying velocity field U_t writes as $U_t = (u_t, v_t)$ and u_t, v_t, T_t and the forcing term F_t are all of size 64×64 (i.e. $p = q$). We rewrite f as $f = (f_T, f_U)$ acting respectively on T and U:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} T_t \\ U_t \end{pmatrix} = \begin{pmatrix} f_T(X_t) \\ f_U(X_t) \end{pmatrix} \tag{17}$$

In this work, f_T and f_U from eq. (17) can be interpreted as follows: f_T represents the dynamics of the observed T and f_U represents the dynamics of the unobserved U. In the context of section 2.2.3 (synthetic data), $f_T(X) = -\nabla.(TU) + F$. Although f_U is not known, we show in appendix B.1 that U follows an ODE, i.e. that there exists f_U such that $\frac{\mathrm{d}U}{\mathrm{d}t} = f_U(U, t)$. In the context of section 2.2.4 (realistic NATL60 data), $f_T(X) = -\nabla.(TU) + D^T + F^T$ (eq. (11b)) and $f_U(U) = -(U.\nabla)U + \gamma \wedge U - g'\nabla h + D^U + F^U$ (eq. (11a)).

2.3.2 *Dynamical Hypothesis*

We study hybrid models by assuming available a partial knowledge of the dynamics of the observed T_t:

$$\frac{\mathrm{d}T_t}{\mathrm{d}t} = f_T(X_t) = f_T^p(X_t) + f_T^d(X_t) \tag{18}$$

where f_T^p amounts for the physics-based part of f_T. f_T^d amounts for the dynamics not encompassed by f_T^p. More formally, $f_T^p \in \mathcal{H}_p$ is a known operator with unknown parameters, and $f_T^d \in \mathcal{H}_d$ is the unknown residual dynamics. $\mathcal{H}_p$ and $\mathcal{H}_d$ denote function spaces.

OCEAN HYBRID MODELING In the case of eq. (11b), F^T accounts for surface forcing terms. D^T accounts for parameterizations of small-scale physics: it represents subgrid scale physics, i.e. important small-scale physical processes that occur at length-scales that cannot be adequately resolved on a computational mesh. For instance, turbulent motions are never explicitly solved, even partially. Instead, they are parameterized (Madec et al., 2017). Even though the estimation of F^T and D^T are most important for long-term simulations, there is still an incomplete understanding of subgrid scale parameterizations, and forcing fields are still poorly known (Chassignet, Le Sommer, and Wallcraft, 2019). Thus, we will only consider advection as part of the prior physical knowledge, and we will aim at learning both exchanges fluxes F^T and parameterizations D^T as residuals.
We can rewrite eq. (17):

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \begin{pmatrix} f_T^p(X_t) + f_T^d(X_t) \\ f_U(X_t) \end{pmatrix} \tag{19}$$

where, in the case of the synthetic dataset Adv+F from section 2.2.3,

$$\begin{aligned} f_T^p(T,U) &= -\nabla.(TU) \\ f_T^d &= D^T + F^T \\ f_U &\text{ is unknown.} \end{aligned}$$

and, in the case of NATL60 in section 2.2.4,

$$\begin{aligned} f_T^p(T,U) &= -\nabla.(TU) \\ f_T^d &= F \\ f_U(U) &= -(U.\nabla)U + \gamma \wedge U - g'\nabla h + D^U + F^U. \end{aligned}$$

Note that, with this assumptions, the parameters of f_T^p are in fact the velocity fields U. In this work, f_T is our target function and our objective will be to learn an estimate of f_T based on our available knowledge consisting in prior assumptions on f_T^p and observations. The training problem is described precisely below.

NOTE ON THE ADDITIVE HYPOTHESIS OF THE DECOMPOSITION Note that the additive hypothesis in eq. (18) is not restrictive. First, in the case of a metric space the decomposition as defined in eq. (18) always exists. Let us detail an intuition for the well-posedness of such decomposition. Let $\mathcal{H}_p$ be a closed convex subset of functions of an Hilbert space $(E, \langle , \rangle)$, and f the function we want to approximate with partial knowledge (represented by the space of hypothesis $\mathcal{H}_p$). Then, thanks to Hilbert projection lemma, we have the uniqueness of the minimizer of $\min_{g \in \mathcal{H}_p} \|f - g\|$,

i.e. there exists one unique $h_p \in \mathcal{H}_p$ such that: $\forall g \in \mathcal{H}_p, \|f - h_p\|_2 \leq \|f - g\|_2$. Finally, the additive decomposition hypothesis presents a remarkable advantage in the case of a vector space. Indeed, if $\mathcal{H}_p$ is a (closed) vector space, let $\mathcal{H}_p^\perp$ be its complementary in E, then the additive decomposition hypothesis allows us to assume the uniqueness in the decomposition: $f = f_{\mathcal{H}_p} + f_{\mathcal{H}_p^\perp}$, where $f_{\mathcal{H}_p^\perp} \in \mathcal{H}_p^\perp$ and $f_{\mathcal{H}_p} \in \mathcal{H}_p$. The existence and uniqueness coming directly from the additive decomposition hypothesis, this might explain why such assumption is common when bridging ML and physic-based hypothesis.

2.3.3 *Objectives*

LEARNING PROBLEM We aim at predicting trajectories of T, i.e. to model the evolution of the observable part following $\frac{\mathrm{d}T_t}{\mathrm{d}t} = f_T(X_t)$ with an hybrid model. We approximate f_T with a function $h_T \in \mathcal{H}$ learned from the observed data, where $\mathcal{H}$ is an hypothesis space.
Following eq. (18), we assume $h_T = h_T^p + h_T^d$ where $h_T^p \in \mathcal{H}_p$, i.e. the physical model belongs to the same hypothesis space as f_T^p: it has the same parametric form. Its parameters are denoted θ_p. We take as physical prior on the dynamics: $h_T^p(T, \theta_p) = -\nabla.(T\theta_p)$. h_T^p should thus capture the advection part in the dynamics f_T. We aim at learning parameters θ_p of h_T^p, that is the unobserved U. Free-form h_T^d aims at learning f_T^d, i.e. the forcing terms F. $h_T^d \in \mathcal{H}_d$ is represented by a free form functional with parameters θ_d, e.g. a neural network. Finally, the learning problem is to estimate from data the parameters of h_T^p so that they match the true physical ones and h_T^d to approximate at best the unknown dynamics f_T. Note that, in some parts of our work, we will also consider a function h_U, modeling the dynamics f_U of the velocity fields U.

INTUITIVE TRAINING OBJECTIVE In this regard, an intuitive training objective is to enforce $\frac{\mathrm{d}T_t}{\mathrm{d}t} = h_T(T_t)$, i.e. to minimize the distance between $h_T = h_T^p + h_T^d$ and f_T, such as:

$$\min_{h_T \in \mathcal{H}} \mathbb{E}_{s \sim p_S} \|h_T(s) - f_T(s)\|_2 \tag{20}$$

where p_S is the distribution of the state X that accounts for varying initial states. Each s defines a training sample.
Such an approach does not provide any physical guarantees on our model. Indeed, minimizing eq. (20) with $h_T = h_T^p + h_T^d$ enables to predict accurate trajectories but may have an infinite number of solutions. For instance, h_T^d may bypass the physical hypothesis h_T^p. Thus, interpretability is not guaranteed. Our goal is not only to predict accurate trajectories of T, but also to ensure that we learn physically meaningful decomposition $h_T = h_T^p + h_T^d$, i.e. to overcome ill-posedness.

We can refine our learning tasks into two specific objectives: system identification, i.e. estimating the parameters of the physical model (the currents velocity fields) from observations (the SST), and prediction, i.e. recovering the trajectories associated to the dynamics (of both the velocity and the SST). While solving both problems using model-based formulation admits well-known numerical solutions, for example using the adjoint method (Courtier, Thépaut, and Hollingsworth, 1994; Le Dimet and Talagrand, 1986), the combination of physical models and deep learning is still an open area of research. In this context, ML applications mainly focus on the prediction task, at the expense of the system identification. Indeed, Ayed et al. (2020) show that without any prior knowledge, the recovered estimates of a dynamical system states are not physically plausible despite accurate predictions. Moreover, as noted by Yin, Le Guen, et al. (2021), learning a linear MB/ML decomposition with the sole supervision on the system trajectories is ill-posed

and admits an infinite number of decompositions. Such observations highlight the need to incorporate physically motivated constraints in the learning of hybrid models, e.g. through regularization penalties. Several works already propose additional constraints to guide the model towards physical solutions (Jia, Willard, Karpatne, Read, Zwart, Michael S Steinbach, et al., 2019; Linial et al., 2021; Yin, Le Guen, et al., 2021).

In this thesis, we propose refinements of the objective eq. (20), using physical knowledge to derive new constraints. In chapter 3, we propose a regularization ensuring that we obtain physically consistent parameters U. In chapter 4, we go further and propose a framework for the well-posed learning of hybrid models. Those two contributions are theoretical studies and we present experiments in the ideal setting of section 2.2.3. In this case, $f_T^p = \nabla.(TU)$ where U are the velocity fields generated from eq. (16) and $f_T^d = F$ where F is computed from section 2.2.3. In chapter 5, we confront our models to the real data simulation Natl60 (section 2.2.4). In this case, $f_T^p = \nabla.(TU)$ where U are the velocity fields from Natl60 and $f_T^d = D^T + F^T$.

DISTANCE WITH FLOWS Note that, in practice, f_T is unknown and eq. (20) is not tractable. To train, we rely on the trajectories associated to the dynamics. We minimize the distance between the ODE flows ϕ_h and ϕ_f defined by h and f (the defintion of the flow can be found in section 2.1.1), $\delta(\phi_h, \phi_f)$, over all initial conditions X_0:

$$\delta(\phi_h, \phi_f) = \mathbb{E}_{X_0} \int_{t_0}^{t} \left\| \phi_h(\tau, X_0) - \phi_f(\tau, X_0) \right\|_2 \mathrm{d}\tau \tag{21}$$

We have $\delta(\phi_h, \phi_f) = 0 \implies \phi_h = \phi_f \implies h = f$. The gradients of $\delta(\phi_h, \phi_f)$ with respect to the parameters of h can be either estimated analytically using the adjoint method (R. T. Q. Chen et al., 2018) or using explicit solvers, e.g. Rk45, and computing the gradients thanks to backpropagation, see Onken and Ruthotto (2020).

To compute eq. (21), we rely on a temporal sampling of X: our datasets are composed of n sequences of observations of length N, $X^i = (X^i_{t_0}, \dots, X^i_{t_0 + N\Delta t})$, where each sequence X^i follows eq. (18) and corresponds to one initial condition $X^i_{t_0}$. We then sample the space of initial conditions $X^i_{t_0}$ to compute a Monte-Carlo approximation of $\delta(\phi_h, \phi_f)$. Let ODESolve be the function integrating any arbitrary initial state X_{t_0} up to time t with dynamics h, so that $X_t = \text{ODESolve}(X_{t_0}, h, t)$. The estimate of $\delta(\phi_h, \phi_f)$ then writes as:

$$\delta(\phi_h, \phi_f) \approx \frac{1}{nk} \sum_{i=1}^{n} \sum_{j=1}^{k} \left\| \text{ODESolve}(X^i_{t_0}, h, t_0 + j\Delta t) - X^i_{t_0 + j\Delta t} \right\|_2$$

In other words, we rely on the trajectories associated to the dynamics:

$$\text{ODESolve}(X^i_{t_0}, h, t_0 + j\Delta t) = X^i_{t_0} + \int_{t_0}^{t_0 + j\Delta t} h(X^i_t)\, \mathrm{d}t$$

Note that in practice, we don't learn the actual dynamics h_T but its numerical integration. Throughout the manuscript, we first introduce the learning objective in continuous form. We then explicit the practical loss $\mathcal{J}$ we optimize for training. Finally, for the sake of simplicity, we will refer to $\delta(\phi_h, \phi_f)$ as $\delta(h, f)$.

Part II

METHODOLOGY

This part is mostly theoretical and deals with the ill-posedness inherent in learning hybrid MB/ML models. We study the dynamics of the temperature h_T, assuming it is modelled by $h_T = h_T^p + h_T^d$. We take as physical prior on the dynamics: $h_T^p(T, \theta_p) = -\nabla.(T\theta_p)$. Our goal is to learn the parameters θ_p of h_T^p, i.e. the unobserved velocity fields U. More specifically, we are concerned with the physical plausibility of h_T^p.

In chapter 3 we propose to regularize the learning of the parameters θ_p of h_T^p. To do this, we rely on a dynamical hypothesis that assumes that these parameters are the solution of an ODE. We constrain them accordingly. However, this assumption does not solve ill-posedness.

In chapter 4 we go further and present a general framework that allows us to recover well-posedness when learning hybrid models. This is based on the optimisation of an upper bound on the prediction error. We also present an alternate optimization algorithm for which we provide a convergence analysis in a simplified case.

In both chapters we test our models on our synthetic dataset from section 2.2.3. We perform ablations to support our claims and compare our models to baselines. Both the data and the code will be made available.

3

DYNAMICAL REGULARIZATION FOR THE LEARNING OF HYBRID MODELS

In this chapter, we propose a first approach to overcome the ill-posedness of the learning problem developped in section 2.3. We propose to regularize the learning loss corresponding to the sole minimization of the difference between our model h_T and the real dynamics f_T given in eq. (20). To ensure that h_T^p is physically plausible, we propose to integrate prior knowledge into the learning process.
In our case, to identify the parameters θ_p of h_T^p amounts to solving an inverse problem (D. L. T. Anderson and Willebrand, 1989). Indeed, the SST is an ocean tracer, i.e. it can be used to track currents and deduce large-scale ocean circulation (Bigg and Killworth, 1988; England and Maier-Reimer, 2001). Thus, as in Ayed et al. (2020) and Bezenac, Pajot, and Gallinari (2019), we propose to learn θ_p from observations of T. However, having no observations of U, one can only rely on the prediction of future T for training. To further constrain learning, we propose to leverage prior dynamical knowledge by introducing a dynamical regularization on the unobserved states. Taking from the momentum balance eq. (11a), we posit that U is the solution to an ODE. Our proposition is thus to enforce U to follow a dynamics described by a differential equation. Note that this dynamics is unknown and we approximate it with a neural network, using their interpretation as numerical discretization of differential equations (He et al., 2016; Lu et al., 2018).
Finally, we shift from our initial problem, namely the estimation of U and the learning of the dynamics of solely T, to a new formulation: the learning of the dynamics of the full state $X = (T, U)$. We show that this proposition enables to estimate velocity fields that are physically plausible.

This chapter is organized as follows. In section 3.1, we propose a regularization term for the learning of our predictive model. The implementation of our model is exposed in section 3.2. We experiment on two datasets depicting simplified ocean dynamics, and give results and metrics in section 3.3. Finally, the supplementary material referenced in this chapter is available in appendix B.

3.1 MODEL

Our contributions lie along two intertwined research axes: 1. learning estimates of U; 2. learning the whole dynamics of X (both observed T and unobserved U) by incorporating partial physical knowledge. We give below a formulation of these two steps.

Recall some notations: we study the dynamics of the temperature h_T, assuming it is modelled by $h_T = h_T^p + h_T^d$. h_T^p is our physical prior on the dynamics: $h_T^p(T, \theta_p) = -\nabla.(T\theta_p)$. h_T^d is the residual part. In this chapter, we also consider the unknown dynamics h_U of the unobserved velocity fields U.

3.1.1 *State Estimation*

We aim at learning the dynamics of $X = (T, U)$. As we have no observations of U, we first need to estimate it. To that end, we use the observations of T. Indeed, T is a flow tracer, i.e. it can be used to deduce flow pattern, in our case that is the velocity U. Thus, as in Ayed et al. (2020) and Bezenac, Pajot, and Gallinari (2019), we propose to learn U from past observations of T. We formulate this problem as retrieving the full state X_t given a sequence of observed variables T up to t, i.e. we solve an inverse problem. To solve this problem, we learn a neural network G_θ with parameters θ estimating the unobserved U_t from the k precedent consecutive measurements $T_{t-k:t} = (T_{t-k+1}, \dots, T_t)$:

$$\begin{aligned} G_\theta : \quad \mathcal{T}^k &\to \mathcal{U} \\ T_{t-k:t} &\mapsto \hat{U}_t \end{aligned} \tag{22}$$

Having no observations of U, it is impossible to supervise the learning of G_θ. However, using h_T, we are able to weakly supervise U, based on discrepancy between $\hat{T}_{t+1}$ and the target image T_{t+1}. This is developped in section 3.2. Note in particular that we use G_θ to estimate the initial state $\hat{X}_{t_0} = (T_{t_0}, \hat{U}_{t_0})$, with $\hat{U}_{t_0} = G_\theta(T_{t_0-k:t_0})$.

3.1.2 *Dynamical Model*

There is usually no guarantee that $\hat{U}_t$ is coherent temporally nor physically interpretable (Ayed et al., 2020). Therefore, we propose to estimate its unknown dynamics f_U with a free-form function $h_U \in \mathcal{H}_U$, where $\mathcal{H}_U$ is a parametric space corresponding to a neural network. To regularize the estimation, we learn the dynamics of U using a specific PDE. More precisely, we enforce the unobserved U to obey an explicit PDE and make the trajectory of U well defined from an initial datum estimated thanks to G_θ. h_U is implemented with a ResNet, which can be viewed as approximating a transport equation (see for example Karkar et al. (2020) and Li and Z. Shi (2017)). Also, the method of characteristics provides existence and uniqueness of the solution to the Cauchy problem associated to the transport equation under mild assumptions.

3.1.3 *Learning Objective*

We want to accurately estimate the dynamics of the observed variable T, but also to model the intrinsic dynamics of the unobserved variable U. We have access to partial observations, i.e. to T, up to t_0 and want to forecast the full state from t_0 to the final timestep t_f. We consider the following objective:

$$\min_{G_\theta, h_T^d, h_U} \left\| \hat{U}_{t_f} - \left(\hat{U}_{t_0} + \int_{t_0}^{t_f} h_U(X_t)\, \mathrm{d}t \right) \right\|_2 \text{ subject to } \frac{\mathrm{d}T_t}{\mathrm{d}t} = (h_T^p + h_T^d)(\hat{X}_t) \tag{23}$$

Unfortunately, having no access to the true U we only rely on estimates given by G_θ. In order to solve eq. (23), we introduce two losses: we penalize the forecasts errors in the observed state, and force the unobserved variable U to obey a learned dynamics h_U. In the following, we present those losses and their implementation.

3.2 PRACTICAL OPTIMIZATION

In practice, having no priors on the dynamics f_T^d and f_U, we directly estimate their integrated counterpart h_T^d and h_U using neural networks. We rely on the associated flows ϕ and the integrator ODESolve to calculate the losses on the trajectories. The definition of the flow is to be found in section 2.1.1.

FORECASTING LOSS ON T For U to be estimated properly, it must lead to accurate predictions of T. Thus, we penalise the discrepancy between forecasts of T and their true value using a MSE loss:

$$\mathcal{J}_T(t) = \left\| \text{ODESolve}(\hat{X}_{t_0}, h_T, t) - T_t \right\|_2$$

where

$$\hat{U}_{t_0} = G_\theta(\hat{T}_{t_0-k:t_0})$$

and

$$\text{ODESolve}(\hat{X}_{t_0}, h_T, t) = \hat{X}_{t_0} + \int_{t_0}^{t} h_T(\hat{X}_{t_0}) \mathrm{d}\tau$$

EVOLUTION OF U Besides, we are interested in both the estimation and the dynamics of the unobserved U and constrain it to obey a partial differential equation defined by h_U:

$$\mathcal{J}_U(t) = \left\| \text{ODESolve}(G_\theta(\hat{T}_{t_0-k:t_0}), h_U, t) - G_\theta(\hat{T}_{t-k:t}) \right\|_2$$

where

$$\text{ODESolve}(G_\theta(\hat{T}_{t_0-k:t_0}), h_U, t) = G_\theta(\hat{T}_{t_0-k:t_0}) + \int_{t_0}^{t} h_U(G_\theta(\hat{T}_{t_0-k:t_0})) \mathrm{d}\tau$$

Note that for $t < t_0$, $\hat{T}_t$ refers to actual observations, while for $t \geq t_0, \hat{T}_t$ is the prediction done using former time steps estimations. In practice, t varies from $t_0 - k\Delta t$ to $t_0 + n\Delta t$, whree k and n are hyperparameters. In our experiments, Δt is equal to 8640s.

The optimization of eq. (23) consists in learning the parameters of (G_θ, h_U, h_T^d) by minimizing our overall cost function $\mathcal{J}$ defined by:

$$\mathcal{J} = \sum_{i=1}^{n} \sum_{j=0}^{k} (\mathcal{J}_U(t_0 + j\Delta t) + \lambda_T \mathcal{J}_T(t_0 + j\Delta t)) \tag{24}$$

$$\text{with } \mathcal{J}_T(t) = \left\| \text{ODESolve}(\hat{X}^i_{t_0}, h_T, t) - T^i_t \right\|_2$$

$$\mathcal{J}_U(t) = \left\| \text{ODESolve}(G_\theta(\hat{T}^i_{t_0-k:t_0}), h_U, t) - G_\theta(\hat{T}^i_{t-k:t}) \right\|_2$$

where n is the number of initial conditions in the training set and k is an hyperparameter. We learn jointly all parameters of G_θ, h_T^d and h_U.

To sum up, to predict X for $t \geq t_0$, we estimate both the initial state X_{t_0} and the dynamics of the full state, i.e. h_T and h_U. U_t is estimated with G_θ and used as input to h_T^p: to learn the parameters of h_T^p in fact amounts to learning G_θ. The computational graph from Figures 8 and 9 gives a schematic representation of our learning scheme over two time steps respectively at training time and inference time.

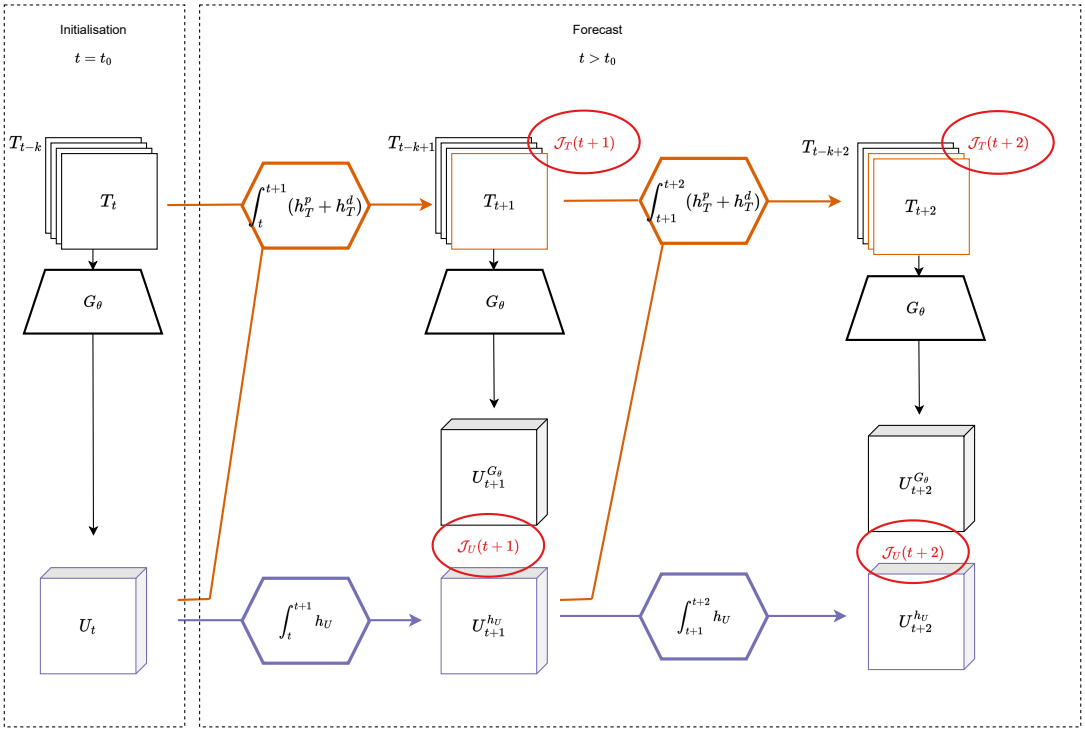

Figure 8: Computational graph of the proposed model on two time steps at training time. At forecast time, i.e. at time $t > t_0$, the evolution of $U_t^{h_U}$ as modeled with an ODE is constrained to coincide with $U_t^{G_\theta}$ as estimated with G_θ. Forecasting amounts to learning h_T and h_U from a sequence of observations of T. At each timestep t, we predict T and U at the next timestep $t+1$. The forcing term F are modeled with h_T^d and learned from a sequence of T. Note that $t-k$ actually refers to the timestep $t - k\Delta t$.

Finally, the learning of h_U enables to regularize the estimation of G_θ, ensuring the physical soundness of estimated velocity. Note that our model can be adapted to various data specific scheme such as fully Lagrangian (Bowman, Yassaei, and Basu, 2015) and more general all-purpose integrator such as `Rk4`, as long as differentiability is maintained while computing h_T^p. G_θ, h_U and h_T^d are modeled with neural networks.

3.3 EXPERIMENTS

In this section, we review the model implementation, giving relevant hyperparameters and architectures. We perform an ablation study: we compare our results to the one obtained without dynamical constraint, i.e. without h_U, and compare our results to baselines: NeuralODE (R. T. Q. Chen et al., 2018) and Aphyity (Yin, Le Guen, et al., 2021) which proposes to inform the forward model and to solve ill-posedness by minimizing the ℓ_2-norm of h_T^d, which we denote with $\|h_T^d\|$. Performances are evaluated via the standard MSE (lower is better). We report between brackets the standard deviation of the metrics over 5 runs.

We investigate two experimental settings: no source term, i.e. $F = 0$, and a non null source term F inspired by (Frankignoul, 1985) (see section 2.2.3). They are hereafter respectively refered

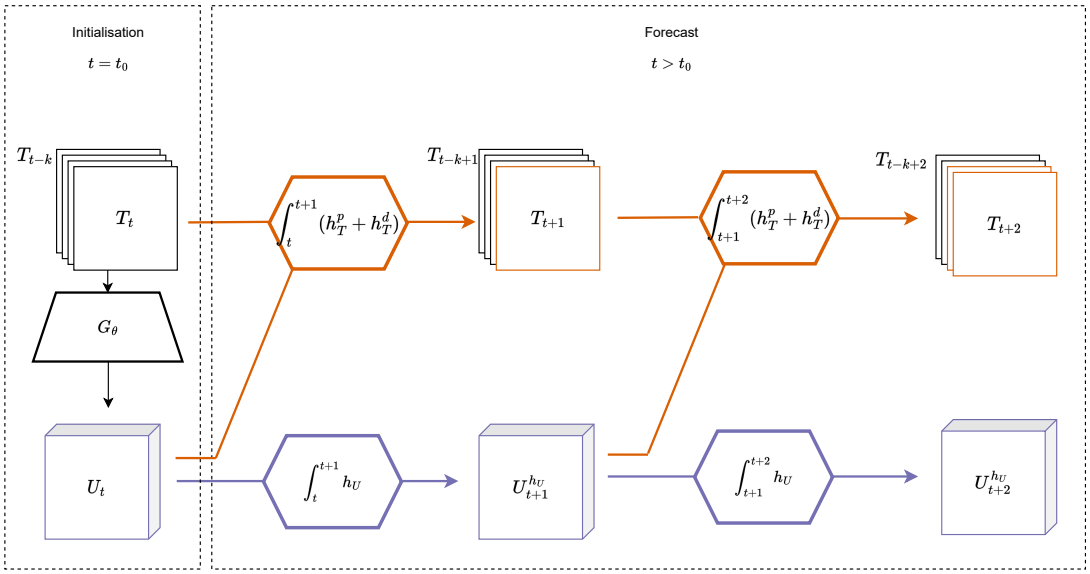

Figure 9: Computational graph of the proposed model on two time steps at inference time. Forecasting amounts to learning h_T and h_U from a sequence of observations of T. At each timestep t, we predict T and U at the next timestep $t+1$. Note that the inverse model G_θ is only used for the initialisation, i.e. at time $t = t_0$.

to as "Advection Only" and "Adv+F". In practice, we don't learn the dynamics h_T^d, h_U but their numerical integration.

3.3.1 *Training Details*

We used Python 3.8 and Pytorch 1.5 to implement our model trained on Nvidia GPU with CUDA 10.1.

ARCHITECTURES h_T^d is a U-net with at most 512 latent channels, following the implementation of (Isola et al., 2017a).
h_U is a convolutional Residual Network with 2 residual blocks. The input are first downscaled using two layers of strided convolutions. Each residual block has 128 channels, following the implementation of (Isola et al., 2017a). Relying on eq. (11a), we infer that the evolution of U is independent from the evolution of T, thus we make h_U takes as input only $\hat{U}$, previously estimated from a sequence of T. We consider h_U such that: $h_U(X_t) = h_U(U_t)$.
G_θ is a U-net with at most 512 latent channels also following the implementation of (Isola et al., 2017a). h_T^p implements a differentiable semi-Lagrangian scheme (Jaderberg et al., 2015)

HYPERPARAMETERS The learning rate for all algorithms and baselines is $lr = 10^{-4}$ using Adam optimizer with $\beta = (0.9, 0.999)$, with batch size 32. The number of input frames for G_θ and h_T^d is 4, i.e in eq. (22) $k = 4$. The number of predicted time steps T is 6. In practice we set $\lambda_T = 1$, and specify another multiplicative hyperparameter λ_U so that $\lambda_U = 0.01$

Table 1: Compared model results for the Advection Only and the Adv+F datasets. We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the forcing term F over 10 time steps on test set. The line Ours (U known) refers to the prediction of T and the estimations of F with the real velocity fields U. The line Ours ($\|h_T^d\|$) refers to our model with an additional loss term to constrain the norm of $\|h_T^d\|$, as is done in Yin, Le Guen, et al. (2021). The line Ours (no h_U) is an ablation: we train our model without the dynamical constraint; i.e. U is only constrained through the loss $\mathcal{J}_T$ without $\mathcal{J}_U$. n/a means not applicable.

Models	Advection Only			Adv+F		
	T	U	F	T	U	F
Ours (U known)	0.00	n/a	0.00	0.19	n/a	0.12
Ours	2.11(0.1)	4.90(0.92)	0.23(0.62)	4.97(0.41)	4.96(0.98)	0.89(0.71)
Ours ($\|h_T^d\|$)	2.04(0.19)	4.49(0.97)	0.07(0.6)	3.11(0.39)	10.10(1.64)	1.11(0.8)
Ours (no h_U)	0.95(0.1)	8.28(1.1)	1.07(0.69)	2.98(0.33)	15.71(2.1)	9.06(1.0)
Aphynity	1.07(0.29)	9.07(1.28)	1.09(0.8)	1.00(0.34)	11.74(2.82)	4.48(1.2)
NeuralODE	3.17(0.03)	n/a	n/a	5.24(0.07)	n/a	n/a

BASELINES For Neural ODE baseline, G_θ is a 3-layer convolutional networks. It is integrated using RK4 scheme available from https://github.com/rtqichen/torchdiffeq. For experiments with minimisation of $\|h_T^d\|$, a cost is added to the original cost function eq. (24): $\mathcal{J}_{h_T^d} = 0.01 \times \|h_T^t(T_{(t-k:t)})\|_2$. For Aphynity, we keep the same architectures for h_T^p, G_θ and h_T^d as described previously. We rely on the training described in Yin, Le Guen, et al. (2021), optimizing a cost $J = J_T + \lambda_{h_T^d}\|h_T^d\|$, with $\lambda_{h_T^d} = 0.0001$.

3.3.2 *Results*

INVERSE PROBLEM The inverse problem we aim to solve is the estimation of the velocity U from observations T. Figures 10 and 11 show examples of the estimated hidden states and the columns labeled U in Table 1 give the MSE between estimation and target hidden state.
Both Figure 10 and Table 1 show that G_θ truthfully estimates the hidden state using our framework. Our ablation study evidences that constraining U to follow an ODE with h_U indeed regularizes the learning: our model without h_U shows poor results on the inverse problem resolution. However, note that it performs well on predicting T. This evidences the ill-posedness inherent to hybrid modeling: poor estimation of U does not harm prediction but forces h_T^d to capture the whole dynamics. In other words, h_T^d compensates for the bad h_T^p. Unlike our estimations of F, our model without h_U thus also fails at estimating an interpretable forcing term F. Therefore, our dynamical prior on U helps solving the inverse problem and the ill-posedness inherent to hybrid modeling.

FORWARD PROBLEM The forward problem refers to the prediction of T using our hybrid model $h_T^d + h_T^p$. Examples of predictions of T are available in Figures 10 and 11 and columns labeled T in Table 1 give the MSE between prediction and target T. Figure 11 shows example of estimated forcing term F.
Constraining temporally the hidden states harms prediction accuracy despite truthful estimates in both T (Figures 10 and 11) and forcing term F. Constraining $\|h_T^d\|$ helps when $F = 0$ as it

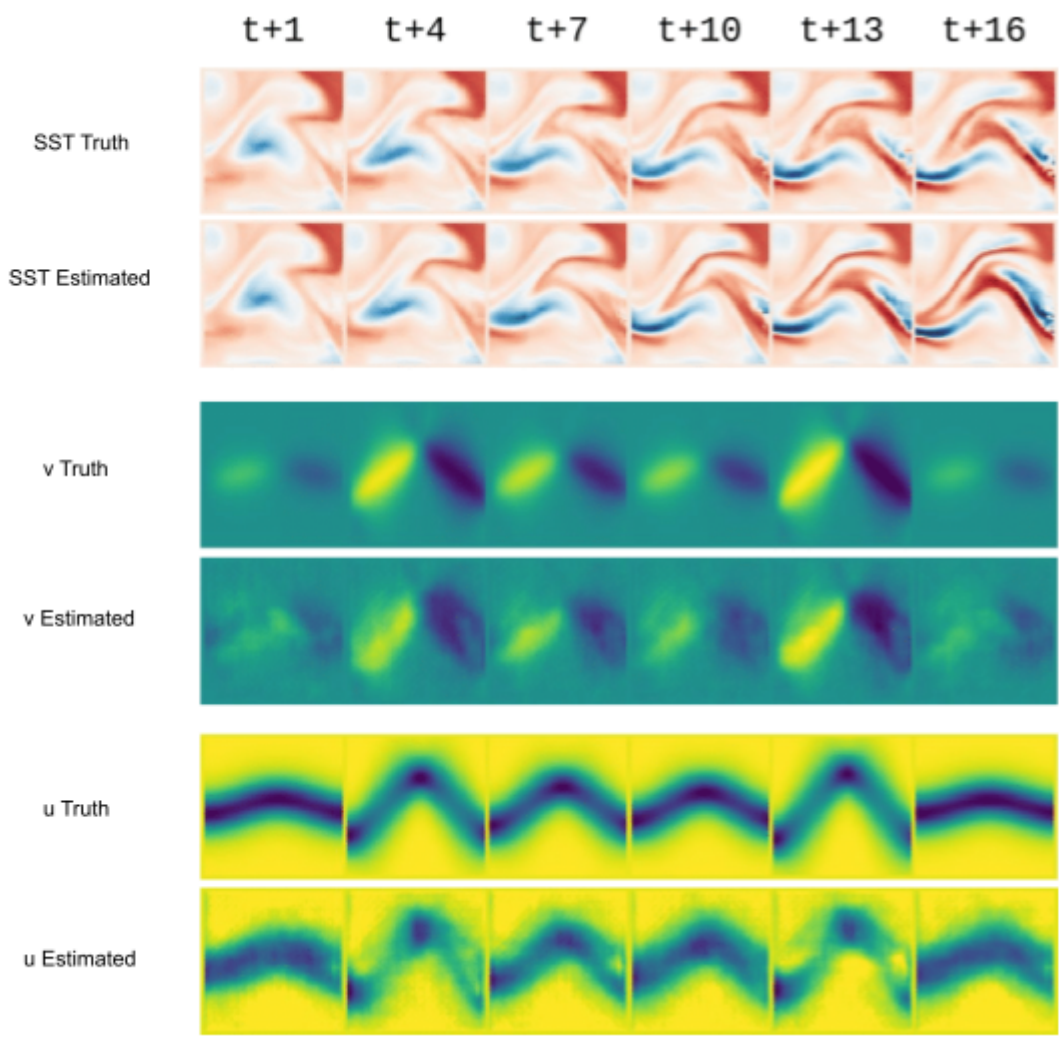

Figure 10: Sequences of estimations of T, $U = (u, v)$ for the dataset with non null physical forcing F, coming from the test set. For each variable T, u and v, the first line shows the ground truth, the second line the estimation with our model. Columns represent the time. Note that $t + 1$ actually means $t + \Delta t$ where $\Delta t = 8640$s. Without ever observing U, our model is able to estimate it from observations of T and make predictions over long time range.

adds plausible physical information. Thus, the estimates of T, U and F are better for Advection Only. However, the estimates obtained by constraining $\|h_T^d\|$ when F is not zero are better without constraining $\|h_T^d\|$. Adding a dynamical constraint on U through the learning of h_U gives thus more physical information than constraining $\|h_T^d\|$.
Finally, Aphynity is better at prediction, this can be explained as follows: even constrained, h_T^d might compensate for the dynamics not encompassed by h_T^p. Data agnostic algorithms such as NeuralODE are performing worse for long term forecasts than informed neural models, confirming that providing knowledge in a data-driven forward model brings stability in the forecasts.

3.4 CONCLUSION

In this chapter, we proposed to bridge PDE-specific numerical scheme with deep networks to solve forward and inverse problem for partially known dynamics. We empirically show that regularizing time varying unobserved states helps to solve both the forward problem and the inverse estimation. Besides, our framework applies to partial observations. In our example application, we are able to estimate the velocity U and the forcing F without ever observing them.

However, no theoretical considerations on the proposed regularization were investigated. This way, the learning of the decomposition $h_T = h_T^p + h_T^p$ is still ill-posed. For instance, $U_t = 0$ for all t is a solution of eq. (23). In the following chapter, we propose a learning framework ensuring the well-posedness in the learning of hybrid models.

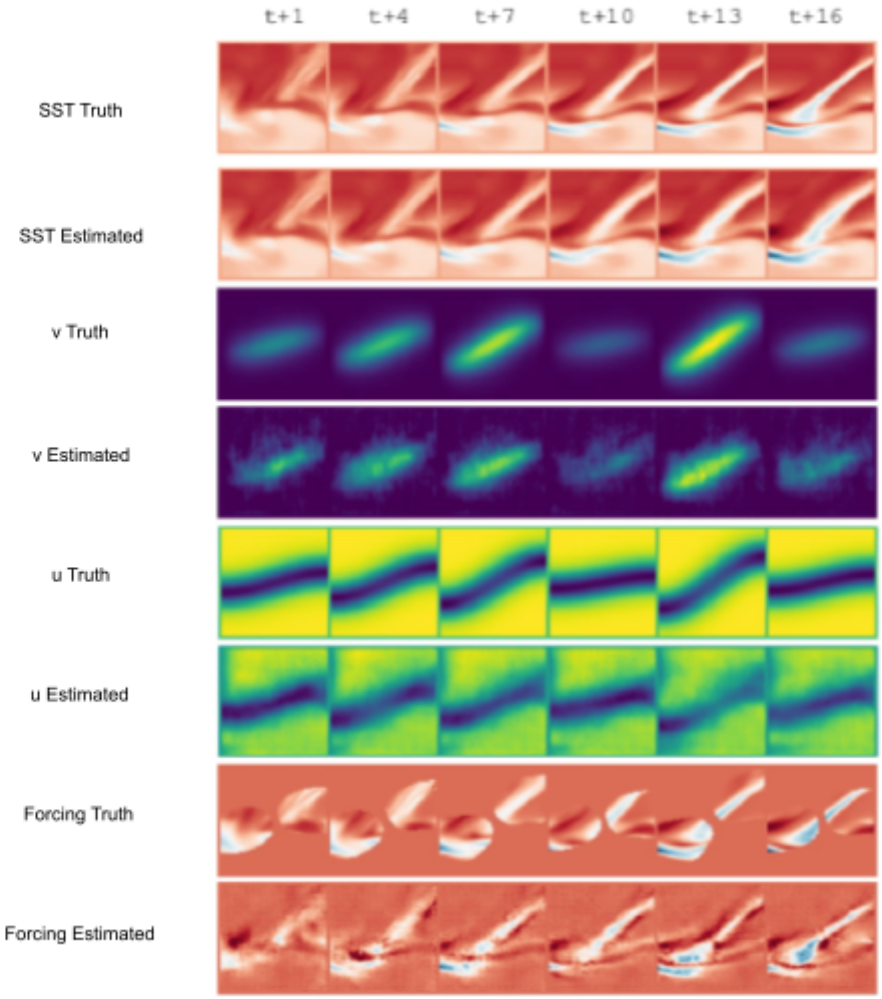

Figure 11: Sequences of estimations of T, $U = (u, v)$ and F for the dataset with non null physical forcing F, coming from the test set. For each variable T, u and v, F, the first line shows the ground truth, the second line the estimation with our model. Columns represent the time. Note that $t + 1$ actually means $t + \Delta t$ where $\Delta t = 8640$s. Without ever observing U or F, our model is able to estimate them from observations of T and make predictions over long periods of time.

4

CONSTRAINED PHYSICAL-STATISTICS MODELS FOR DYNAMICAL SYSTEM IDENTIFICATION AND PREDICTION

4.1 INTRODUCTION

In this chapter, not only are we interested in the learning of both physically sound velocity U and accurate trajectories of temperature T but also in solving the ill-posedness induced by the learning of hybrid ML/MB models introduced in chapter 2. In this sense, we go further than chapter 3. Besides, contrary to chapter 3, we do not make any assumptions neither on the velocity fields U nor on its associated dynamics f_U.

Recall that we consider the partially observed state $X_t = (T_t, U_t)$, where the temperature T is observed and the velocity fields U are unobserved. We aim to model the dynamics f_T of T with an hybrid model h_T. Our proposition is to reformulate the learning problem induced by eq. (20) in chapter 2, i.e.

$$\min_{h_T} \mathbb{E}_{s\sim p_S} \|h_T(s) - f_T(s)\|_2 \quad \text{with} \quad h_T = h_T^p + h_T^d$$

by introducing an upper bound on the prediction error of a physical-statistical model. This allows us to control the contribution of both the physical and statistical components to the overall prediction. In other words, we aim to recover well-posedness and interpretability of the decomposition: while h_T^p should account for the physical model, h_T^d should account for the residual not modeled by h_T^p. From this upper-bound, we work out a principled framework that generalizes previous attempts in the regularization of hybrid models. In particular, our proposition goes further than Yin, Le Guen, et al. (2021), which ensures the uniqueness in the decomposition by constraining the norm of the ML component. We also propose a novel alternate-optimization algorithm to learn hybrid models, for which we provide an analysis of the convergence on a simplified case. As our ultimate aim is to challenge real world problems, i.e. NATL60 data, in a second time we propose an extension of our framework to incorporate auxiliary data providing further physical evidence and get closer to complex real-world scenario. Finally, we emphasize that the method proposed in this chapter is very general and may be applied to many physical data. We propose experiments on various datasets which are not related to ocean dynamics in appendix C.2.
Unless otherwise indicated, within this chapter, we refer to the distance d between two functions h and f as defined in eq. (20) in section 2.3, i.e.

$$\mathrm{d}(h, f) = \mathbb{E}_{s\sim p_S} \|h(s) - f(s)\|_2$$

where s is a training sample. Let us recall that this distance is convex, as shown in appendix A.2.

This chapter is organized as follows. In section 4.2, we propose two upper-bounds and a well-posed learning framework for the learning of physical-statistical models. The implementation of our model is detailed in section 4.3. We experiment on our dataset depicting simplified ocean dynamics, and give results and metrics in section 4.4. The supplementary material referenced in this chapter is available in appendix C.

4.2 METHOD

Recall that we study hybrid models by assuming available a partial knowledge of the dynamcics of the observed T_t:

$$\frac{\mathrm{d}T_t}{\mathrm{d}t} = f_T(X_t) = f_T^p(X_t) + f_T^d(X_t)$$

where f_T^p amounts for the physics-based part of f_T. f_T^d amounts for the dynamics not encompassed by f_T^p. We approximate f_T with a function h_T learned from the observed data.
Following eq. (18), we assume $h_T = h_T^p + h_T^d$ where h_T^p belongs to the same hypothesis space as f_T^p: it has the same parametric form. In this context, we aim to enforce $\frac{\mathrm{d}T_t}{\mathrm{d}t} = h_T(T_t)$, i.e. to enforce h_T to represent the dynamics f_T of T. An intuitive objective could thus be to minimize the distance between $h_T = h_T^p + h_T^d$ and f_T, such as:

$$\min_{h_T \in \mathcal{H}} \mathbb{E}_{s \sim p_S} \| h_T(s) - f_T(s) \|_2 \tag{25}$$

In our hybrid modeling framework, two criteria are essentials: 1. identifiability, i.e. the estimated parameters of h_T^p should correspond to the true physical ones (in our case that is the velocity fields U); 2. prediction power, i.e. the statistical component h_T^d should complete h_T^p so that $h_T = h_T^p + h_T^d$ performs accurate prediction over the system states. To control the contribution of each term h_T^p and h_T^d, we work out upper bounds out of eq. (25) (section 4.2.1). We then propose to minimize $\mathrm{d}(h_T, f_T)$ while constraining the upper bounds, which provide us with a well-posed learning framework (section 4.2.2). Besides, we show that several previous works that introduced constrained optimization to solve related problems are specific cases of our formulation (Jia, Willard, Karpatne, Read, Zwart, Michael S Steinbach, et al., 2019; Linial et al., 2021; Yin, Le Guen, et al., 2021). Finally, we introduce an alternate optimization algorithm which convergence is shown in section 4.3.2 for a linear approximation of f_T.

4.2.1 *Structural Constraints for Dynamical Systems*

To ensure identifiability, we derive regularizations on h_T^p and h_T^d flowing from the control of an upper bound of $\mathrm{d}(h_T, f_T)$. In particular, to minimize $\mathrm{d}(h_T^p, f_T^p)$ would enable us to accurately interpret h_T^p as the true f_T^p, and h_T^d as the residual dynamics f_T^d. However, since we do not access the parameters of f_T^p, computing $\mathrm{d}(h_T^p, f_T^p)$ is not tractable. We then consider two possible situations. In the setting 1 below, the only available information on the physical system is the parametric form of f_T^p (or equivalently of h_T^p). In the setting 2 below, we consider available auxiliary information about f_T^p that will be used to minimize the distance between h_T^p and f_T^p. While the first setting is the more general, the physical prior it relies on is often insufficient to effectively handle real world situations. The second setting makes use of more informative priors on the physics and better corresponds to real cases as shown in chapter 5.

Setting 1: Controlling the ML Component and the MB Hypothesis

We propose a general approach to constrain the learning of hybrid models when one solely access the functional form of h_T^p. In this case, to make h_T^p accountable in our observed phenomena,

a solution is to minimize $\mathrm{d}(h_T^p, f_T)$. Following the triangle inequality we link up both errors $\mathrm{d}(h_T, f_T)$ and $\mathrm{d}(h_T^p, f_T)$:

$$\begin{aligned} \mathrm{d}(h_T, f_T) &= \mathrm{d}(h_T, f_T) + \mathrm{d}(h_T^p, f_T) - \mathrm{d}(h_T^p, f_T) \\ &\leq \mathrm{d}(h_T^p, f_T) + |\mathrm{d}(h_T, f_T) - \mathrm{d}(h_T^p, f_T)| \\ &\leq \mathrm{d}(h_T^p, f_T) + \mathrm{d}(h_T, h_T^p) \end{aligned} \tag{26}$$

As $\mathrm{d}(h_T, h_T^p) = \mathrm{d}(h_T^d, 0)$, we finally have

$$\boxed{\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^d, 0) + \mathrm{d}(h_T^p, f_T)} \tag{27}$$

We want the physical-statistical model $h_T = h_T^p + h_T^d$ to provide high quality forecasts. Minimizing the sole upper bound does not ensure such aim, as h_T^d is only penalized through $\mathrm{d}(h_T^d, 0)$ and is not optimized to contribute to predictions. We thus propose to minimize $\mathrm{d}(h_T, f_T)$ while controlling both $\mathrm{d}(h_T^d, 0)$ and $\mathrm{d}(h_T^p, f_T)$. Such a control of the upper bound of eq. (27) amounts to balancing the contribution of the ML and the MB components. This will be formally introduced in section 4.2.2.

LINK TO THE LITERATURE The optimization of $\mathrm{d}(h_T^p, f_T)$ to match the physical model with observations is studied in (Forssell and Lindskog, 1997). We propose to optimize an upper bound to $\mathrm{d}(h_T, f_T)$ based on the introduction of the term $\mathrm{d}(h_T^p, f_T)$ in eq. (26). While the least action principle on the ML component, i.e. constraining $\mathrm{d}(h_T^d, 0)$, is invoked for a geometric argument in (Yin, Le Guen, et al., 2021), it appears as a co-product of the introduction of $\mathrm{d}(h_T^p, f_T)$ in eq. (27). Thus, our framework allows to constrain both components of a hybrid model, thus strengthening the soundness of the physical component h_T^p compared to (Yin, Le Guen, et al., 2021), as shown in our experiments (section 4.4).

The general approach of eq. (27) allows us to perform prediction (via h_T) and system identification (via h_T^p) on simple problems (see experiments in section 4.4 and appendix C). However, the learning of real-world complex dynamics, via data-driven hybrid models, often fails at yielding a physically sound estimation. This suggests that, given the decomposition assumption $f_T = f_T^p + f_T^d$, the observations associated with the dynamics are not sufficient to estimate f_T^p. Thus, learning complex dynamics requires additional information.

In many real-world cases, auxiliary information is available in the form of measurements providing complementary information on f_T^p. Indeed, a common issue in physics is to infer an unobserved variable of interest (in our case f_T^p parameters U) from indirect or noisy measurements. For instance, one can access a physical quantity but only at a coarse resolution, as in (Belbute-Peres, Economon, and Kolter, 2020; Um et al., 2020). Let us denote f_T^{pr} the coarse version of f_T. It is natural to make the same decomposition assumptions: f_T^{pr} obeys the additive decomposition hypothesis of eq. (18), so that $f_T^{pr} = f_T^{p,pr} + f_T^{d,pr}$.

In this thesis, we aim at estimating the surface velocity fields U (that is f_T^p parameters). In reality, we cannot access observations of high resolution surface currents. However, thanks to observations of the Sea Surface Height (SSH), ocean surface currents can be estimated at a coarse resolution. Figure 12 shows different resolutions of ocean surface currents, that could be used to estimate f_T^p. Such information could then be used in order to approximate $\mathrm{d}(h_T^p, f_T^p)$ instead of $\mathrm{d}(h_T^p, f_T)$. This will be explored in the next subsection.

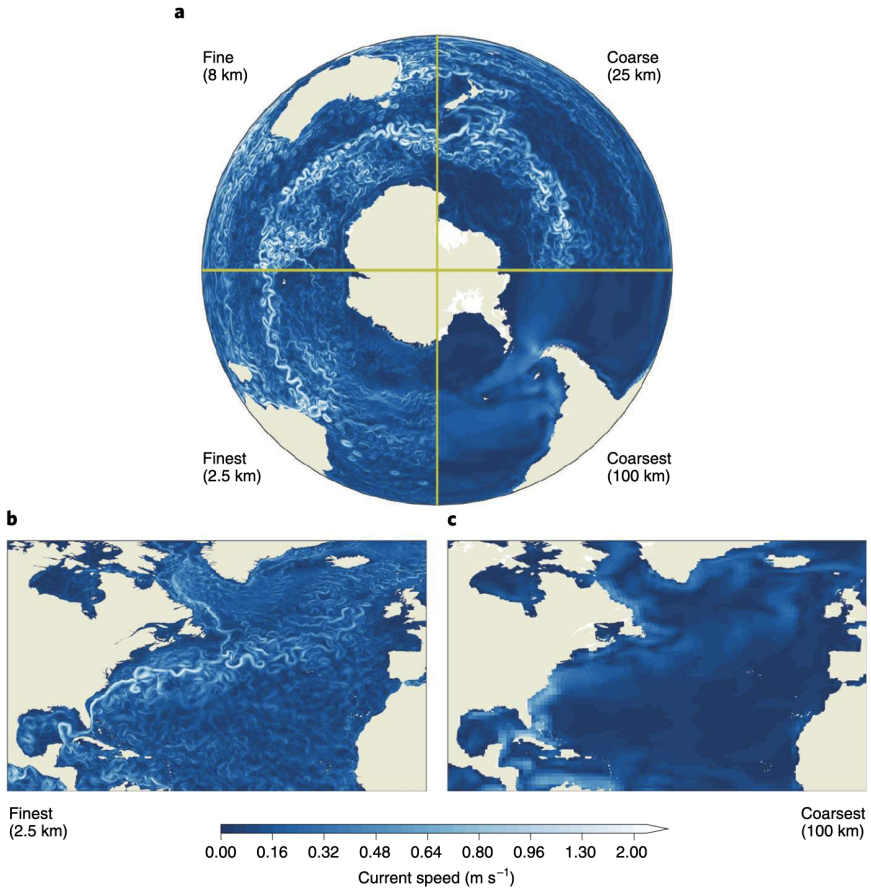

Figure 12: Multi scale resolution of ocean surface currents in the North Atlantic Ocean. From Hewitt et al. (2022). For example, the finest resolution (2.5km) corresponds to real f_T^p parameters. But only the coarse resolution (25km) is available. This one is denoted $f_T^{p,pr}$.

Setting 2: Matching the Physical Hypotheses by introducing Auxiliary Data

We here assume one accesses a coarse version of f_T^p, denoted $f_T^{p,pr} \in \mathcal{H}_k$. More precisely, we assume that $f_T^{p,pr}$ comes from a dynamics f_T^{pr}, which is a first-guess model of the true dynamics f_T. f_T^{pr} obeys the additive decomposition hypothesis of eq. (18), so that f_T^{pr} and $f_T^{p,pr}$verify $f_T^{pr} = f_T^{p,pr} + f_T^{d,pr}$. Our goal is to adapt our framework to incorporate such auxiliary information, bringing the regularization induced by $f_T^{p,pr}$ within the scope of the control of an upper bound. This enables us to extend our proposition towards the solving of real world physical problems, still largely unexplored by the ML community. With computations similar to eq. (27), we have:

$$\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T, f_T^{pr}) + \mathrm{d}(f_T^{pr}, f) \tag{28}$$

Then:

$$\begin{aligned}
\mathrm{d}(h_T, f_T^{pr}) &= \mathrm{d}(h_T, f_T^{pr}) + \mathrm{d}(h_T^p, f_T^{p,pr}) - \mathrm{d}(h_T^p, f_T^{p,pr}) \\
&\leq \mathrm{d}(h_T^p, f_T^{p,pr}) + |\mathrm{d}(h, f_T^{pr}) - \mathrm{d}(h_T^p, f_T^{p,pr})| \\
&\leq \mathrm{d}(h_T^p, f_T^{p,pr}) + |\mathrm{d}(h, f_T^{pr}) - \mathrm{d}(h_T, f_T^{p,pr}) - \mathrm{d}(h_T, f_T^{p,pr}) + \mathrm{d}(h_T^p, f_T^{p,pr})| \\
&\leq \mathrm{d}(h_T^p, f_T^{p,pr}) + |\mathrm{d}(h, f_T^{pr}) - \mathrm{d}(h_T, f_T^{p,pr})| + |\mathrm{d}(h_T^p, f_T^{p,pr}) - \mathrm{d}(h_T, h_T^p)| \\
&\leq \mathrm{d}(h_T^p, f_T^{p,pr}) + \mathrm{d}(f_T^{pr}, f_T^{p,pr}) + \mathrm{d}(h_T, h_T^p)
\end{aligned} \tag{29}$$

Combining Equations (28) and (29), we obtain:

$$\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^p, f_T^{p,pr}) + \mathrm{d}(h_T, h_T^p) + \mathrm{d}(f_T^{pr}, f_T^{p,pr}) + \mathrm{d}(f_T^{pr}, f_T) \tag{30}$$

Recall that we don't access f_T. f_T^{pr} is the coarse version of f_T, following the same decomposition asumption, i.e. $f_T^{pr} = f_T^{p,pr} + f_T^{d,pr}$. We suppose access to $f_T^{p,pr}$ parameters.

Finally, we have:

$$\boxed{\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^d, 0) + \mathrm{d}(h_T^p, f_T^{p,pr}) + \Gamma} \tag{31}$$

where $\Gamma = \mathrm{d}(f_T^{pr}, f_T^{p,pr}) + \mathrm{d}(f_T^{pr}, f_T)$ is a constant of the problem that cannot be optimized. It depends only on f_T^{pr}, $f_T^{p,pr}$ and f_T, variables that are beyond our control. As above, we propose to minimize $\mathrm{d}(h_T, f_T)$ while controlling both $\mathrm{d}(h_T^d, 0)$ and $\mathrm{d}(h_T^p, f_T^{p,pr})$, as described in section 4.2.2.

LINK TO THE LITERATURE In (Linial et al., 2021) $f_T^{p,pr}$ stands for true observations used to constrain a learned latent space, minimizing $\mathrm{d}(h_T^p, f_T^{p,pr})$. Jia, Willard, Karpatne, Read, Zwart, Michael S Steinbach, et al. (2019) uses synthetic data as $f_T^{p,pr}$ to pre-train their model which amounts to the control of an upper bound. Finally, this setting finds an extension, when the model $f_T^{p,pr}$ is a learned model, for example trained using eq. (27).

4.2.2 *Learning Algorithm and Optimization Problem*

From the upper bounds, we recover the well-posedness of the optimization and derive a theoretical learning scheme. Its practical implementation is developed in section 4.3.

RECOVERING WELL-POSEDNESS We reformulate the ill-posed learning of $\min_{h_T^p, h_T^d \in \mathcal{H}_p \times \mathcal{H}_d} \mathrm{d}(h_T, f_T)$, by instead optimizing $\mathrm{d}(h_T, f_T)$ while constraining the upper bounds. Let us define $\mathcal{S}_p$ and $\mathcal{S}_d$ as

$$\mathcal{S}_p = \{\, h_T^p \in \mathcal{H}_p \mid \ell(h_T^p) \leq \mu_p \,\} \qquad \mathcal{S}_d = \{\, h_T^d \in \mathcal{H}_d \mid \mathrm{d}(h_T^d, 0) \leq \mu_d \,\} \tag{32}$$

where μ_p, μ_d are two positive scalars and $\ell(h_T^p) = \mathrm{d}(h_T^p, f_T)$ in the case of setting 1 in section 4.2.1 and $\ell(h_T^p) = \mathrm{d}(h_T^p, f_T^{p,pr})$ in the case of setting 2 in section 4.2.1 . Our proposition then amounts to optimizing $\mathrm{d}(h_T, f_T)$:

$$\begin{aligned} &\min_{h_T \in \mathcal{S}_p + \mathcal{S}_d} \mathrm{d}(h_T, f_T) \\ &\text{with } \mathcal{S}_p + \mathcal{S}_d = \{\, h_T = h_T^p + h_T^d \mid h_T^p \in \mathcal{S}_p, h_T^d \in \mathcal{S}_d \,\} \end{aligned} \tag{33}$$

This constrained optimization setting enables us to recover the well-posedness of the optimization problem under the relative compactness of the family of function $\mathcal{H}_p$ (proof in appendix C.1.2).

Proposition 1 (Well-posedness)**.** *Under the relative compactness of $\mathcal{S}_p$, eq. (33) finds a solution h that writes as $h = h_T^p + h_T^d \in \mathcal{S}_p + \mathcal{S}_d$. Moreover, this solution is unique.*

ALTERNATE OPTIMIZATION ALGORITHM As the terms in both upper bounds of eqs. (27) and (31) specifically address either h_T^p or h_T^d, we isolate the losses relative to h_T^p and h_T^d and alternate projections of h_T^p on $\mathcal{S}_p$ and h_T^d on $\mathcal{S}_d$, as described in Algorithm 1. Said otherwise, we learn h_T by alternately optimizing h_T^p (h_T^d being fixed) and h_T^d (h_T^p being fixed). In practice, we rely on a dual formulation, which we develop in section 4.3. Besides, note that f_T is unknown, the practical computation of $\mathrm{d}(h_T^p, f_T)$ is developed in section 4.3.

Algorithm 1 Alternate estimation: General Setting

Result: Converged h_T^p and h_T^d
Set $h_T^{d,0} = 0,\ h_T^{p,0} = \min_{h_T^p \in \mathcal{H}_p} \mathrm{d}(h_T^p, f_T)$, $tol \in \mathbb{R}^+$
while $\mathrm{d}(h_T, f_T) > tol$ **do**

$$\begin{aligned} h_T^{p,n+1} &= \arg\min_{h_T^p \in \mathcal{S}_p} \mathrm{d}(h_T^p + h_T^{d,n}, f_T) \\ h_T^{d,n+1} &= \arg\min_{h_T^d \in \mathcal{S}_d} \mathrm{d}(h_T^{p,n+1} + h_T^d, f_T) \end{aligned} \tag{34}$$

  $n \leftarrow n + 1$
end

The convergence of the alternate projections is well studied for the intersection of convex sets or smooth manifolds (Lewis and Malick, 2008; Neumann, 1950) and has been extended in our setting of Minkowski-sum of convex sets (Lange, Won, and J. Xu, 2019). Because d as defined in section 2.3, i.e. $\mathrm{d}(h, f) = \mathbb{E}_{s \sim p_S} \|h(s) - f(s)\|_2$ is convex, $\mathcal{S}_d$ and $\mathcal{S}_p$ are convex sets as soon as $\mathcal{H}_p$ and $\mathcal{H}_d$ are convex (Appendix A.2). Thus, if $\mathrm{d}(., f_T)$ is strongly convex, eq. (34) finds one and only one solution (S. Boyd, S. P. Boyd, and Vandenberghe, 2004). However, neither the convexity of $\mathcal{H}_d$ nor of $\mathcal{H}_p$ is practically ensured. Nonetheless, we recover the well-posedness of eq. (33) and show the convergence of Algorithm 1 in the simplified case where h_T is an affine function of X_t (see section 4.3.2). For complex cases, for which theoretical analysis cannot be conducted, we validate our approach experimentally and we evidence in section 4.4 that this formulation enables us to recover both an interpretable decomposition $h_T = h_T^p + h_T^d$ and improved prediction and identification performances.

In the next section, we develop the practical implementation of the alternate estimation algorithm. In particular, we explain the calculation of eq. (34) and give the actual implementation of Algorithm 1.

4.3 PRACTICAL OPTIMIZATION AND THEORETICAL ANALYSIS

4.3.1 *Practical Optimization*

As discussed in chapter 2(section 2.3), because f_T is unknown, $\mathrm{d}(h_T, f_T)$ is not tractable. Thus, we rely on the flow-based distance δ (introduced in section 2.3):

$$\delta(h_T, f_T) = \mathbb{E}_{X_0} \int_{t_0}^{t} \left\| \phi_{h_T}(X_0, \tau) - \phi_{f_T}(X_0, \tau) \right\|_2 \mathrm{d}\tau$$

While eq. (32) involves the choice of μ_p and μ_d, in practice we don't calculate them. We implement the projection algorithm by descending gradients on the parameters of h_T^p and h_T^d according to the alternate algorithm 1, with respect to the following optimization problems:

$$\begin{aligned} &\min_{h_T^p} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \ell(h_T^p) \\ &\min_{h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^d} \delta(h_T^d, 0) \end{aligned} \tag{35}$$

where

$$\ell(h_T^p) = \begin{cases} \delta(h_T^p, f_T) & \text{in the case of setting 1} \\ \delta(h_T^p, f_T^{p,pr}) & \text{in the case of setting 2} \end{cases} \quad \text{in section 4.2.1}$$

where $\lambda_{h_T}, \lambda_{h_T^p}, \lambda_{h_T^d}$ are positive real values.

In practice, we rely on the two following losses:

$$\mathcal{J}_T(t) = \lambda_{h_T} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}, h_T, t) - T_t \right\|_2 + \lambda_{h_T^d} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}, h_T^d, t) \right\|_2 \tag{36}$$

$$\mathcal{J}_U(t) = \lambda_{h_T} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}, h_T, t) - T_t \right\|_2 + \lambda_{h_T^p} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}, h_T^p, t) - T_t \right\|_2 \tag{37}$$

where

$$\mathrm{ODESolve}(\hat{X}_{t_0}, h_T, t) = \hat{X}_{t_0} + \int_{t_0}^{t} h_T(\hat{X}_{t_0}) \mathrm{d}\tau$$

The optimization of eq. (35) consists in learning h_T^p and h_T^d by minimizing the overall cost function $\mathcal{J}$ defined by

$$\mathcal{J} = \sum_{i=1}^{n} \sum_{j=0}^{k} (\mathcal{J}_U(t_0 + j\Delta t) + \lambda_T \mathcal{J}_T(t_0 + j\Delta t)) \tag{38}$$

$$\begin{aligned} \text{with } \mathcal{J}_T(t) &= \lambda_{h_T} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}^i, h_T, t) - T_t^i \right\|_2 + \lambda_{h_T^d} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}^i, h_T^d, t) \right\|_2 \\ \mathcal{J}_U(t) &= \lambda_{h_T} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}^i, h_T, t) - T_t^i \right\|_2 + \lambda_{h_T^p} \left\| \mathrm{ODESolve}(\hat{X}_{t_0}^i, h_T^p, t) - T_t^i \right\|_2 \end{aligned}$$

where $\lambda_{h_T}, \lambda_{h_T^p}, \lambda_{h_T^d}$ are hyperparameters, n is the number of initial conditions in the training set and k is an hyperparameter. We alternate the optimization of the parameters θ_p and θ_d of h_T^p and h_T^d according to algorithm 2.

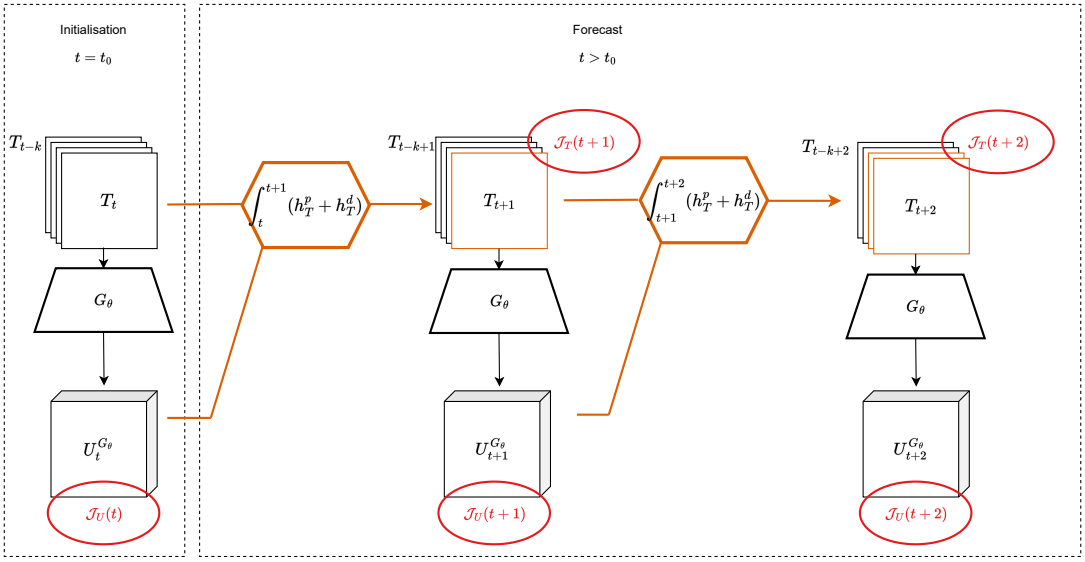

Figure 13: Computational graph of the proposed model on two time steps at training time. Forecasting amounts to learning G_θ and h_T^d from a sequence of observations of T. At each timestep t, we estimate U_t and we predict T at the next timestep $t+1$. Note that the inverse model G_θ is used at every timestep. The forcing term F are modeled with h_T^d and learned from a sequence of T.
Note that $t-k$ actually refers to the timestep $t-k\Delta t$.

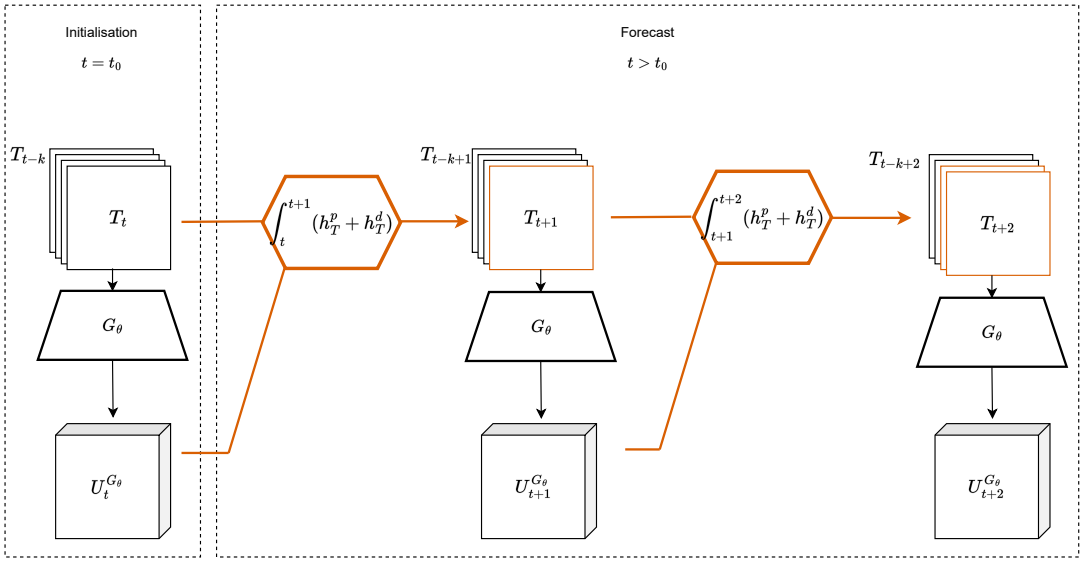

Figure 14: Computational graph of the proposed model on two time steps at inference time. Forecasting amounts to learning G_θ and h_T^d from a sequence of observations of T. At each timestep t, we estimate U_t and we predict T at the next timestep $t+1$. Note that the inverse model G_θ is used at every timestep.

The computational graphs from Figures 13 and 14 givea schematic representation of our learning scheme over two time steps respectively at training and inference time.

Note that the minimisation of $\delta(h_T^p, f_T)$ alone biases our estimate of h_T^p. However, it may yield a good estimation of h_T^p provided that f_T^p contributes significantly to the prediction of h_T. Hence, $\delta(h_T^p, f_T)$ can be interpreted has an initialization loss, yielding a first estimate of θ_p explaining the dynamics. Thus, in most applications, we progressively decrease its magnitude along training. On the other hand, $\delta(h_T^d, 0)$ aims at constraining the free form function h_T^d to make its action as small as possible. Hence, it can be interpreted as a balance loss, preventing the neural networks to override the physical component. Finally, in order to recover the exact trajectories of the observations, we proceed as suggested in (Yin, Le Guen, et al., 2021), progressively increasing the hyperparameter λ_{h_T} according to a rate τ_2. The practical implementation is summarized in the following algorithm, where θ_d are the parameters of h_T^d and θ_p are the parameters of h_T^p, τ_1 is the learning rate.

Algorithm 2 Alternate estimation: Practical Setting

Initialization: $\theta_d^0 = 0,\ \theta_p^0 = \min_{h_T^p \in \mathcal{H}_p} \delta(h_T^p, f_T)$, λ_{h_T}, $\lambda_{h_T^p}$, $\lambda_{h_T^d}$

for $epoch = 1 : N_{epochs}$ **do**

  for $batch = 1 : B_k$ **do**

$$\theta_p^{n+1} = \theta_p^n - \tau_1 \nabla_{\theta_p} [\lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \ell(h_T^p)]$$

  end

  for $batch = B_k : B_u$ **do**

$$\theta_d^{n+1} = \theta_d^n - \tau_1 \nabla_{\theta_d} [\lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^d} \delta(h_T^d, 0)]$$

  end

  $\lambda_{h_T} = \tau_2 \lambda_{h_T}$; $\lambda_{h_T^p} = \frac{1}{\tau_2} \lambda_{h_T^p}$; $\lambda_{h_T^d} = \frac{1}{\tau_2} \lambda_{h_T^d}$

end

4.3.2 *Theoretical Analysis for a Linear Approximation*

We investigate the validity of our proposition when approximating an unknown derivative with an affine function, which is a classical first guess approximator. We here consider h_T^p as a linear function. We do not assume any information on f_T, thus relieving this section from the need of an accurate prior knowledge f_T^p. In this context, we show the convergence of the learning scheme introduced in Algorithm 1 with $\ell = \delta(h_T^p, f_T)$, demonstrating the validity of our framework in this simplified setting. For more complex cases, for which theoretical analysis cannot be conducted, our framework is validated experimentally in section 4.4. All proofs of this section are conducted using the distance δ from section 2.3:

$$\delta(h, f) = \mathbb{E}_{X_0} \int_{t_0}^{t} \left\| \phi_h(X_0, \tau) - \phi_f(X_0, \tau) \right\|_2 \mathrm{d}\tau$$

Let X^s be the unique solution to the initial value problem:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = f_T(X_t) \quad \text{with} \quad X_{t=0} = X_0 \tag{39}$$

With $h_T^p(X) = AX$ and $h_T^d(X) = D_A$, where $A \in \mathcal{M}_{p,p}(\mathbb{R})$, $D_A \in \mathbb{R}^p$, and X^s the solution to eq. (39), the affine approximation of f_T writes as:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = AX_t + D_A \quad \text{with} \quad X_{t=0} = X_0 \tag{40}$$

We write X^D the solution to eq. (40) when $D_A \neq 0$ and X^A the solution to eq. (40) when $D_A = 0$. The alternate projection algorithm with the distance δ writes as:

$$\begin{aligned}\hat{A} &= \arg\min_{h_T^p} \lambda_{h_T}\delta(h_T, f_T) + \lambda_{h_T^p}\delta(h_T^p, f_T) \\ &= \arg\min_{A} \int_{t_0}^{t} \left\|X^s(\tau) - X^D(\tau)\right\|_2 \, d\tau + \lambda_A \int_{t_0}^{t} \left\|X^s(\tau) - X^A(\tau)\right\|_2 \, d\tau \end{aligned} \tag{41}$$

$$\begin{aligned}\hat{D}_A &= \arg\min_{h_T^d} \lambda_{h_T}\delta(h_T, f_T) + \lambda_{h_T^d}\delta(h_T^d, 0) \\ &= \arg\min_{D_A} \int_{t_0}^{t} \left\|X^s(\tau) - X^D(\tau)\right\|_2 \, d\tau + \lambda_D \|D_A\|_2 \end{aligned} \tag{42}$$

where $\lambda_D = \frac{\lambda_{h_T^d}}{\lambda_{h_T}}$, $\lambda_A = \frac{\lambda_{h_T^p}}{\lambda_{h_T}} > 0$. We detail in Algorithm 3 the alternate projection algorithm in a linear setting.

Algorithm 3 Alternate estimation: Linear Setting

Result: $A \in \mathcal{M}_{p,p}(\mathbb{R}), D \in \mathbb{R}^p$
$k = 0$, $D^0 = 0$, $A_0^{-1} = 0$ $A_0^0 = min_A \|X^s - XA\|$
while $\|D^k - D^{k-1}\| > \epsilon$ *and* $\|A^k - A^{k-1}\| > \epsilon$ **do**
  $D^{k+1} = \min_D \|X^s - XA^k - D\|_2^2 + \lambda_D \|D\|_2^2$
  $A^{k+1} = \min_A \|X^s - XA - D^{k+1}\|_2^2 + \lambda_A \|X^s - XA\|_2^2$
  $k \leftarrow k + 1$
end

As the optimization of eq. (41) is not convex on A, the solution existence and uniqueness is not ensured. The well-posedness w.r.t A can be recovered by instead considering a simple discretization scheme, e.g. $X_{t+1} \approx (AX_t + D_A)\Delta t + X_t$ and solving the associated least square regression, which well-posedness is guaranteed, see details in appendix C.1.1. Such strategy is common practice in system identification. Theoretical considerations on existence and uniqueness of solutions to eqs. (41) and (42) are hard to retrieve. If A is an invertible matrix (proof is available in appendix C.1.3):

Proposition 2 (Existence and Uniqueness)**.** *If A is invertible, There exists a unique D_A, hence a unique X^D, solving eq. (42).*

Finally, formulating Algorithm 1 as a least square problem in an affine setting we prove the convergence of the alternate projection algorithm (appendix C.1.4):

Proposition 3. *For λ_D and λ_A sufficiently high, the algorithm that alternates between the estimation of A and the estimation of D_A following eqs. (41) and (42) converges.*

4.4 EXPERIMENTS

In this section, we validate Algorithm 1 on the Adv+F dataset depicting idealistic dynamics (its generation is detailed in chapter 2 in section 2.2). We assess the relevance of our proposition based on eq. (27), against NeuralODE (R. T. Q. Chen et al., 2018), Aphynity (Yin, Le Guen, et al., 2021) and ablation studies. Note that we conduct experiments on eq. (31) by considering additional auxiliary information, but present the associated results on the NATL60 data in chapter 5.

We consider three optimization problems. The original problem:

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \delta(h_T^p, f_T) + \lambda_{h_T^d} \delta(h_T^d, 0) \tag{43}$$

We call "Ours" the results with training induced by eq. (43). When $\delta(h_T^p, f_T)$ is not considered in the optimization of eq. (43),we then train according to:

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \delta(h_T^p, f_T) \tag{44}$$

When $\delta(h_T^d, 0)$ is not considered in the optimization of eq. (43), we then train according to:

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^d} \delta(h_T^d, 0) \tag{45}$$

When h_T is trained by only minimizing the discrepancy between actual and predicted trajectories, i.e. by only optimizing according to the loss $\delta(h_T, f_T)$, we train according to:

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) \tag{46}$$

4.4.1 *Training Details*

All experiments were conducted on NVIDIA TITAN X GPU using Pytorch (Paszke et al., 2019).

ARCHITECTURES DETAILS We parameterize G_θ by U-net with at most 512 latent channels also following the implementation of (Isola et al., 2017a), taking as input a sequence of 4 time steps of T: $(T_{t_0}, \dots, T_{t_0+3\Delta t})$. The residual dynamics h_T^d is learned by a convolutional ResNet, with 1 residual block taking as entry the same sequence of T. We implement h_T^p via a semi-lagrangian scheme, taking as input T_t and the estimated U_t to predict T_{t+1}.

HYPERPARAMETERS, SETTING OF EQ. (43) We select the hyperparameters with the lowest prediction errors (i.e lowest $\delta(h_T, f_T)$). We initialize $\lambda_{h_T^p} = 0.1$ and decrease it geometrically down to $\lambda_{h_T^p} = 0.00001$. We initialize $\lambda_{h_T} = 0.01$ and increase it geometrically every epoch up to $\lambda_{h_T} = 1000$. $\lambda_{h_T^d}$ is fixed through training at 0.1. The training time for this dataset is 8 hours.

OPTIMIZATION We use Adam optimizer with learning rate 0.0001 for 30 epochs with batch size 32. We supervise the trajectories up to $t = \Delta t \times 6$, i.e we enforce δ on $(T_{t_0+\Delta t}, \dots, T_{t_0+6\Delta t})$. We alternate projection on $\mathcal{S}_p$ and $\mathcal{S}_d$ by descending the gradient 4-batches on h_T^p then 6-batches on h_T^d.

Table 2: Results for the Adv+F data. We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the forcing term F over 6 time steps on test set.

Models	T	U	F
Ours	**0.74 (0.05)**	**1.99 (0.13)**	**0.17 (0.01)**
Aphynity	0.85 (0.35)	3.07 (0.74)	0.18 (0.05)
NeuralODE	1.35 (0.02)	n/a	n/a

4.4.2 *Results*

As in chapter 3, we don't learn the dynamics h_T but its numerical integration: h_T^p implements a differentiable semi-Lagrangian scheme (Jaderberg et al., 2015) and h_T^d is a ResNet. As in chapter 3, we estimate U from observations of T, using an inverse model G_θ. G_θ is a UNet which takes as input 4 timesteps of predicted temperature T.
Performances are evaluated via the standard metric MSE (lower is better). We report between brackets the standard deviation of the metrics over 5 runs.

IDENTIFICATION AND PREDICTION RESULTS To make predictions of T, we aim at both estimating the hidden parameters of h_T^p, i.e. estimating the velocity U, and learning h_T^d, i.e. estimating the forcing term F. Figures 15 to 18 show examples of estimated hidden states. Columns labeled U and F in Tables 2 and 3 give the MSE between estimation and target hidden state.
Table 2 indicates that for the Adv+F dataset, we estimate accurately the unobserved velocity fields U and forcing term F. Qualitatively, Figure 15 shows that controlling our proposed upper bound eq. (27) facilitates the recovery of truthful velocity fields U along with an accurate prediction of T. Regarding prediction performances on the Adv+F data, Table 2 shows that thanks to our truthful estimates of U and F, our model provides more precise prediction of T than NODE and Aphynity. Besides, adding prior knowledge in the prediction systems improves prediction performances: NODE minimizes $\delta(h_T, f_T)$ by predicting average and blurred frames. This shows the need for regularization when learning on structured physical data.

ABLATION STUDY We present in Table 3 an ablation study on the Adv+F dataset evidencing the influence of our learning choices on the solution of both identification and prediction tasks. "Joint" rows of Table 3 indicate that the learning of h_T^d and h_T^p is done simultaneously. Table 3 shows that the sole optimization of $\delta(h_T, f_T)$ fails at estimating physically sounded U. This evidences the ill-posedness in such unconstrained optimization. Table 3 indicates that all introduced regularizations improve the recovery of U w.r.t. the «Only $\delta(h_T, f_T)$» baseline, while adding $\delta(h_T^d, 0)$ significantly improves both prediction performances and velocity fields estimation. We highlight that the alternate optimization performs better compared to optimizing jointly all parameters of h_T^p and h_T^d. Notably, our proposition to optimize h_T^p and h_T^d alternately beats all baselines on both T prediction and U identification (Table 3, Joint rows). Finally, jointly trained models fail at estimating U in Table 3, forcing h_T^d to capture the whole dynamics.

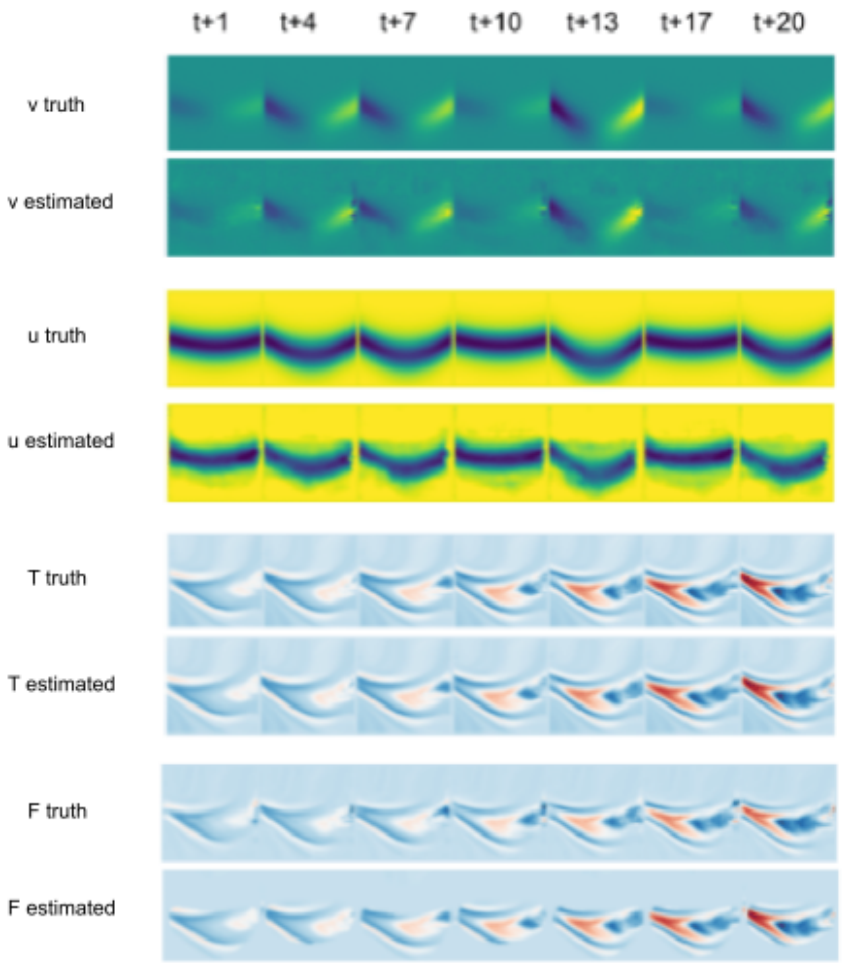

Figure 15: Sequence of estimations of F, T and $U = (u, v)$ on the Adv+F dataset, coming from the test set. For each variable T, u and v, the first row shows the ground truth, the second row the estimation with our model optimization based on eq. (43). Columns represent the time. Note that $t + 1$ actually means $t + \Delta t$ where $\Delta t = 8640$s. Without ever observing U, our model is able to estimate it from observations of T and make predictions over long time range.

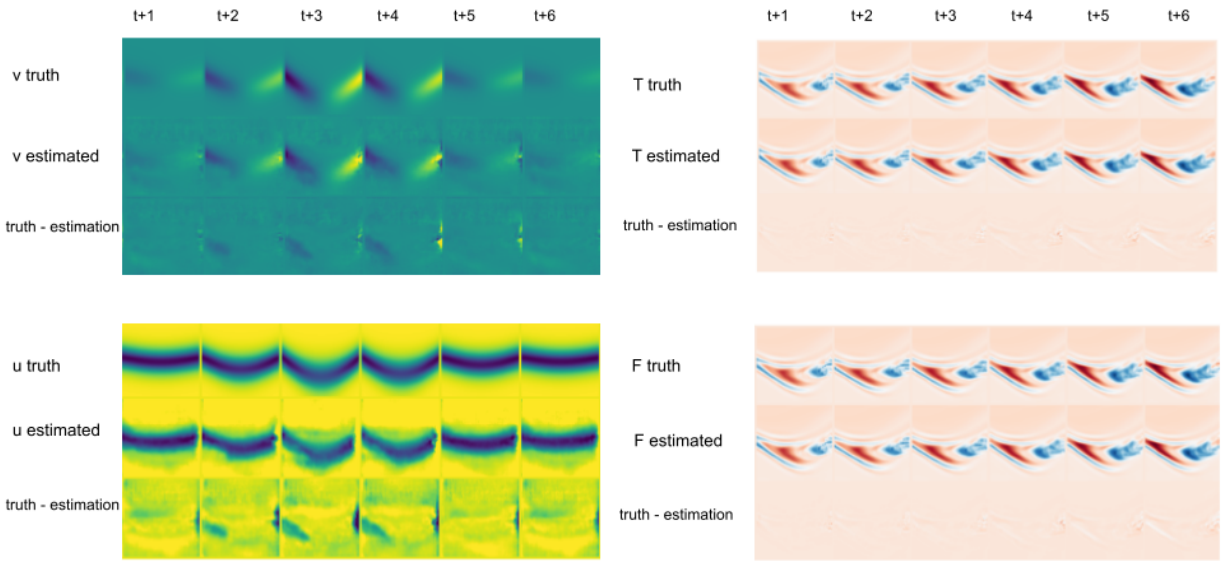

Figure 16: Estimations, targets and differences between estimations and targets on T, $U = (u, v)$ and F on the Adv+F dataset, coming from the test set. Each column refers to a time step. Note that $t + 1$ actually means $t + \Delta t$ where $\Delta t = 8640$s. On the left, true and estimated $U = (u, v)$ over 6 time steps, and differences between targets and estimations. On the right, prediction of T and F over 6 time steps, and differences between targets and estimations.

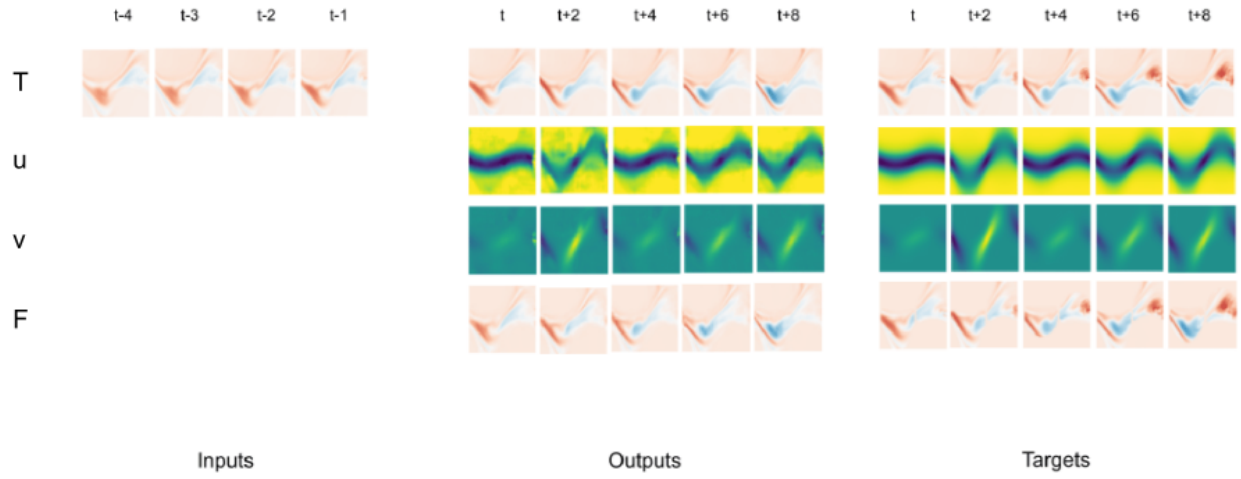

Figure 17: Estimations and targets on T, $U = (u, v)$ and F on the Adv+F dataset, coming from the test set. Each column refers to a time step. Note that $t + 1$ actually means $t + \Delta t$ where $\Delta t = 8640$s. On the left, sequence of T inputs (4 time steps). In the middle, prediction of T, $U = (u, v)$ and F over 8 time steps. On the right, true T, U and F over 8 time steps.

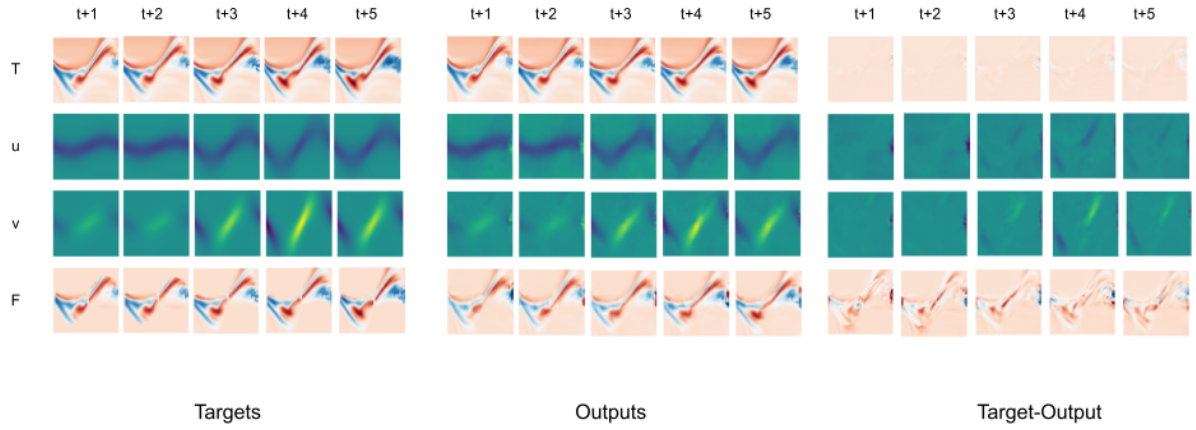

Figure 18: Estimations, targets and differences between estimations and targets on T, $U = (u, v)$ and F on the Adv+F dataset, coming from the test set. Each column refers to a time step. Note that $t + 1$ actually means $t + \Delta t$ where $\Delta t = 8640$s. On the left, true T, U and F over 5 time steps. In the middle, prediction of T, $U = (u, v)$ and F over 8 time steps. On the right, differences between targets and estimations.

Table 3: Ablation Study on Adv+F. We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the forcing term F over 6 time steps. "Joint" rows refer to the simultaneous optim. of h_T^p and h_T^d.

Training	Models	T	U	F
	Ours (U known)	0.52	n/a	0.19
Alternate	Ours	**0.74 (0.05)**	**1.99 (0.13)**	**0.17 (0.01)**
	eq. (46)	1.02 (0.16)	4.08 (0.23)	0.19 (0.06)
	eq. (44)	1.02 (0.09)	3.66 (0.15)	0.19 (0.03)
	eq. (45)	0.77 (0.06)	2.38 (0.17)	0.19 (0.01)
Joint	Ours	1.44 (0.08)	3.30 (0.18)	0.30 (0.03)
	eq. (46)	1.38 (0.19)	6.96 (0.21)	0.39 (0.08)

4.5 CONCLUSION

We propose in this work an algorithm to learn hybrid MB/ML models. For interpretability purposes, we impose constraints flowing from an upper bound of the prediction error and derive a learning algorithm in a general setting. We prove its well-posedness and its convergence in a linear approximation setting. Empirically, we evidence the soundness of our approach thanks to ablation studies and comparison with recent baselines on our simplified ocean dynamics dataset. In the next part iii, we go further and confront this framework to model data representative of real-world ocean dynamics.

Part III

REAL-LIKE OCEAN DATA

This part is more practical than methodological. We dig deeper than part ii, exploring the possibility to model real-like ocean dynamics with ML. The idea introduced so far highlights a principled methodology for completing physical models with ML components. Our experiments show their validity for simple dynamics and their associated simulated data.

In chapter 5, we confront the models developed in part ii to the data from NATL60. When dealing with real or real-like dynamics and data, one is faced with a "reality gap" due to their increased complexity so that this principled approach shall be further enhanced with e.g. the incorporation of additional information sources. In this part, we make a first attempt towards this direction by augmenting the model from chapter 4 with additional information.

5 PHYSICAL-STATISTICS MODELS FOR REAL-LIFE LIKE OCEAN DATA

5.1 INTRODUCTION

In this section, we focus on NATL60 data, which are simulations of real data. Even if many uncertainties inherent to real observational data are not present, e.g. uncertainties due to cloud cover or measuring devices, these data reflect closely the complexity of real observations. Note that ocean dynamics is a tri-dimensional spatial phenomenon, involving many variables on several interrelated scales. Thus, whereas the simulated data used in part ii are bi-dimensional, the phenomena studied in this section is tri-dimensional. For convenience, we only focus on the surface velocity fields, and ignore the vertical components.

In line with several works presented in chapter 2, we seek to confront data-driven models to real-life like data. With this in mind, we do not claim to solve a problem from the oceanography field. We rather aim at detecting the limits of our theoretical models, when confronted to complex dynamics. Thus, this chapter extends the previous chapters, adapting the methodologies presented to simulations of real data. First, section 5.2 is a reminder of our model proposed in chapter 4. Then, in section 5.3, we analyse the performance of the model introduced in chapter 4 on the challenging NATL60. We then introduce and discuss different attempts for adapting our model to the complexity of NATL60. Finally, in section 5.4, research directions are proposed to improve the results and pave the way for the development of new models.

5.2 HYPOTHESIS AND MODEL REMINDERS

We access a partially observed dynamical state $X_t = (T_t, U_t)$ where the ocean surface temperature T is observed and the ocean surface current velocity fields U are unobserved. We aim at estimating the dynamics f_T of T with an hybrid model $h_T = h_T^p + h_T^d$, where h_T^p is the physical part with known form and h_T^d is the data-driven part, completely unknown. h_T^p depends on unknown parameters θ_p (in our case that is the velocity U), which we aim to estimate from observations of T, using as inverse model a neural network G_θ.

To learn a physically sound $h_T = h_T^p + h_T^d$, we propose to add constraints on both components h_T^p and h_T^d. To that end, in chapter 4, we propose to derive our cost function from an upper bound. Constraining h_T^p to participate as many as possible for f_T leads us to eq. (27):

$$\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^d, 0) + \mathrm{d}(h_T^p, f_T)$$

associated with the optimization problem:

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \delta(h_T^p, f_T) + \lambda_{h_T^d} \delta(h_T^d, 0) \tag{47}$$

where

$$\delta(h_T, f_T) = \mathbb{E}_{X_0} \int_{t_0}^{t} \left\| \phi_{h_T}(X_0, \tau) - \phi_{f_T}(X_0, \tau) \right\|_2 \mathrm{d}\tau$$

However, we show in the next section that this seems insufficient when dealing with real-like data. Thus, we propose to add information on f_T^p. To that end, we assume one accesses a coarse version of f_T^p, denoted $f_T^{p,pr} \in \mathcal{H}_k$. Note that this is a plausible assumption when dealing with physical problems. More precisely, we assume that $f_T^{p,pr}$ comes from a dynamics f_T^{pr}, which is a first-guess model of the true dynamics f_T. f_T^{pr} obeys the additive decomposition hypothesis of eq. (18), so that f_T^{pr} and $f_T^{p,pr}$ verify $f_T^{pr} = f_T^{p,pr} + f_T^{d,pr}$. We adapt our framework to incorporate such auxiliary information, bringing the regularization induced by $f_T^{p,pr}$ within the scope of the control of an upper bound. We aim to constrain $\mathrm{d}(h_T^p, f_T^{p,pr})$, which leads us to eq. (31):

$$\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^d, 0) + \mathrm{d}(h_T^p, f_T^{p,pr}) + \Gamma$$

The optimization problem is then the following:

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \delta(h_T^p, f_T^{p,pr}) + \lambda_{h_T^d} \delta(h_T^d, 0) \tag{48}$$

In the following section, we explain how we adapt these two bounds and associated optimizations in the context of the NATL60 dataset.

5.3 WHEN THEORY MEETS REALITY

In this part, we examine the model proposed in chapter 4, as it is theoretically grounded. First, we show that the first bound is not sufficient to estimate U in the case of NATL60. To rely on the second upper-bound proposed in section 4.2, we assume that we access $f_T^{p,pr}$ parameters. In section 5.3.1 we present how we implement such bound and propose adaptation to deal with the complexity of NATL60 data. We also present experimental results. Then, in section 5.3.2, we extend this model, adding the regularization proposed in chapter 3. Finally, in section 5.3.3, we review the limitations of our model and present the avenues explored to address the encountered problems.

5.3.1 *Model Adjustment*

We train the model from chapter 4 on NATL60 data using both upper-bounds eqs. (47) and (48). Remember that the velocity fields U are the parameters of f_T^p, and we aim at estimating the parameters θ_p of h_T^p.

Using eq. (47)

We first rely on the optimization problem eq. (47), i.e.

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \delta(h_T^p, f_T) + \lambda_{h_T^d} \delta(h_T^d, 0)$$

Modeling the evolution of T in NATL60 is challenging as its dynamics is chaotic and highly non-linear, which is representative of the complexity encountered in real world data. According to Figure 19 and first row of table 4, the principled approach of eq. (47) is insufficient here and one must resort to additional physical information. We then propose to use eq. (48).

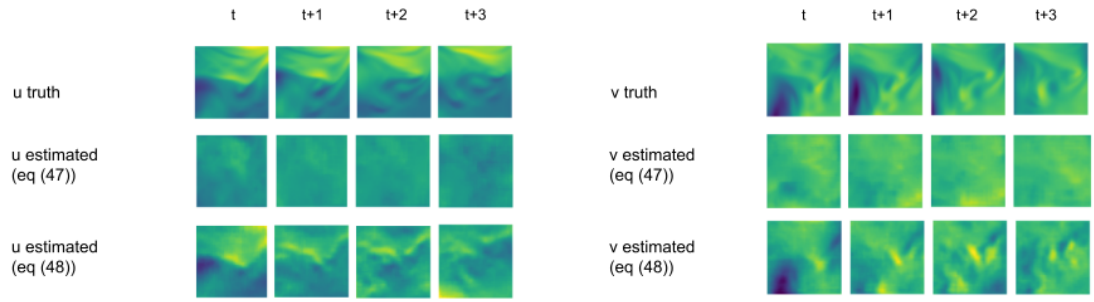

Figure 19: Sequence of estimations θ_p of $U = (u, v)$ for the NATL60 data. Left and right figures respectively illustrate both components u and v of the two dimensional $U = (u, v)$. The second and third row respectively refer to training according to eq. (47) and eq. (48). The loss term $\delta(h_T^p, f_T^{p,pr})$ in eq. (48) enables our model to learn more accurate velocity fields than when only trained following eq. (47).

Using eq. (48) with auxiliary data

We thus propose to rely on the optimization problem eq. (48), i.e.

$$\min_{h_T^p, h_T^d} \lambda_{h_T} \delta(h_T, f_T) + \lambda_{h_T^p} \delta(h_T^p, f_T^{p,pr}) + \lambda_{h_T^d} \delta(h_T^d, 0)$$

To use eq. (48), we need to calculate $\delta(h_T^p, f_T^{p,pr})$. As both f_T^p and $f_T^{p,pr}$ belong to the same functional space $\mathcal{H}_p$, they have identical parametric forms. This means that θ_p^{pr} are in fact estimations of real parameters of f_T^p, such that $f_T^{p,pr} = h_T^p(., \theta^{pr}) \approx f_T^p$. We thus propose to enforce $\mathrm{d}(h_T^p, f_T^{p,pr})$ directly into the parameter space, i.e. to minimize $\|\theta_p - \theta_p^{pr}\|_2$. Indeed, minimizing $\|\theta_p - \theta_p^{pr}\|_2$ will bring h_T^p closer to $f_T^{p,pr}$ and thus to f_T^p.
As a concrete example, consider Figure 20. In this figure, θ_p are the true surface currents, i.e. what is shown in high resolution in the red box. θ_p^{pr} are the currents derived from the SSH, i.e. what is shown in low resolution across the North Atlantic Ocean. They are rough estimates of high resolution surface currents.
To enforce $\mathrm{d}(h_T^p, f_T^{p,pr})$, we estimate θ_p with G_θ and supervise it with θ_p^{pr}. Note that we use as θ_p^{pr} the velocity fields U. However, in a real-case scenario, one does not access such observations: only coarse approximations of U are computable from the SSH, see chapter 2.
For the highly complex NATL60, fig. 19 and Table 4 (second row) show that the introduction of auxiliary data following the formulation in eq. (48) helps identification. The dynamics is too complex to be able to recover physically interpretable velocity fields using the bound of eq. (47). We thus propose to add more physical information to our model.

Adding More Information

There are conditions that have not been considered so far and for which the model can easily be adapted. Firstly in terms of spatial coordinates, and secondly regarding the Courant-Friedrichs-Lewy condition. We develop both aspects below.

IRREGULAR GRID NATL60 data results from an ocean model simulation at a resolution close to the kilometer, i.e one pixel represents the average value of the measured field over a surface of

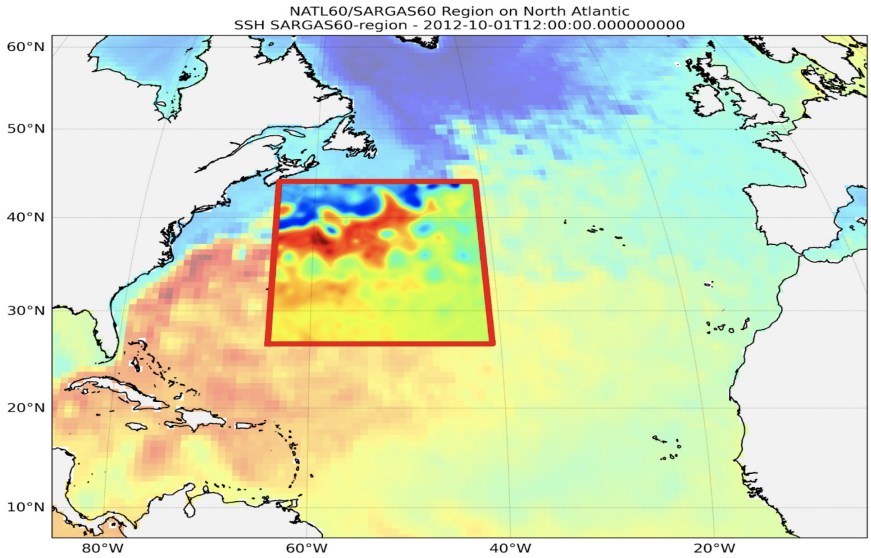

Figure 20: Sea Surface Height in the North Atlantic Ocean. The red box shows the ocean surface currents at a resolution of about 2.5km. Out of the red box, the resolution is about 50km. From Hewitt et al. (2022).

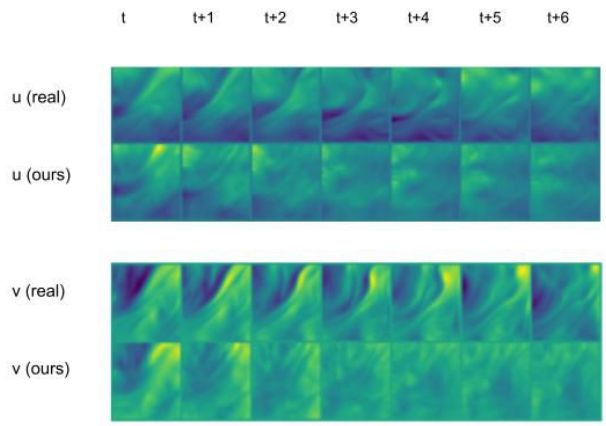

Figure 21: Sequence of estimations on T and $U = (u, v)$ for the NATL60 data. Each column refers to a time step and predictions range from 1 to 7 days. Estimations deteriorate very quickly.

Table 4: Ablation Study for the NATL60 data: we test our two upper-bounds from chapter 4 (first two rows) as well as the second bound with further physical knowledge (third row). We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the forcing term F over 3 time steps on test set.

Models	T	U	F
Ours eq. (47)	8.27 (0.06)	11.72 (0.07)	6.01 (0.08)
Ours eq. (48)	7.37 (0.12)	10.9 (0.1)	4.98 (0.09)
Ours eq. (48) $\star$	**6.86 (0.12)**	**6.81 (0.07)**	**4.35 (0.11)**
Aphynity	8.18 (0.16)	11.75 (0.49)	6.02 (0.02)
NeuralODE	8.83 (0.98)	n/a	n/a

approximately 1km $\times$ 1km. As the grid is not uniform across all the NATL60 domain, $\mathrm{d}x$ and $\mathrm{d}y$ are computed for each latitude and longitude, according to:

$$\mathrm{d}x = R\cos\phi\,\mathrm{d}\psi$$
$$\mathrm{d}y = R\,\mathrm{d}\phi$$

where $R = 6378$km is the Earth radius, ϕ is the latitude and ψ is the longitude. NATL60 is at a resolution of $1/60$ °, i.e. $\mathrm{d}\phi = \mathrm{d}\psi = 1/60$. We have $\Delta t = 86400\text{s} = 1\text{day}$. To run the semi-Lagrangian scheme, representing the advection of T by U, we thus compute $\mathrm{d}x$ and $\mathrm{d}y$ accordingly.

DEALING WITH THE CFL CONVERGENCE CONDITION There are difficulties inherent to the NATL60 data. As data are sampled every day, about every 2 km, the CFL convergence condition might therefore not be respected. Moreover, we access data for one year only, which leads to problems when learning and testing the modeled dynamics: the observation conditions are not the same in winter and summer, for example. To deal with the CFL convergence condition, we propose to adapt the size of images of T given as h_T^p inputs. Considering that an ocean water particle approximately travels 86 km per day, when computing the advection of T with the Semi-Lagrangian scheme, we thus need to give a border condition of about 40 grid points. To deal with numerical limits, we thus estimate U, T and F as 256×256 images. Note that this practice considerably increases the convergence time of the training and limits the forecast horizon.

Table 4 (third row denoted Ours eq. (48) $\star$) shows that adding such information enhances results. Thus, it seems that one cannot do without physical knowledge when dealing with real data. However, fig. 21 shows that the velocity fields reconstructed from eq. (48) are not yet satisfactory, especially regarding long-term predictions. In the following, we propose to add the dynamical constraint proposed in chapter 3 to overcome this limit.

5.3.2 *Hybrid models with dynamical regularization*

Long-term prediction seems to be a problem for learning on the NATL60 dataset. In fact, the accuracy of neural networks predictions for long horizons strongly depends on the capacity of the model in producing accurate outputs at each time-step. Indeed, error may accumulate leading to aberrant or unrealistic predictions.

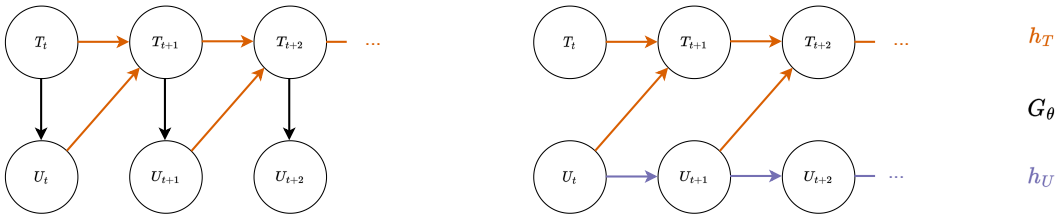

Figure 22: Chapter 4 (left) and Chapter 3 (right) modeling. Whereas we estimate the hidden state U_t from observations T_t at each time step in chapter 4, we learn the dynamics h_U to predict U at each time step in chapter 3. This prevents the error on T to propagate.

To constrain our model to learn the dynamics of the velocity U, we propose to combine both methods from chapters 3 and 4. First, we recall the regularization proposed in chapter 3. Then, we propose a new training objective coupling the optimization of the second bound in chapter 4 and the regularization from chapter 3.

Model Reminders

In chapter 3, we propose to learn the whole dynamics of the state $X = (T, U)$, following the objective defined in eq. (23):

$$\min_{G_\theta, h_U, h_T^d} \left\| \hat{U}_{t_f} - \left(\hat{U}_{t_0} + \int_{t_0}^{t_f} h_U(X_t)\, \mathrm{d}t \right) \right\|_2 \text{ subject to } \frac{\mathrm{d}T_t}{\mathrm{d}t} = (h_T^p + h_T^d)(\hat{X}_t)$$

where G_θ is an inverse model so that $\hat{U}_t = G_\theta(T_{t-k:t})$, and $\hat{X}_t = (T_t, \hat{U}_t)$. h_U is a dynamical model used to constrain U to follow an ODE. This formulation amounts to simulate both dynamics f_T and f_U.

Model Adaptation

To constrain the dynamics of U, we propose to learn both dynamics associated to T and U, while retaining the constraints associated with eq. (48). This amounts to combine both methods from chapters 3 and 4. We learn the dynamics of U with a neural network, and use the predictions of U to predict T (see fig. 22). Contrary to chapter 3, we do not enforce the dynamics of U using an inverse model. Instead, we use the constraints derived in chapter 4 using the observations θ_p^{pr}. We thus propose to constrain the dynamics associated to both T and U using the following objective:

$$\min_{h_T \in \mathcal{S}_p + \mathcal{S}_d, h_U} \left\| \theta_{t_f}^{p,pr} - \left(\hat{U}_{t_0} + \int_{t_0}^{t_f} h_U(X_t)\, \mathrm{d}t \right) \right\|_2 \text{ subject to } \frac{\mathrm{d}T_t}{\mathrm{d}t} = (h_T^p + h_T^d)(\hat{X}_t) \tag{49}$$

To solve eq. (49), we introduce the same constraints as in chapter 4, i.e. $\delta(h_T^d, 0)$, $\delta(h_T^p, f_T^{p,pr})$, $\delta(h_T, f_T)$. The difference with chapter 4 comes from the use of a neural network to model h_U (see fig. 23).

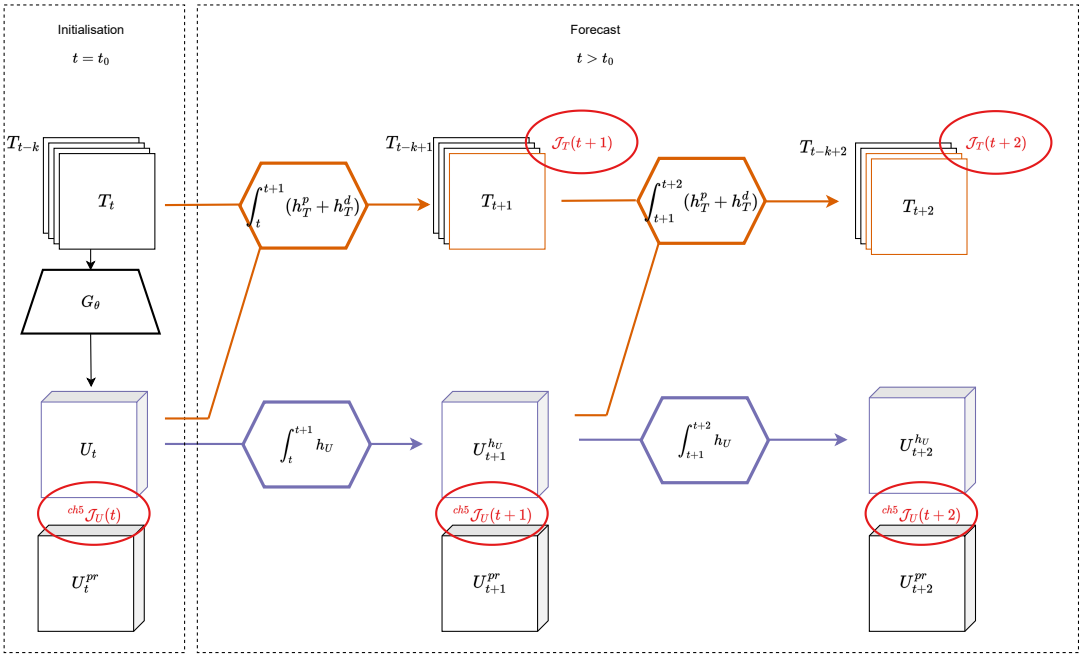

Figure 23: Computational graph of the method proposed in this chapter on two time steps at training time. In this case, $^{ch5}\mathcal{J}_U(t)$ is defined in eq. (50). At forecast time, i.e. at time $t > t_0$, the evolution of $U_t^{h_U}$ as modeled with an ODE is constrained to coincide with U_t^{pr}. Contrary to chapter 3, $\mathcal{J}_T$ also encompasses a loss term on h_T^d. Forecasting amounts to learning h_T and h_U from a sequence of observations of T. At each timestep t, we predict T and U at the next timestep $t+1$. The forcing term F are modeled with h_T^d and learned from a sequence of T. Note that $t-k$ actually refers to the timestep $t-k\Delta t$

The optimization of eq. (49) consists in learning h_T^p, h_T^d and h_U by minimizing the overall cost function $\mathcal{J}$ defined by

$$\mathcal{J} = \sum_{i=1}^{n}\sum_{j=0}^{k}(\mathcal{J}_U(t_0 + j\Delta t) + \lambda_T \mathcal{J}_T(t_0 + j\Delta t)) \tag{50}$$

$$\text{with } \mathcal{J}_T(t) = \lambda_{h_T}\left\|\text{ODESolve}(\hat{X}_{t_0}^i, h_T, t) - T_t^i\right\|_2 + \lambda_{h_T^d}\left\|\text{ODESolve}(\hat{X}_{t_0}^i, h_T^d, t)\right\|_2$$

$$\mathcal{J}_U(t) = \lambda_{h_T}\left\|\text{ODESolve}(\hat{X}_{t_0}^i, h_T, t) - T_t^i\right\|_2 + \lambda_{h_U}\left\|\text{ODESolve}(\hat{X}_{t_0}^i, h_U, t) - \theta_t^{i,p,pr}\right\|_2$$

where $\lambda_{h_T}, \lambda_{h_U}, \lambda_{h_T^d}$ are hyperparameters, n is the number of initial conditions in the training set and k is an hyperparameter. We alternate the optimization of the parameters θ_p and θ_d of (h_T^p, h_U) and h_T^d.

The computational graphs from Figures 23 and 24 give a schematic representation of our learning scheme over two time steps respectively at training and inference time. We also recall the computational graphs Figures 25 and 26 respectively from chapters 3 and 4. The method proposed here is a combination of both approaches: as in chapter 3, we use h_U to constrain U to follow an ODE, while enforcing our model with the loss proposed in chapter 4.

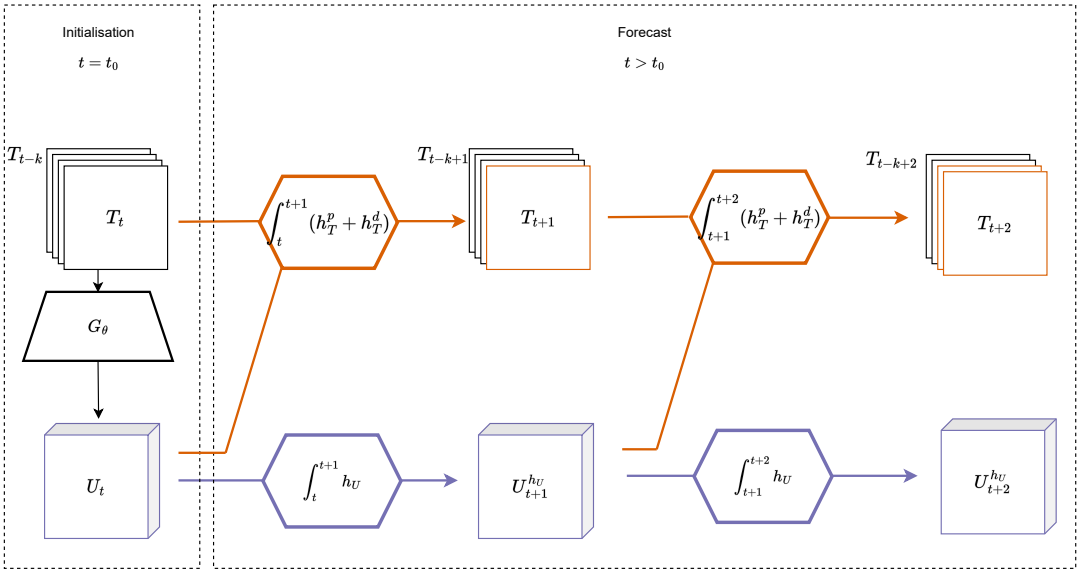

Figure 24: Computational graph of the method proposed in this chapter on two time steps at inference time. Forecasting amounts to learning h_T and h_U from a sequence of observations of T. At each timestep t, we predict T and U at the next timestep $t+1$. Note that the inverse model G_θ is only used for the initialisation, i.e. at time $t = t_0$.

Table 5: Ablation Study for the NATL60 data: comparison between estimations made from eq. (48) and eq. (49). In both cases, the $\star$ denotes that we use information about latitide/longitude and CFL condition. We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the forcing term F over 3 time steps on test set.

Models	T	U	F
Ours eq. (48) $\star$	6.86 (0.12)	6.81 (0.07)	**4.35 (0.11)**
Ours eq. (49) $\star$	**6.23(0.21)**	**6.1(0.2)**	4.42(0.09)
Aphynity	8.18 (0.16)	11.75 (0.49)	6.02 (0.02)
NeuralODE	8.83 (0.98)	n/a	n/a

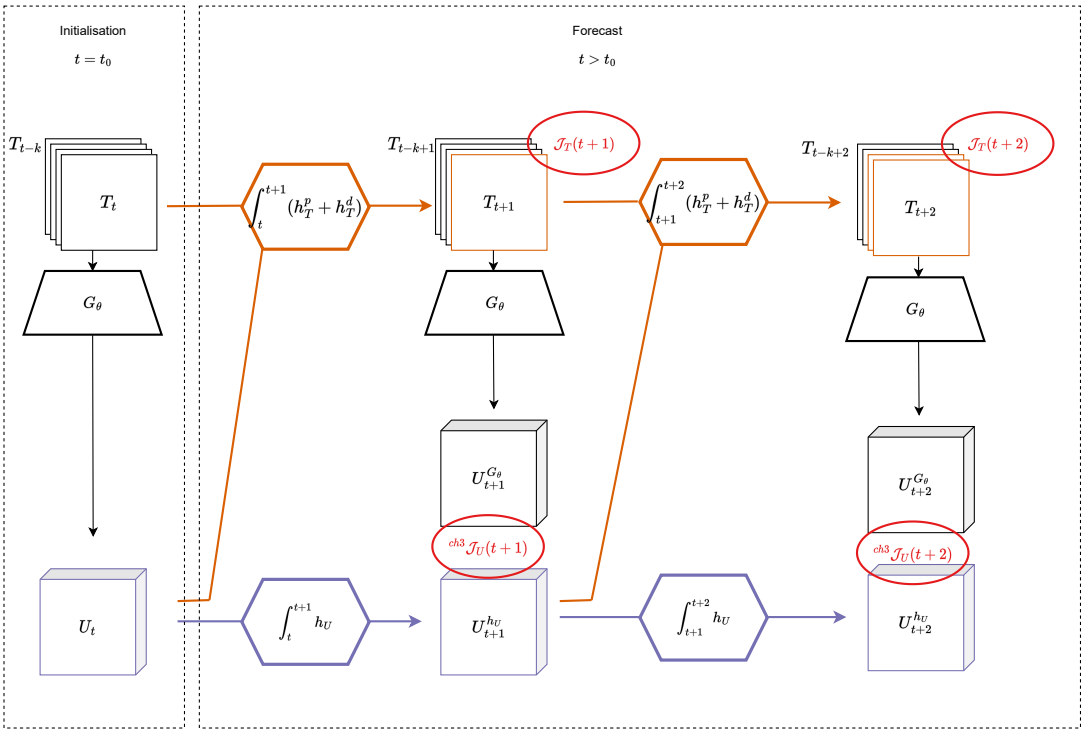

Figure 25: Computational graph of the method proposed in chapter 3 on two time steps. At forecast time, i.e. at time $t > t_0$, the evolution of $U_t^{h_U}$ as modeled with an ODE is constrained to coincide with $U_t^{G_\theta}$ as estimated with G_θ. In this case, ${}^{ch3}\mathcal{J}_U(t)$ uses the estimation from G_θ to constrain h_U.

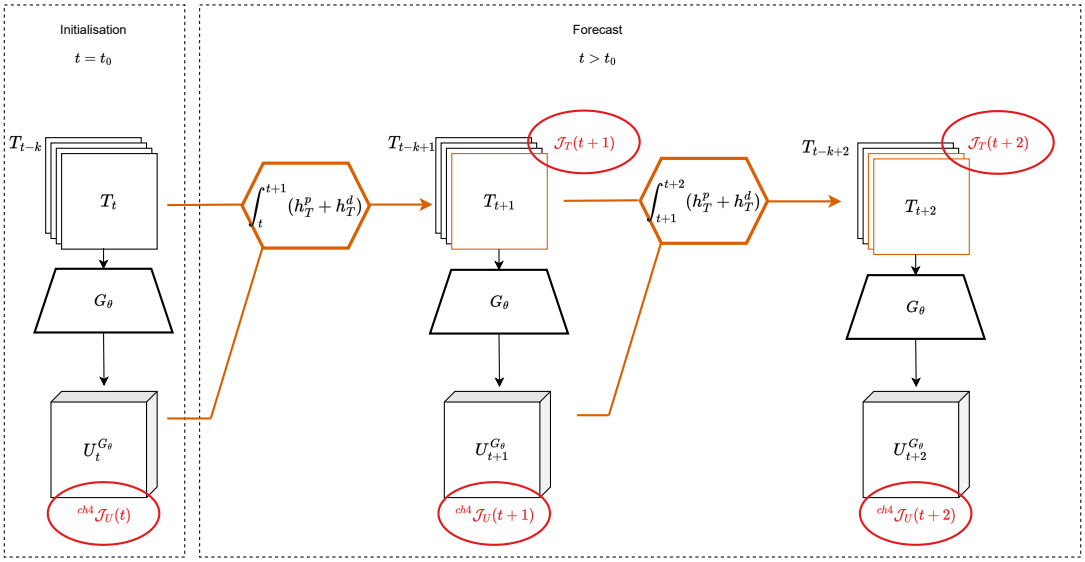

Figure 26: Computational graph of the method proposed in chapter 4 on two time steps. At each timestep t, we estimate U_t using G_θ. In this case, ${}^{ch4}\mathcal{J}_U(t)$ is computed between T and the estimation of h_T^p.

Table 5 and fig. 27 show the relevance of the introduction of the dynamical regularization. Note that the physical information about latitude/longitude and CFL is retained in the experiments. However, despite the improvement in results, a number of pitfalls persist. Notably, we observe a strong overfitting to T, i.e. the model focus on the modeling of T and bypass h_T^p. In the following section, we discuss these effects and list experiments designed to combat them.

Training details

ARCHITECTURE DETAILS The architectures in this setting are identical to the ones described in chapter 4, section 4.4.1.

OPTIMIZATION We use Adam optimizer with learning rate 0.00001 for 50 epochs with batch size 32. We enforce δ over 6 time-steps, i.e we supervise the predictions on timesteps: $(t_0 + \Delta t, \dots, t_0 + 6\Delta t)$. We use dropout in both G_θ and h_T^d.

HYPERPARAMETERS, SETTING OF EQ. (47) The selected model is the one with lowest prediction errors on validation set (i.e lowest $\delta(h_T, f_T)$), sampling uniformly the hyperparameters: $\lambda \sim \mathcal{U}(1, 0.1, \dots, 10^{-4})$. λ_{h_T} geometrically increases from 0.01 up to 100. We initialize $\lambda_{h_T^p} = 0.1$ and decrease it geometrically down to $\lambda_{h_T^p} = 0.00001$. $\lambda_{h_T^d}$ is fixed through training at 0.1. We alternate projection on $\mathcal{S}_p$ and $\mathcal{S}_d$ by descending the gradient 10-batches on both h_T^p and h_T^d.

HYPERPARAMETERS, SETTING OF EQ. (48) AND EQ. (49) The selected model is the one with lowest $\delta(h_T, f_T) + \delta(h_T^p, f_T^{p,pr})$ error, sampling uniformly the hyperparameters: $\lambda \sim \mathcal{U}(1, 0.1, \dots, 10^{-4})$. Because the dynamics of NATL60 is highly non linear and chaotic, we follow Jia, Willard, Karpatne, Read, Zwart, Michael S Steinbach, et al. (2019) and first warm-up the parameters recognition

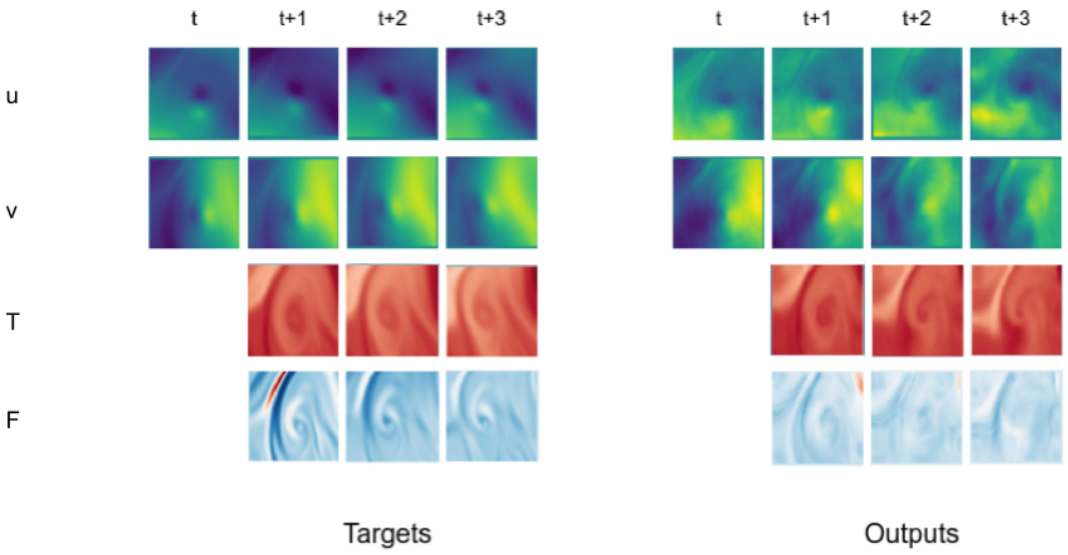

Figure 27: Sequence of prediction on T, u, v, F for the NATL60 data across 3 days trained according to eq. (49). From the test set. On the left, true variables. On the right, outputs of our model designed from eq. (49).

network G_θ on the velocity fields proxies for 10 epochs. For this setting, λ_{h_T} geometrically increase from 0.01 up to 1. $\lambda_{h_T^p}$ is set equal to λ_{h_T}. $\lambda_{h_T^d}$ is fixed through training at 0.01. After warm-up, we alternate projection on $\mathcal{S}_p$ and $\mathcal{S}_d$ by descending the gradient 100-batches on h_T^p and 300 on h_T^d.

BASELINES For the training of Aphinity (Yin, Le Guen, et al., 2021), we set the learning rate at 0.0001 and train on 30 epochs. We initialize $\lambda_{h_T} = 0.01$ and increase it geometrically every epoch up to $\lambda_{h_T} = 100$. $\lambda_{h_T^d}$ is fixed through training at 0.1. For the training of NODE (R. T. Q. Chen et al., 2018), we set the learning rate at 0.00004 and train on 50 epochs. To perform prediction, we first encode the 4-consecutive measurements of T (as a $3 \times 64 \times 64$ state) then learn to integrate this state in time thanks to a 3-layer convolutional networks, with 64 hidden channels. It is integrated using RK4 scheme available from `https://github.com/rtqichen/torchdiffeq`.

5.3.3 *Analysis*

Neither conventional ML methods nor the addition of physical knowledge have proven effective for the complex NATL60 dataset. We now discuss several aspects that may be limiting the learning on NATL60 dataset. First, we underline the gap between theory and practice. Second, we list several attempts made to overcome the observed limitations of our model.

Regarding the reality gap

Even if we rely on a theory based on proofs of convergence, in practice the assumptions made are not respected. For example, for the optimization problem eq. (33) to be well-posed, both the relative compactness of $\mathcal{H}_p$ and the convexity of d (as defined in eq. (20)) are necessary. Moreover, to ensure the convergence of algorithm 1, both $\mathcal{H}_p$ and $\mathcal{H}_d$ should be convex and d should be strongly convex. However, in reality, there is no reason for $\mathcal{H}_p$ to be convex. Besides, we use the loss δ defined in eq. (21) which is not convex either.

More than a gap between theory and practice, there is also a gap between the Adv+Source dataset used in part ii and the NATL60 data. For instance, h_T^d accounts for more phenomena, as here F not only accounts for ocean-atmosphere heat exchanges but also for interactions with deep ocean layers. Besides, whereas the velocity fields generated following eq. (16) are periodic, those from NATL60 are not, which makes the learning more complicated. Finally, whereas the Adv+Source data accounts for 10 different velocities, in NATL60 there are as many velocity fields as zones, i.e. 320 different fields. This touches on problems of generalisation, which we address in section 5.4.

The fact that the model learns poorly on NATL60 data is evidenced through three observations: 1. U is smoothed out as learning progresses; 2. the model strongly overfits T; 3. we observe a high sensitivity to hyperparameters. In the following, we give insights on reasons for such limitations and propose solutions to overcome them.

Experimental Trials

The failure of our model to learn on NATL60 data can be attributed to several reasons. The overfitting may be due to a lack of data or an inadequate physical model. Our hybrid model assumption may also be questioned. We present below several tests that have been carried out to improve the results.

FIGHTING SMOOTHING The smoothing probably comes from the use of the MSE loss function to learn U. Indeed, the MSE is known to lead to blurry predictions (Mathieu, Couprie, and LeCun, 2015). We attempted to retropropagate using the L1 loss function and the PSNR, even though it is usually used as metrics. It did not led to an improvement in results.

FIGHTING OVERFITTING To predict SST, the model ignores U. That means that no physics is learned, i.e. G_θ does not learn the relationship between T and U. Classical solutions have been tried, such as weight decay or dropout on both G_θ and h_T^d. Note that increasing the batch size was impossible, due to the high dimensionality of the images and the limit in GPU memory. It has not led to an improvement in results.

PHYSICAL STANDPOINT To reduce overfitting, another intuition would be to incorporate more physical knowledge into the model. One wonders which information, and how. We are trying to learn two phenomena from observations of T: the velocity fields U and the forcing term F. Whereas the link between T and U is evidenced in (Bigg and Killworth, 1988), to learn a source term from the sole SST seems fairly ambitious. Indeed, the source terms mainly depend on the environment: for instance, if the atmosphere is warmer than the ocean, the exchanges will go from the atmosphere to the ocean, which is the case at the equator but not at the poles. Thus, providing geographic information such as latitude and longitude associated to each region could help the model.

A first attempt was made by providing to h_T^d part of the forcing term: those are divided into the radiative part (i.e. the solar flow) and the turbulent part (i.e. from evaporation and conduction at the surface). We tried to inform h_T^d with the radiative fluxes and let it learn the turbulent part.

A second attempt was made to provide h_T^d with information about the area. The training sequences have been reorganised to include spatio temporal information, so that each sequence corresponds to a season and a zone. This information was provided to h_T^d as a code corresponding to an area and to a season. Another attempt was made to provide latitude and longitude to G_θ.

The model as we have designed it does not work on simulations of reality. Several made attempts do not enhance the results. In the next section, we propose some perspectives for modifying our model in order to solve the aforementioned issues.

5.4 PERSPECTIVES

In this last section, prospects for overcoming the identified issues are proposed. Notably, we tackle the smoothing of U and the generalization across regions not seen through training. We propose several model reformulation, focusing on the learning of both h_T^p and h_T^d. The changes we propose are based on both the network architectures and the optimization framework.

5.4.1 *On the learning of h_T^d*

It has been underlined that it seems almost impossible for h_T^d to learn a forcing term from the sole SST. To give spatial information seems more relevant. We distinguish two solutions to meet this condition.

USING IMPLICIT NEURAL REPRESENTATIONS One could take as input to h_T^d the coordinates where S is to be estimated. This is reminiscent of mesh-agnostic approaches for solving PDEs such as (Raissi, Perdikaris, and G. E. Karniadakis, 2019; Sirignano and Spiliopoulos, 2018). Recently, Sitzmann et al. (2020) coined such approaches as implicit neural representations (INR). Besides, one could imagine giving temporal information, such as the month, as additional input to h_T^d.

DOMAIN GENERALIZATION Instead of providing spatial information, this one could be directly learned by the network. Drawing from domain generalization (J. Wang et al., 2022) and especially generalization in dynamical systems (Kirchmeyer et al., 2022), one could rely on a spatiotemporal code concentrating information on one domain. This could be learned from a sequence of SST, and h_T^d could take as input a spatiotemporal code instead of the SST. Remaining questions are how to constrain such code and how to perform adaptation at test time.

5.4.2 *On the learning of h_T^p*

The main issues encountered when learning U are smoothing and overfitting. We distinguish three propositions to overcome them.

INITIALIZATION Global surface currents velocity fields are nowadays inferred from satellite observations of sea surface heights (SSH). Even though those estimations have a low spatiotemporal

resolution, they could be used as inputs to G_θ, which would then perform a super resolution task.

ADVERSARIAL LEARNING OF U To solve the smoothing of U inherent to the use of the MSE, adversarial learning could be of great help. As high resolution of U is not accessible in a real life setting, one could consider an unsupervised learning of the velocity fields, for instance using a conditional GAN (Mirza and Osindero, 2014). This would use a sequence of SST as condition, and use the velocity fields modeled with NATL60 for the training. One would thus learn the probability distribution of interest $p_{U|T}$. Notably, one could take from (Isola et al., 2017b) and enforce L2 constraint on the SST, i.e. on the outputs of the generator.

HYBRID MODELING OF f_U As for f_T, the dynamics f_U encompasses several phenomena, some of which are well known. To better constrain the learning of U, one could also encode h_U as a hybrid model $h_U = h_U^p + h_U^d$, where h_U^p would account for a known process, and h_U^d would account for a process learned from data. Recall that f_U account for the dynamics

$$\frac{\partial U}{\partial t} = -(U.\nabla)U - g'\nabla h + f \wedge U + D^U + F^U \tag{51}$$

For instance, in eq. (51), h_U^p could amount to the advection term and the derivative of the SSH, i.e. $h_U^p(U) = (U.\nabla)U - g'\nabla h$. h_U^d would then encompass for the Coriolis force effect and the forcings, i.e. $h_U^d = f \wedge U + D^U + F^U$. As developed above regarding h_T^d, one should carefully care for the inputs of h_U^d, which could be modeled by an INR. Note that the vertical component of the velocity should be taken into account when computing the advection. Otherwise, this would be reflected in the term h_U^d, which would then lose some of its physical meaning.

Part IV

CONCLUSION

6 CONCLUSION

To conclude, in section 6.1 we first review the main contributions of this thesis. Then, in section 6.2, we confront ML with real world problems and provide insights on major hazards that reseacrh in ML could face in the coming decades.

6.1 HYBRID MODELING OF OCEAN DYNAMICS

In this thesis, we took an interest in the current research path aiming at solving real-world physical problems using deep learning. To that end, we chose to study the oceanic system to analyze the practical aspect of our research. More specifically, we consider deep learning algorithms to model ocean currents directly from SST observations. To exploit the long gained physical knowledge and guarantee the physical consistency of our estimations, we use hybrid models, combining both a data-driven and a physical model. However, the underlying learning is inherently ill-posed: the high versatility of neural networks may lead the data-driven part to bypass the physical part.

As a first step, we have conducted a theoretical study on the learning of hybrid models and their well-posedness. To that end, we worked on a synthetic dataset which is a simplified representation of ocean dynamics. This led to two contributions. First, we proposed a dynamical regularization constraining the estimated velocity fields U to follow an ODE. Second, we proposed a well-posed framework for the learning of hybrid models, relying on the optimization of an upper-bound of the original ill-posed loss.

As a second step, we conducted an experimental investigation, during which we confronted our framework to real-like ocean observations of SST and velocity fields from the Gulf Stream current in the North Atlantic (from the NATL60 model). However, this did not lead to the expected results. This can be explained in several ways: on the one hand the assumptions made in theory are not valid in practice, on the other hand there is a gap between synthetic and real data. Indeed, the NATL60 data forcing term synthesises more phenomena than that generated for the synthetic Adv+F dataset. It includes vertical advection, lower ocean forcing and air-sea exchange. Besides, velocity vectors have more complex, finer-scale and non-periodic structures than those generated synthetically. Such gaps lead to difficulties through training, which is evidenced with strong overfitting. Several attempts have been made to overcome this issue. We questioned the loss function, attempted to address overfitting using classical ML methods, and added physical knowledge. None of these attempts proved effective.

Still, there are some solutions that are relatively easy to implement that would quickly solve some of the problems. Firstly, training on more than one year of data would probably reduce overfitting. In addition, we had access to only one observation per day, which did not allow us to predict scales as fine as those in the NATL60 dataset, and led to an explosion in prediction error. Secondly, the 2D advection scheme we use could be replaced by a 3D scheme, better adapted to the phenomenon modelled by NATL60. This would also make the forcing term more interpretable.

In the short term, adaptations of our model could facilitate learning: for example, using a hybrid model to learn the dynamics associated with ocean surface velocity fields U, or using an INR

to associate the forcing with a location. In the long term, attention could be given to mesh free modeling, domain generalization and unsupervised learning of U.

6.2 ML FOR PHYSICS: REAL WORLD PROBLEMATICS

This thesis aimed at exploring how deep learning models translate in real settings. Among the many issues related to ocean modelling, one of the main ones is the study of the impacts of climate change. I am convinced that climate change is probably the most important issue of our time.

ML FOR CLIMATE CHANGE ? Climate change refers to long-term shifts in temperatures and weather patterns, driven by human activities. The burning of fossil fuels like coal, oil and gas generate greenhouse gas (GHG) emissions, for instance carbon dioxide and methane. Those absorb solar energy, trapping heat within the atmosphere (Romm, 2022). This induce both near and long term consequences, including hotter temperatures, more severe storms, increased drought, a warming and rising ocean, loss of species, food shortage, health risks, poverty and displacement (Pörtner et al., 2022). Unless we rapidly reduce GHG emissions, Masson-Delmotte et al. (2018) shows that we head towards a rise in temperatures of 4°C by 2100. Under such warming, the sea level would rise of nearly 9 meters, puting approximately 700 million people at risk, that is 10 times the France population. To deal with these threats, climate change adaptation and mitigation is to be considered in our everyday activities, including our research.

If applied to real-life data, ML could be useful in combating climate change. For example, short-term forecasting of local risks (such as floods or fires) is a major challenge in terms of adaptation. However, current models do not allow for such forecasts. It could also be useful in quantifying the uncertainties inherent in current models. Recently, Rolnick et al. (2022) describe how ML can be a powerful tool in reducing greenhouse gas emissions and helping society adapt to a changing climate. Their paper lists areas where ML could be of help such as electric systems, transportation, agriculture and forest preservation but also climate prediction and solar geoengineering. Before rushing into techno-solutionism, let's refocus our attention on the many limitations that ML will have to overcome before it appears as a viable solution.

TOWARDS THE LOW-TECH ERA? Firstly, ML has a significant carbon cost, mainly due to the infrastructure needed to operate it but also to the training of models. Accurate reporting of energy and carbon consumption is essential to understanding the potential climate impacts of ML research. Henderson et al. (2020) proposes a framework that makes this easier by providing a simple interface for tracking realtime energy consumption and carbon emissions.

Secondly, we must be wary of extractivism. Bihouix (2019) warns us of resource scarcity: metal consumption in the information and communication technologies sector tripled between 1980 and 2010 (Bihouix and De Guillebon, 2021). Notably, supply with rare earth elements is likely to be disrupted in the near future (De Boer and Lammertsma, 2013). To avoid reaching dearth, we should regulate our consumption of resources and adopt more economical practices (Vidal, 2018).

Lastly, we can ask ourselves about the future we are heading towards. Whereas the physical paradigm focuses on understanding the physical laws behind a phenomenon, the data paradigm relies on a vision that can be described as more materialistic, refraining from any understanding and relying only on the accumulation of data to model a phenomenon. In the long term, this

paradigm shift raises philosophical questions: are we ready to abandon the goal of understanding our world and entrust it to machines? Moreover, climate change and planetary limits will sooner or later push us towards a technically sustainable civilisation, a society based on low-technology systems (Bihouix, 2014).

Part V

APPENDIX

A

SUPPLEMENTARY MATERIAL FOR CHAPTER 2

A.1 SEMI LAGRANGIAN SCHEME

A semi-Lagrangian scheme is a numerical method for solving PDEs. It is mainly used to describe physical systems with advective behaviour, such as fluid flows. For cases where the fluid velocity is much larger than the diffusive and dissipative effects, traditional finite difference or finite element methods can produce numerical errors and spurious oscillations, but the semi-Lagrangian method is more stable and accurate. It is therefore widely used in computational fluid dynamics, especially in meteorology and atmospheric modelling. In this method, the solution is approximated by tracking fluid elements along their characteristic curves and updating the solution at each time step.

CHARACTERISTIC CURVE The characteristic curve of a fluid element is the path followed by that element in a fluid as it moves over time. In other words, it is the trajectory of the fluid particle in space and time. In a fluid flow, the velocity field determines the movement of fluid elements, and the characteristic curve of a fluid element is obtained by integrating the velocity field along the particle's path. The characteristic curve can be thought of as a mapping from the particle's initial position to its position at some future time.

The characteristic curve is a key concept in the semi-Lagrangian scheme. In this method, the solution is updated by integrating the PDE along the characteristic curves of the fluid elements, and the solution at each time step is approximated by interpolating the values at the particle locations.

THE SEMI-LAGRANGIAN METHOD The semi-Lagrangian method starts by discretizing the spatial domain into a fixed grid. At each time step, the solution is updated by integrating the PDE along the characteristic curves of the fluid, which are the paths followed by fluid elements (fig. 28). The solution at the new time step is then interpolated onto the grid for storage and post-processing. The semi-Lagrangian scheme is thus computationally efficient, as it only requires the evaluation of the solution at the particle locations.

In our case, consider a solution to the advection equation:

$$\frac{\partial T}{\partial t} = -\nabla \cdot (TU) + F$$

with $F = 0$. The method of characteristics consists in exhibiting curves $(x(s), t(s))$ along which the derivative of the solution T is simple, i.e $\dfrac{\partial T}{\partial s}(x(s), t(s)) = 0$. For a 1D constant advection scheme, computations lead to:

$$\frac{\mathrm{d}t}{\mathrm{d}s} = 1 \implies s = t$$
$$\frac{\mathrm{d}x}{\mathrm{d}s} = U \implies x = x_0 + Ut$$

giving therefore, $T(x,t) = T_0(x - Ut)$, linking the value of the solution at all time to its initial condition. Therefore from a single observation at t_0, it suffices to estimate the original departure

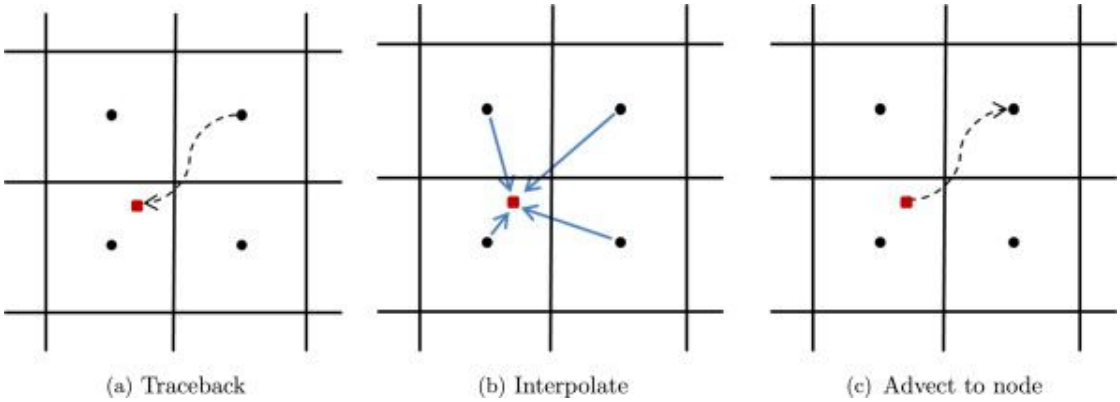

Figure 28: Basic steps involved in semi-Lagrangian transport. Imagine a fluid flowing over a surface and the velocity field at each point in the fluid is known. The semi-Lagrangian scheme would track the movement of fluid elements along the velocity field to calculate their position at a future time. This process is repeated at each time step to simulate the evolution of the fluid flow over time. For instance, on the left, the red particle figures $T(x - Ut)$ with known value. The black particle on the top right figures $T(x, t)$, which estimate from $T(x - Ut)$. From Verma, Xuan, and Blanquart (2014).

points $x_0 - Ut$ to infer the prediction at t.
However, when U is not constant in time, the method remains doable, not along characteristic *lines* defined by : $(x_0 + Ut)$, but along characteristic *curves* which are given by:

$$\begin{aligned} \frac{\mathrm{d}t}{\mathrm{d}s} &= 1 \implies s = t \\ \frac{\mathrm{d}x}{\mathrm{d}s} &= U(x, t) \end{aligned} \tag{52}$$

SEMI-LAGRANGIAN AND FULLY LAGRANGIAN SCHEMES Semi-Lagrangian and fully Lagrangian schemes are both numerical methods for solving PDEs, but they differ in the way they approximate the solution. A semi-Lagrangian scheme uses a combination of Lagrangian, i.e. particle-based and Eulerian, i.e. grid-based techniques. In this method, fluid particles are tracked along their characteristic curves and the derivatives of the solution are approximated at each time step. The resulting solution is then interpolated onto a fixed grid for storage and post-processing. A fully Lagrangian scheme, on the other hand, is a purely particle-based method in which the solution is represented by a set of discrete particles that move and deform over time to approximate the evolution of the fluid. In this method, there is no fixed grid, and the solution is only defined at the particle locations.

A.2 DISTANCE

A.2.1 *Distance Between Dynamics*

We here give the definition of the distance d. Let u and v be two functions of $\mathcal{L}^2(\mathbb{R}^p, \mathbb{R}^p)$. We consider the distance:

$$\mathrm{d}(u, v) = \mathbb{E}_{X \sim p_X} \left\| u(X) - v(X) \right\|_2 \tag{53}$$

Naturally, eq. (53) verifies the triangle inequality, the symmetry and the positiveness. Moreover, in this case, for all functions f, $\mathrm{d}(., f)$ is convex. Indeed, for u, v two functions, and $\lambda \in [0, 1]$:

$$\begin{aligned}
\mathrm{d}(\lambda u + (1-\lambda)v, f) &= \mathbb{E}_{X\sim p_X} \left\| \lambda u(X) + (1-\lambda)v(X) - f(X) \right\|_2 \\
&= \mathbb{E}_{X\sim p_X} \left\| \lambda u(X) - \lambda f(X) - (1-\lambda)f(X) + (1-\lambda)v(X) \right\|_2 \\
&\leq \lambda \mathbb{E}_{X\sim p_X} \left\| u(X) - f(X) \right\|_2 + (1-\lambda)\mathbb{E}_{X\sim p_X} \left\| v(X) - f(X) \right\|_2
\end{aligned}$$

Hence the convexity of $\mathrm{d}(., f)$. This consideration suffices to ensure the convexity of $\mathcal{S}_p$ and $\mathcal{S}_d$ defined in chapter 4, section 4.2.

A.2.2 *Distance Between Flows*

With the definition of eq. (6), we can define the distance between two flows of ODE as:

$$\delta(\phi_u, \phi_f) = \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \left\| \phi_u(t, X_0) - \phi_f(t, X_0) \right\| \mathrm{d}t \tag{54}$$

δ is positive and symmetric. Let ϕ_u, ϕ_v be two flows, we have the triangle inequality:

$$\begin{aligned}
\delta(\phi_u, \phi_f) &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \left\| \phi_u(t, X_0) - \phi_f(t, X_0) \right\| \mathrm{d}t \\
&= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \left\| \phi_u(t, X_0) - \phi_v(t, X_0) + \phi_v(t, X_0) + \phi_f(t, X_0) \right\| \mathrm{d}t \\
&\leq \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \left\| \phi_u(t, X_0) - \phi_v(t, X_0) \right\| + \left\| \phi_v(t, X_0) + \phi_f(t, X_0) \right\| \mathrm{d}t \\
&\leq \delta(\phi_v, \phi_v) + \delta(\phi_v, \phi_f)
\end{aligned}$$

Let ϕ_f be fixed, we also have the convexity of $\delta(., \phi_f)$ with respect to the first argument. Indeed for $\lambda \in [0, 1]$:

$$\begin{aligned}
\delta(\lambda\phi_u + (1-\lambda)\phi_v, f) &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \left\| \lambda\phi_u(t, X_0) + (1-\lambda)\phi_v - \phi_f(t, X_0) \right\| \mathrm{d}t \\
&= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \left\| \lambda\phi_u(t, X_0) + (1-\lambda)\phi_v - \lambda\phi_f(t, X_0) - (1-\lambda)\phi_f(t, X_0) \right\| \mathrm{d}t \\
&\leq \lambda\, \delta(\phi_u, \phi_v) + (1-\lambda)\delta(\phi_v, \phi_f)
\end{aligned}$$

However, in this case the convexity is not ensured with respect to u and v.This is the reason why for theoretical investigations, we consider the distance d instead of δ. Nonetheless, $\delta(\phi_u, \phi_f) = 0 \implies \phi_u = \phi_f \implies u = f$.

B

SUPPLEMENTARY MATERIAL FOR CHAPTER 3

B.1 PROOF THAT U FOLLOWS AN ODE

With

$$f(x,y,t) = y - B(t)\cos(kx) \tag{55}$$

$$g(x,t) = \sqrt{1 + k^2 B(t)^2 sin^2(kx)} \tag{56}$$

Let

$$\mathcal{H}(x,y,t) = -\tanh\left(\frac{f(x,y,t)}{g(x,t)}\right) + cy \tag{57}$$

We denote $\mathcal{H}_x = \frac{\partial \mathcal{H}}{\partial x}$, $f_x = \frac{\partial f}{\partial x}$, $f_t = \frac{\partial f}{\partial t}$,...

We have $U = (u,v) = \left(\frac{\partial \mathcal{H}}{\partial y}, -\frac{\partial \mathcal{H}}{\partial x}\right) = (\mathcal{H}_y, -\mathcal{H}_x)$.

Let's calculate

$$\mathcal{H}_x = \frac{f_x g - f g_x}{g^2}\left(\tanh^2(\frac{f}{g}) - 1\right) = -v \tag{58}$$

$$\mathcal{H}_y = \frac{f_y}{g}\left(\tanh^2(\frac{f}{g}) - 1\right) + c = \frac{1}{g}\left(\tanh^2(\frac{f}{g}) - 1\right) + c = u \tag{59}$$

We wonder whether U follows an ODE, i.e. there is a function l such that $\frac{\mathrm{d}U}{\mathrm{d}t} = l(U)$. We have $\frac{\mathrm{d}U}{\mathrm{d}t} = \left(\frac{\mathrm{d}u}{\mathrm{d}t}, \frac{\mathrm{d}v}{\mathrm{d}t}\right)$. We calculate

$$\begin{aligned}
\frac{\mathrm{d}u}{\mathrm{d}t} &= -\frac{g_t}{g^2}\left(\tanh^2(\frac{f}{g}) - 1\right) + \frac{1}{g}\frac{f_t g - f g_t}{g^2} 2\tanh(\frac{f}{g})(1 - \tanh^2(\frac{f}{g})) \\
&= -\frac{1}{g}\left(\tanh^2(\frac{f}{g}) - 1\right)\left(\frac{g_t}{g} + \frac{f_t g - f g_t}{g^2} 2\tanh(\frac{f}{g})\right) \\
&= -u\left(\frac{g_t}{g} + \frac{f_t g - f g_t}{g^2} 2\tanh(\frac{f}{g})\right) + c\left(\frac{g_t}{g} + \frac{f_t g - f g_t}{g^2} 2\tanh(\frac{f}{g})\right)
\end{aligned}$$

With $a = \frac{f_x g - f g_x}{g^2}$, we calculate

$$\begin{aligned}
\frac{\mathrm{d}v}{\mathrm{d}t} &= a_t\left(\tanh^2(\frac{f}{g}) - 1\right) + a\frac{f_t g - f g_t}{g^2} 2\tanh(\frac{f}{g})(1 - \tanh^2(\frac{f}{g})) \\
&= -v\frac{f_t g - f g_t}{g^2} 2\tanh(\frac{f}{g}) + a_t\left(\tanh^2(\frac{f}{g}) - 1\right)
\end{aligned}$$

Hence, U as defined in section 2.2.3 follows an ODE, which justifies the constraint from section 3.1.

C

SUPPLEMENTARY MATERIAL FOR CHAPTER 4

C.1 PROOFS

C.1.1 *ODE Identification*

Consider the following set: $\mathcal{S}_A = \{X(t) \in \mathcal{C}^1([0,T],\mathbb{R}^p) \text{ such that: } \exists A \in \mathcal{M}_{p,p}(\mathbb{R}), X' = AX\}$, where $T > 0$.
$\mathcal{S}_A$ is not a convex set. Consider u and v in $\mathcal{S}_A$, and consider A^u and A^v so that $u'(t) = A^u u(t)$ and $v'(t) = A^v v(t)$. For $\lambda \in]0,1[$: we have:

$$\begin{aligned}[\lambda u + (1-\lambda)v]' &= \lambda u' + (1-\lambda)v' \\ &= \lambda A^u u + (1-\lambda)A^v v\end{aligned}$$

In general the last term is not equal to $A^{\lambda u+(1-\lambda)v}(\lambda u+(1-\lambda)v)$, for some matrix $A^{\lambda u+(1-\lambda)v}$. Thus $\mathcal{S}_A$ is not a convex set. However, discretizing the trajectories and employing a simple integration scheme leads to considering the following cost function:

$$\mathcal{L}(A) = \sum_t \|\Big(X^s(t+1) - (A\Delta t + Id)X^A(t)\Big)\|_2^2 \tag{60}$$

As a least square regression problem, $\mathcal{L}(A)$ is convex with respect to A. A least square regression setting can also be recovered using more complex integration schemes, or several time steps integration.

C.1.2 *Proof for the Well-posedness of Equation* (33)

We set ourselves in the Hilbert space of squared integrable functions with the canonical scalar product $\Big(\mathcal{L}^2(\mathbb{R}^p,\mathbb{R}^p), <,> \Big)$. For further consideration on such functional space we refer to (Droniou, 2001).
We assume that $\mathcal{H}_k$ hence $\mathcal{S}_p$ is convex and a relatively compact family of functions.

CONVEXITY OF $\mathcal{S}_p$ Let $u, v \in \mathcal{S}_p$:

$$\begin{aligned}\mathrm{d}(tu + (1-t)v, f) = \|tu + (1-t)v - f\| &= \|tu - tf + (1-t)v - (1-t)f\| \\ &\leq t\mu_1 + (1-t)\mu_1 = \mu_1\end{aligned}$$

Hence the convexity of $\mathcal{S}_p$.

CONVEXITY OF $\mathcal{S}_d$ Let $t \in [0,1]$ and $u, v \in \mathcal{S}_d$.

$$\begin{aligned}\mathrm{d}(h_k, h_k + tu + (1-t)v) &= \mathrm{d}(0, tu + (1-t)v) \\ &\leq t\mathrm{d}(u,0) + (1-t)\mathrm{d}(v,0) \\ &\leq \mu_2\end{aligned}$$

Hence the convexity of $\mathcal{S}_d$.

CONVEXITY OF $\mathcal{S}_p + \mathcal{S}_d$ Let $\mathcal{S} = \mathcal{S}_p + \mathcal{S}_d = \{f | \exists f_p \in \mathcal{S}_p, f_d \in \mathcal{S}_d, f = f_p + f_d\}$. Let $f, g \in \mathcal{S}$ and $\lambda \in]0, 1[$:

$$\lambda f + (1-\lambda)g = \lambda f_p + (1-\lambda)g_k + \lambda f_d + (1-\lambda)g_u \in \mathcal{S}_p + \mathcal{S}_d$$

Hence the convexity of $\mathcal{S}$.

Closeness of $\mathcal{S}_d$ We show that $\mathcal{S}_d$ is a closed set. Indeed, $\mathcal{S}_d = g^{-1}([0, \mu_u])$, where $g(u) = \|u\|$, Because g is 1-Lipschitz (using the triangle inequality), g is continuous. Therefore $\mathcal{S}_d$ is closed set as the inverse image of a closed set by continuous function.

SEQUENTIAL LIMIT We now show that $\mathcal{S}$ is a closed set thanks to the sequential characterisation: let f^n a converging sequence of elements of $\mathcal{S}$ and denote f its limit. We prove that f^n converges in $\mathcal{S}$.
Because $\forall n, f_n \in \mathcal{S}$, we have: $f^n = f_k^n + f_u^n$, where $f_u^n \in \mathcal{S}_d$ and $f_k^n \in \mathcal{S}_p$.
Thanks to the relative compactness of $\mathcal{S}_p$, we can extract a converging sub-sequence, of indexes n_j, from f_k^n so that $f_k^{n_j} \to f_p \in \mathcal{S}_p$.
Because $f^n \to f$, the sub-sequence f^{n_j} converges: $f^{n_j} \to f$.
By definition, $f_u^{n_j}$ is a sequence of $\mathcal{S}_d$ and we also have that: $f_u^{n_j} = f^{n_j} - f_k^{n_j}$. Because the right member of the equation converges (as a sum of converging functions), the left member of the equation converges i.e. $f_u^{n_j}$ converges.
Since $\mathcal{S}_d$ is a closed set $f_u^{n_j}$ converges in $\mathcal{S}_d$. We write f_d its limit. Therefore, $f_u^{n_j} = f^{n_j} - f_k^{n_j} \to f - f_p = f_d \in \mathcal{S}_d$. Hence, $f = f_d + f_p$ with $f_d \in \mathcal{S}_d$ and $f_p \in \mathcal{S}_p$.
Therefore $\mathcal{S}$ is a closed set.
Finally, we can apply Hilbert projection lemma on the closed convex set $\mathcal{S}$ and retrieve the uniqueness of the minimizer of eq. (33).

REMARK The relative compactness of a family of functions is a common assumption in functional analysis. For example, in the study of differential equation Cauchy-Peano theorem provides the existence to the solution of an ODE under the assumption of relative compactness. Also, Ascoli theorem provides the relative compactness of a family of function $\mathcal{F}$ under the hypothesis of the equi-continuity of $\mathcal{F}$ and the relative compactness of the image space $A(x) = \{f(x) | f \in \mathcal{F}\}$.

C.1.3 *Proof of Proposition 2*

We now set ourselves in the Hilbert space $\left(\mathcal{L}^2([0,T], \mathbb{R}^p), <,> \right)$ of squared integrable functions, where $<,>$ is the canonical scalar product of $\mathcal{L}^2([0,T], \mathbb{R}^p)$.

Proposition 4 (Existence and Uniqueness)**.** *If $\hat{A}$ is invertible, There exists a unique D_A, hence a unique X^D, solving eq. (42).*

Proof. Let A be a given invertible matrix. We consider the following space $\mathcal{S}_D = \{X \in \mathcal{C}^1([0,T], \mathbb{R}^p)$ such that: $\exists D \in \mathbb{R}^p, X' = AX + D$ and $X(t=0) = X_0\}$, where $T > 0$. We show that $\mathcal{S}_D$ is a closed convex set.

CONVEXITY Indeed, let $\lambda \in]0, 1[$ and $u, v \in \mathcal{S}_D$. $\lambda u + (1-\lambda)v$ is differentiable and:

$$[\lambda u + (1-\lambda)v)]' = \lambda u' + (1-\lambda)v' = A(\lambda u + (1-\lambda)v) + D,$$

Where $D = \lambda D_u + (1-\lambda)D_v$. Hence $\lambda u + (1-\lambda)v \in \mathcal{S}_D$.

CLOSENESS VIA AFFINE-SPACE To prove the closeness of $\mathcal{S}_D$, we prove that it is an affine space of finite dimension. Let g the application that to any vector $D \in \mathbb{R}^d$ associate the solution X^D. Let $D_0 \in \mathbb{R}^D$, we show that $g_{D_0} : D \to g(D_0 + D) - g(D_0)$ is a linear application. Naturally, for $g_{D_0}(0_{\mathbb{R}^p}) = 0_{\mathcal{L}^2}$. Then for $D \neq 0_{\mathbb{R}^p}$ we have:

$$\begin{aligned} g_{D_0}(D) &= e^{At}(X_0 + A^{-1}(D_0 + D)) - A^{-1}(D_0 + D) - e^{At}(X_0 + A^{-1}(D_0) + A^{-1}D_0 \\ &= e^{At}A^{-1}D \end{aligned}$$

Therefore g_{D_0} is a linear function and g is an affine function. Moreover, g is an injection. Indeed, if two functions are equals, then they have at most one inverse image by g thanks to Cauchy-Lipschitz theorem. Therefore it defines a bijection of $\mathbb{R}^d$ in $g(\mathbb{R}^d)$. Since, $\mathcal{S}_D = g(\mathbb{R}^d)$, $\mathcal{S}_D$ is an affine space of dimension p and g is continuous in particular for the canonical norm induced on $\mathcal{L}^2([0,T],\mathbb{R}^p)$. Therefore $\mathcal{S}_D$ is an affine space of finite dimension and is a closed set.

FINDING A UNIQUE MINIMIZER We conclude by applying Hilbert projection lemma: our problem of minimizing $\int_0^T \left\|X^s(\tau) - X^D(\tau)\right\|$, amounts to an orthogonal projection problem. Because $\mathcal{S}_D$ is a closed convex set, we have existence and uniqueness of such projection. Therefore, it exists a unique function $X_D \in \mathcal{S}_D$ and a unique vector D minimizing its distance to the function X^s. □

C.1.4 *Proof to Proposition 3*

Proposition 5. *For λ_D and λ_A sufficiently high, the algorithm that alternates between the estimation of A and the estimation of D_A following eqs. (41) and (42) converges.*

Naturally, one could estimate jointly D and A using least square regression. However, the idea is to verify the convergence of such alternate algorithm in a simple case. We conduct the proof for the first dimension of $\mathcal{Y}$ to lighten notations, meaning that we are regressing the first dimension of Y against the X. A similar reasoning for the other dimension completes the proof.

Proof. We first give the analytical solution for D. Let A^n be fixed.

ESTIMATION OF D Consider:

$$\mathcal{L}_D = \|Y - XA^n - D\|_2^2 + \lambda \|D\|_2^2 \tag{61}$$

where $D = (d, \dots, d) \in \mathbb{R}^Q$. For Q samples, we find d so that $\dfrac{\partial L}{\partial d} = 0$:

$$\begin{aligned} \frac{\partial L}{\partial d} = 0 &\Leftrightarrow -2 * \sum_{i=1}^{Q}(y_i - X_i A^n - d) + 2\lambda d = 0 \\ &\Leftrightarrow Qd + \lambda d = \sum_{i=1}^{Q}(y_i - X_i A^n) \\ &\Leftrightarrow d(Q + \lambda) = \sum_{i=1}^{Q}(y_i - X_i A^n) \\ &\Leftrightarrow d = \frac{\overline{Y - XA}}{1 + \lambda/Q} \end{aligned}$$

where $\overline{Y - XA} = \frac{1}{Q}\sum_{i=1}^{Q}(y_i - X_i A^n)$.

ESTIMATION OF A Let D be fixed and consider:

$$\mathcal{L}_A = \|Y - XA - D\|_2^2 + \gamma \|Y - XA\|_2^2 \tag{62}$$

Similarly, we aim to cancel the first derivative of $\mathcal{L}_A$ with respect to all parameters of $A = (a_1, .., a_p)$:

$$\begin{aligned}
\frac{\partial \mathcal{L}_A}{\partial a_j} = 0 \Leftrightarrow & -2 * \sum_{i=1}^{Q} x_{i,j}(y_i - a_0 x_{i,0} + \cdots + a_p x_{i,p} - d) \\
& - 2\gamma * \sum_{i=1}^{Q} x_{i,j}(y_i - a_0 x_{i,0} + \cdots + a_p x_{i,p}) = 0 \\
\Leftrightarrow & -2X^t(Y - XA - D) - 2\gamma X^t(Y - XA) = 0 \\
\Leftrightarrow & (1+\gamma)X^t XA - X^t(Y - D) - \gamma X^t Y = 0 \\
\Leftrightarrow & (1+\gamma)X^t XA = X^t\Big(\gamma Y + (Y - D)\Big) \\
\Leftrightarrow & A = \frac{B^{-1}X^t}{1+\gamma}\Big((1+\gamma)Y - D\Big)
\end{aligned} \tag{63}$$

where $B = X^t X$. Equation (63) indicates that as soon a D converges, A^n converges. Thus, we now prove the convergence of (D^n). Then, for $n > 1$ consider:

$$\begin{aligned}
\|D^{n+1} - D^n\| &= \frac{1}{1+\lambda/Q}\left\|\overline{Y - XA^n} - \overline{Y - XA^{n-1}}\right\| \\
&= \frac{1}{1+\lambda/Q}\left\|\overline{X(A^n - A^{n-1})}\right\| \\
&= \frac{1}{(1+\lambda/Q)(1+\gamma)}\left\|\overline{XB^{-1}X^t\Big([(1+\gamma)Y - D^n] - [(1+\gamma)Y - D^{n-1}\Big)]}\right\| \\
&= \frac{1}{(1+\lambda/Q)(1+\gamma)}\left\|\overline{XB^{-1}X^t[D^{n-1} - D^n]}\right\| \\
&\leq \frac{K}{(1+\lambda/Q)(1+\gamma)}\|D^{n-1} - D^n\|
\end{aligned}$$

where $K = \|XB^{-1}X^t\|$. Therefore, for λ, γ, sufficiently large, $\frac{K}{(1+\lambda/Q)(1+\gamma)} < 1$. $\|D_n - D_{n-1}\|$ converges as a positive decreasing sequence. Finally, the sequence of (D_n) converge and so the sequence of (A_n). In conclusion, the proposed algorithm converges. □

C.2 ADDITIONAL RESULTS ON DATASETS DEPICTING OTHER DYNAMICS

In this section, we illustrate the learning scheme induced by eq. (27) on fully observed low dimensional dynamics: a simple example emerging from Newtonian mechanics and a population dynamics model. Performances are evaluated via standard metrics: MSE (lower is better) and relative Mean Absolute Error (rMAE, lower is better)

C.2.1 *Damped Pendulum*

ARCHITECTURE DETAILS The physical parameters to be learned is a scalar of dimension 1, and h_u is a 1-hidden layer MLP with 200-hidden neurons with leaky-relu activation.

OPTIMIZATION For this dataset we use RMSProp optimizer with learning rate 0.0004 for 100 epochs with batch size 128. We supervise the trajectories up to $t = \Delta t \times 50$, i.e we enforce d_ϕ over $(t_0 + \Delta t, .., t_0 + 50\Delta t)$. Overall the number of optimization subsequences for training is 17000. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 10-batches on h_k then 10-batches on h_u.

HYPERPARAMETERS We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.001$. We initialize $\lambda_h = 0.1$ and increase it geometrically up to $\lambda_h = 100$. λ_{h_u} is fixed through training at 0.1. The hyper-parameters were chosen by randomly exploring the hyper-parameters space by sampling them so that $\lambda \sim \mathcal{U}(1, 0.1, \dots, 10^{-4})$. We select the ones with the lowest prediction errors, i.e with lowest $\mathrm{d}_\phi(h, f)$. For the ablation study of Table 6, we set to 0 the hyper-parameters associated to the non-considered loss. The training time for this dataset is 1 hour.

C.2.2 *Lotka-Volterra*

ARCHITECTURE DETAILS The physical parameters to be learned is a vector of dimension 2 accounting for (α, β) in eq. (65), and h_u is a 1-hidden layer MLP with 200-hidden neurons with leaky-relu activation.

OPTIMIZATION We use Adam optimizer with learning rate 0.0005 for 200 epochs with batch size 128. Overall the number of sequences for training is 15000. We supervise the trajectories up to $t = \Delta t \times 25$, i.e we enforce d_ϕ over $(t_0 + \Delta t, .., t_0 + 25\Delta t)$. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 10-batches on h_k then 10-batches on h_u.

HYPERPARAMETERS We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.001$. We initialize $\lambda_h = 0.001$ and increase it geometrically up to $\lambda_h = 1$. λ_{h_u} is fixed through training at 0.001. The hyper-parameters were chosen by randomly exploring the hyper-parameters space by sampling them so that $\lambda \sim \mathcal{U}(1, 0.1, \dots, 10^{-4})$. We select the ones with the lowest prediction errors (i.e lowest $\mathrm{d}(h, f)$). For the ablation study of Table 6, we set to 0 the hyper-parameters associated to the non-considered loss. The training time for this dataset is 2 hours.

DAMPED PENDULUM (DPL) Now a standard benchmark for hybrid models, we consider the motion of a pendulum of length L damped due to viscous friction (Greydanus, Dzamba, and Yosinski, 2019; Yin, Le Guen, et al., 2021). Newtonian mechanics provide an ODE describing the evolution of the angle x of the pendulum:

$$\ddot{x} - g/L \sin(x) + k\dot{x} = 0 \tag{64}$$

We suppose access to observations of the system state $Z = (x, \dot{x})$. We consider as physical motion hypothesis $h_k(x, \theta_k) = \theta_k \sin(x)$. The true pulsation $\theta^* = g/L$ of the pendulum has to be estimated with θ_k. The viscous friction term $k\dot{x}$ remains to be estimated by h_u.

POPULATION DYNAMICS (LV) Lotka-Volterra ODE system models a prey/predator population dynamics describing the growth of the preys (x) without predators (y), and the extinction of predators without preys, the non linear terms expressing the encounters between both species:

$$\dot{x} = \alpha x - \beta xy, \quad \text{and} \quad \dot{y} = -\gamma y + \delta xy \tag{65}$$

We observe the system state $Z = (x, y)$ and set as prior knowledge: $h_k(x, y) = (\theta_k^1 x, -\theta_k^2 y)$. $\theta^\star = (\alpha, \gamma)$ has to be estimated by $\theta_k = (\theta_k^1, \theta_k^2)$. h_u accounts for the non linear terms $(\beta xy, \delta xy)$.

EXPERIMENTAL SETTING For both DPL and LV experiments, we consider the following setting: we sample the space of initial conditions building 100/50/50 trajectories for the train, validation and test sets. The sequences share the same parameters; respectively $(\frac{g}{L}, k)$, for DPL, and $(\alpha, \beta, \gamma, \delta)$ for LV. The parameter θ_k is set to a neuron (of dimension 1 in the pendulum and 2 for LV) and h_u is a 2-layer MLP.

Table 6: Experimental Results for PDL and LV data. The presented metric for parameter evaluation is the rMAE reported in %. Pred. columns report the prediction log MSE on trajectories on test set.

Model	PDL		LV	
	rMAE$(\theta_k, \theta^\star)$	Pred. logMSE	rMAE$(\theta_k, \theta^\star)$	Pred. logMSE
Ours eq. (27)	**1.56 (0.009)**	**-13.7 (0.84)**	**7.80 (0.011)**	-9.28 (0.75)
Only d(h, f)	9.35 (0.04)	-13.3 (0.65)	24.5 (0.017)	-9.21 (0.91)
d(h, f) + d(h_k, f)	1.82 (0.01)	-13.4 (0.56)	7.91 (0.02)	-9.01 (0.99)
d(h, f) + d$(h_u, 0)$	11.1 (0.03)	-12.9 (0.29)	9.80 (0.098)	-9.45 (0.55)
Aphynity	6.15 (0.009)	-12.2 (0.13)	21.1 (0.016)	**-9.89 (0.53)**
NeuralODE	–	-10.1 (0.32)	–	-9.11 (1.1)

IDENTIFICATION AND PREDICTION RESULTS Table 6 shows that despite accurate trajectory forecasting, the unconstrained setting «Only d(h, f)» fails at estimating the models parameters, showing the need for regularization for identification. Constraining the norm of the ML component can be insufficient: for LV data, both Aphynity and d(h, f)+d$(h_u, 0)$ do not accurately estimate the model parameters. However, the control of d(h_k, f), following eq. (27), significantly improves the parameter identification for both datasets. Indeed, in the PDL case, h_k and f are (pseudo)-periodic of the same period, hence the gain in the performances. Finally, our proposition based on eq. (27) is able to identify the parameters of DPL and LV equation with a precision of respectively 1.56% and 7.8% beating all considered baselines. Regarding prediction performances, in under-constrained settings («Only d(h, f)» in Table 6), h_u learns to corrects the inaccurate h_k. Table 6 and figs. 30 and 31 (appendix C.2.3) show that our proposition provides more consistent prediction performances. These experiments confirm that the constraints on h_k and h_u arising from the control of the upper bound of eq. (27) increase interpretability and maintain prediction performances.

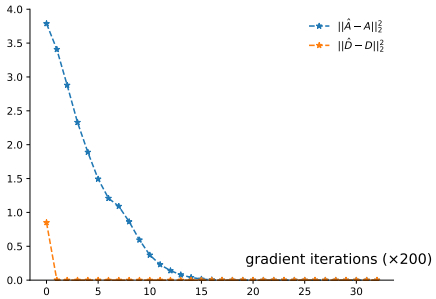

Figure 29: Affine Case : Evolution of the MSE between estimated dynamics $(\hat{A}, \hat{D})$ and the true one (A, D) with the number of gradients steps for linearized DPL.

THROWBACK TO THE AFFINE CASE We verify the convergence proved in section 4.3.2 using the damped pendulum (eq. (64)) linearized in the small oscillations regime (see appendix C.2.2). Making an affine hypothesis following eq. (40), we apply our alternate projection algorithm and

optimize A and D_A alternately using SGD. Figure 29 shows that we are able to accurately estimate A and D using our proposition, recovering both the oscillation pulsation and the damping coefficient.

C.2.3 *Results for Pendulum and Lotka-Volterra Datasets*

We provide respectively in figs. 30 and 31 phase diagrams for the damped pendulum and Lotka-Volterra experiments. Both graphs in the phase space indicate that the trajectories and their nature are well handled by the learned decomposition, providing a periodic phase space for Lotka-Volterra (fig. 31), and a converging spiral for the damped pendulum (fig. 30).

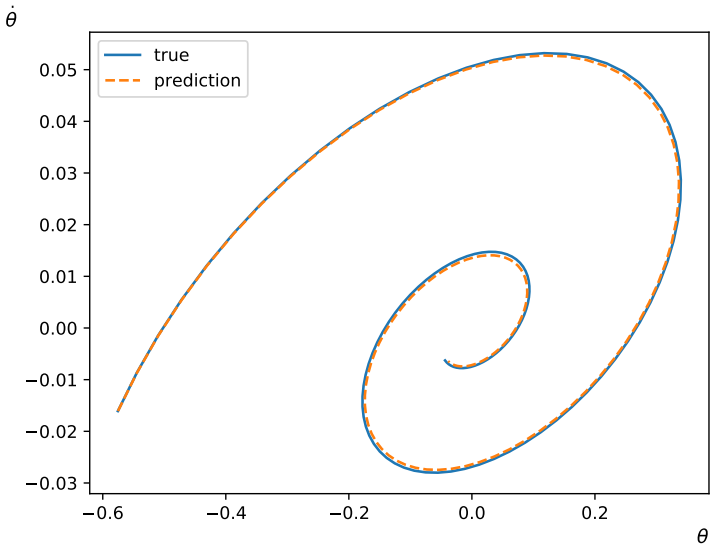

Figure 30: Damped Pendulum Phase Diagram. The true phase diagram (blue) and learned (orange dashed) are close, indicating consistency in the prediction

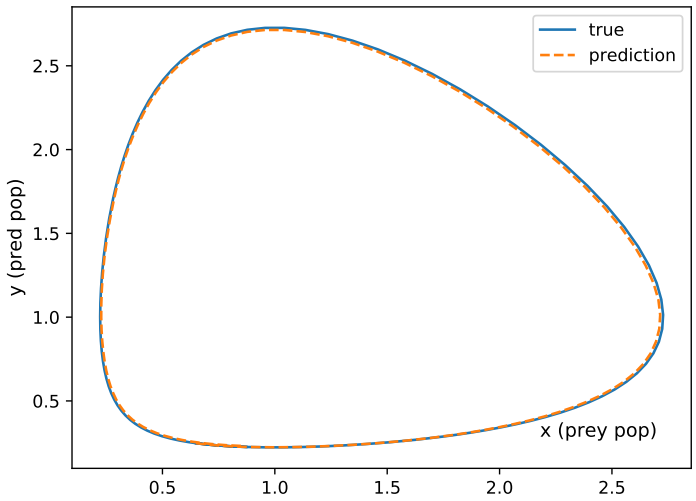

Figure 31: Lotka-Volterra Phase Diagram. The true phase diagram (blue) and learned (orange dashed) are close, indicating consistency in the prediction

Part VI

RÉSUMÉ EN FRANÇAIS

D

RÉSUMÉ EN FRANÇAIS

D.1 INTRODUCTION

L'apprentissage automatique a pour objectif de construire des systèmes informatiques s'améliorant automatiquement avec l'expérience. Cette technologie est largement perçue comme l'une des plus disruptives de notre époque, et l'utilisation de méthodes d'apprentissage automatique nécessitant de grandes bases de données se retrouve aujourd'hui dans de nombreux domaines: scientifique, économique ou encore technologique. Le développement du ML est actuellement étendu et amplifié avec l'essor de l'apprentissage profond (DL) et l'utilisation de réseaux neuronaux artificiels profonds, généralement optimisés par des techniques de descente de gradient (Goodfellow, Bengio, and Courville, 2016).
Les méthodes d'apprentissage profond s'appuient sur plusieurs niveaux de représentation pour expliciter des données en modèles: à partir des données d'entrée brutes, chaque niveau transforme son entrée en une représentation à un niveau supérieur, légèrement plus abstrait (LeCun, Bengio, and G. Hinton, 2015). La composition de suffisamment de transformations permet d'apprendre des fonctions très complexes, qui sont, la plupart du temps, difficiles à traduire sous une forme analytique intuitive. L'utilisation d'accélérateurs matériels tels que les processeurs graphiques (GPU) et la création de frameworks efficaces ont permis de construire des systèmes d'apprentissage profond contenant des milliards de paramètres et pouvant être entraînés sur de très grandes collections de données telles que des images, des vidéos et des échantillons de parole. Ainsi, la dernière décennie a été marquée par de nombreuses percées scientifiques et technologiques dans de nombreux domaines tels que la vision par ordinateur (Krizhevsky, Sutskever, and G. E. Hinton, 2012; Szegedy et al., 2017), le traitement du langage naturel (Collobert and Weston, 2008; Y. Wu et al., 2016) ou encore les sciences de la santé (Leung et al., 2014). La récente résolution de problèmes ayant résisté aux meilleures tentatives de la communauté de l'intelligence artificielle pendant de nombreuses années (Silver et al., 2016) pousse la communauté à se demander si ces résultats se transposent à d'autres domaines comme les sciences naturelles et l'étude des processus et systèmes physiques.

D.1.1 *ML et physique*

La compréhension et la prédiction du monde sont depuis toujours les principaux moteurs de la recherche scientifique. Ainsi, des domaines tels que la géophysique, l'astronomie, l'épidémiologie ou la cinétique chimique font l'objet d'études depuis des siècles et sont aujourd'hui dominés par des modèles mécanistes, reposant sur une compréhension approfondie des phénomènes sous-jacents, traduits mathématiquement par des relations statistiques et/ou physiques, autrement dit des lois. La disponibilité croissante de la puissance des superordinateurs dans les années 1970 a rendu possible le développement de simulations numériques (Lynch, 2008), reposant sur l'assimilation de grandes quantités de données pour modéliser l'évolution du système physique dans le temps (Bauer, Thorpe, and Brunet, 2015). De tels outils donnent d'excellents résultats :

les modèles de prévision météorologique ont atteint des performances sans précédent au cours des 40 dernières années, et sont au cœur des modèles climatiques utilisés par le Groupe d'experts intergouvernemental sur l'évolution du climat (GIEC) pour la surveillance du changement climatique (Stockhause and Lautenschlager, 2017). Cependant, ces approches atteignent aujourd'hui leurs limites : d'une part en termes d'outils et de technologies, les modèles traditionnels sont limités en raison de leur coût de calcul ; d'autre part, en termes de connaissances physiques, en raison de la faible compréhension de certains processus, les modèles sont des approximations de la réalité. De plus, à l'ère de l'imagerie satellitaire, un déluge de données sur le système terrestre est devenu disponible, avec des volumes de stockage dépassant déjà largement les dizaines de pétaoctets (Agapiou, 2017), et extraire des informations et des connaissances interprétables de ces données serait d'une grande aide pour faire avancer la science. Plus qu'une simple expansion de la puissance de calcul, l'efficacité des algorithmes utilisés pour traduire les lois dynamiques en calculs pratiques aura un impact direct sur l'efficacité des modèles existants. Ayant déjà rencontré un succès considérable dans de nombreuses applications, la ML pourrait être utile et jouer un rôle important dans l'avenir de la modélisation physique.

ML GUIDÉ PAR LA PHYSIQUE ET MODÈLES HYBRIDES Bien que l'utilisation de méthodes orientées données dans les géosciences se développe rapidement, elle n'en est encore qu'à ses débuts et progresse de manière inégale (Bergen et al., 2019). Alors que les approches orientées données sont des méthodes d'interprétation agnostiques par rapport aux modèles (ces modèles sont souvent qualifiés de boîtes noires), pour représenter le monde physique, nous recherchons des modèles interprétables. En outre, il serait dommage de se passer des connaissances accumulées pendant des années. Pour dériver des modèles qui apprennent au maximum des données tout en respectant notre compréhension évolutive des lois de la nature, la communauté ML est confrontée à de nouveaux défis: 1. utiliser les connaissances physiques disponibles, 2. produire des modèles physiquement cohérents. À cet égard, un nouveau paradigme apparaît, utilisant les connaissances spécifiques au domaine et intégrant les connaissances scientifiques directement dans le cadre du ML (Bergen et al., 2019). Ces nouvelles approches, appelées ML guidées par la physique (Willard et al., 2020), sont fondamentalement différentes des pratiques courantes purement axées sur les données et se sont développées ces dernières années. Dans cette littérature florissante, un domaine de recherche émergent est le couplage des modèles de processus physiques avec le modèle orienté données (Reichstein et al., 2019). Dans cette perspective, le ML est considéré comme une approche complémentaire aux modèles traditionnels basés sur la physique (Dueben and Bauer, 2018). Les deux offrent des avantages : alors que les approches traditionnelles généralisent et extrapolent mieux, les approches ML à haute expressivité bénéficient de la croissance continue des données disponibles telles que les observations satellitaires, avec des coûts réduits. C'est ce qu'on appelle la modélisation hybride, et c'est l'approche sur laquelle nous nous concentrons tout au long de cette thèse.

LE FOSSÉ DE LA RÉALITÉ Les phénomènes physiques sont basés sur des processus impliquant de multiples échelles et variables : par exemple, les processus pertinents pour comprendre le comportement du géosystème terrestre vont de l'échelle atomique à planétaire, de l'échelle temporelle de la milliseconde aux milliards d'années et prennent en compte des processus dynamiques, thermodynamiques, radiatifs et chimiques (Bergen et al., 2019). Le paradigme traditionnel se traduit par la résolution quotidienne d'un système d'équations différentielles non linéaires à environ un demi-milliard de points par pas de temps entre le moment initial et des semaines ou des mois à venir, et par la prise en compte de processus dynamiques, thermodynamiques,

radiatifs et chimiques fonctionnant à des échelles allant de centaines de mètres à des milliers de kilomètres et de secondes à des semaines. Cependant, la plupart des publications de ML guidé par la physique expérimentent sur des problèmes jouets dans des environnements de faible dimension et développent des modèles qui ne sont pas directement applicables à des environnements complexes et à des scénarios réels. Malgré les promesses montrées par les premières études de validation de concept, la communauté a été lente à adopter le ML de manière plus large. Nous inscrivons notre thèse dans cette recherche naissante en étudiant comment les modèles d'apprentissage profond se traduisent dans des situations réelles. En effet, l'apprentissage profond guidé par la physique est un sujet très récent, et même si l'apprentissage profond est un outil prometteur, il n'a pas encore atteint les résultats des modèles basés sur la physique. Cette thèse se situe dans le domaine préliminaire de la recherche en milieu réel.

Dans cette thèse, nous abordons la question de la modélisation de phénomènes physiques évoluant dans le temps et dans l'espace à partir de données, en utilisant l'apprentissage profond, en visant une application pratique en océanographie. Nous menons une étude incrémentale et explorons le potentiel de l'apprentissage profond pour compléter les modèles physiques du système océanique. La section suivante est consacrée à un bref aperçu des défis de la modélisation océanique. Ensuite, nous présentons plus en détail le sujet et les contributions de cette thèse.

D.1.2 *Cas d'étude*

Les courants océaniques sont des masses d'eau en mouvement. À la surface des océans, ils peuvent modifier la topographie de quelques dizaines de centimètres à plus d'un mètre. Par exemple, la gyre de l'Atlantique Nord tourne dans le sens des aiguilles d'une montre et élève le niveau de la mer en son centre. Depuis le début des années 1990, les champs de vitesse des courants de surface mondiaux sont déduits des observations par satellite de la hauteur de la surface de la mer (SSH) (Dohan and Maximenko, 2010), c'est-à-dire des données altimétriques. Cette approximation est utilisée pour estimer la dynamique des courants lents et à grande échelle (Sinha and Abernathey, 2021), c'est-à-dire à résolution spatiale et temporelle d'environ 50km et une semaine. Cette approximation permet de distinguer les échelles de mouvement dites résolues et non résolues. En effet, ces courants peuvent être observés grâce à des données altimétriques ayant une résolution spatiale d'environ 50km et une résolution temporelle d'environ une semaine. Cependant, la dynamique des courants est influencée par des phénomènes opérant à des échelles beaucoup plus fines de l'ordre de $1\text{km} \times 1\text{day}$ (Lévy, P. Klein, and Treguier, 2001). Les structures à ces échelles sont liées aux courants océaniques par des interactions complexes impliquant l'advection d'une part, et les instabilités dynamiques d'autre part (Lévy, P. Klein, and Treguier, 2001). Alors que l'altimétrie ne fournit aucune information sur ces structures fines, l'imagerie satellitaire à haute résolution pourrait être utilisée pour déduire l'évolution spatio-temporelle à fine échelle. L'imagerie satellitaire haute résolution fournit des informations à travers des mesures telles que la température de surface de la mer (SST), dont l'évolution spatio-temporelle devrait permettre d'améliorer les modèles de courants globaux dérivés de l'altimétrie. L'intégration de ces informations avec les données altimétriques pose plusieurs défis tels que la quantité de données générées à ces échelles fines, l'intégration de données provenant de sources à la fois par leur nature (altimétrie versus images haute résolution) et par leur résolution spatio-temporelle. En effet, les images satellites de SST et de chlorophylle de surface sont maintenant disponibles quotidiennement à une résolution kilométrique. Dans cette thèse, nous considérons les algorithmes ML comme une voie alternative pour inférer les

courants de surface à partir de quantités observables par satellite à haute résolution telles que la température de surface de la mer (SST). Plus précisément, nous explorons les dépendances entre la SST et les champs de vitesse des courants et nous étudions des modèles physico-statistiques hybrides pour représenter l'évolution de la SST observable et de la vitesse des courants non observée. Notons que, même si nous nous appuyons sur ce problème concret, cette thèse est avant tout méthodologique. En effet, nous proposons des modèles très généraux applicables à de nombreux problèmes physiques. Néanmoins, pour comprendre l'intuition des modèles présentés, nous nous référerons au problème prototype, la modélisation des courants marins à partir des données SST avec des réseaux de neurones, tout au long de notre travail.

D.1.3 *Contributions*

Notre objectif est de modéliser les champs de vitesse des courants océaniques à partir des observations de la température de surface de la mer en utilisant l'apprentissage profond. Comme nous nous concentrons sur un problème réel, il est essentiel de garantir la plausibilité de nos résultats d'un point de vue physique. Cependant, Willard et al. (2020) souligne l'incapacité des modèles ML de type boîte noire à généraliser ainsi que leur incapacité à produire des résultats physiquement solides. Comme le ML seul ignore les lois fondamentales de la physique et peut aboutir à des problèmes mal posés ou à des solutions non physiques (Alber et al., 2019), nous nous appuierons sur des modèles hybrides, utilisant à la fois des connaissances physiques préalables et l'apprentissage automatique. Dans ce domaine de recherche, l'un des principaux défis consiste à résoudre les problèmes mal posés inhérents à la décomposition entre les modèles physiques et ceux basés sur les données. Par exemple, cela peut être fait en incorporant des contraintes motivées par la physique dans l'apprentissage de modèles hybrides, par exemple par des pénalités de régularisation. Pour compléter les connaissances physiques préalables par une composante basée sur les données et assurer l'interprétabilité de la décomposition, nous considérons d'abord un modèle simplifié de la dynamique de l'océan. Dans ce contexte, nous introduisons deux contributions.

RÉGULARISATION DES MODÈLES DYNAMIQUES HYBRIDES En incorporant des connaissances physiques, notre cadre considère la dynamique de la SST observée et sa dépendance connue à la vélocité inconnue. Si l'on considère les équations réelles de la dynamique océanique, la dynamique de la vitesse actuelle devrait suivre une équation différentielle ordinaire (ODE). Pour faire face à la difficulté de l'identification et récupérer une estimation significative, nous introduisons une régularisation dynamique sur la vitesse estimée, l'obligeant à suivre une ODE. Cette contribution a conduit à la publication suivante dans un workshop de conférence internationale.

> Marie Déchelle et al. (2020). "Bridging Dynamical Models and Deep Networks to Solve Forward and Inverse Problems". In: *NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning*.

FRAMEWORK FOR THE LEARNING OF HYBRID MODELS Bien qu'elle permette d'obtenir de bons résultats expérimentaux, la régularisation proposée n'est pas fondée sur le plan théorique et ne sera en aucun cas suffisante pour garantir le caractère bien posé de l'approche hybride. Afin de retrouver le caractère bien posé et interprétable de l'apprentissage des modèles hybrides, nous proposons de contrôler une limite supérieure de l'erreur de prédiction et proposons un nouvel

algorithme d'optimisation alternatif. Cette contribution a conduit à la publication suivante dans une conférence internationale.

> Jérémie Dona*, Marie Déchelle*, Marina Levy, et al. (2021). "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction". In: *International Conference on Learning Representations*.

CAS D'ÉTUDE RÉEL Enfin, nous confrontons le cadre proposé à la modélisation de la dynamique océanique réelle. Nous étudions les limitations et proposons une adaptation de notre modèle, incorporant de nouvelles sources de connaissances physiques. Ceci améliore les performances dans les tâches d'identification et de prédiction par rapport au seul modèle théorique.

D.2 DONNÉES ET NOTATIONS

D.2.1 *Données*

Tout au long de cette thèse, nous travaillons sur des données représentatives de la dynamique de la surface de l'océan.

VARIABLES DANS L'OCÉAN L'océanographie vise à comprendre les processus océaniques. Pour cela, on étudie l'évolution des propriétés physiques de l'océan, telles que sa température $\boldsymbol{T}$, sa salinité $\boldsymbol{S}$, sa densité ρ, la concentration en chlorophylle, la vitesse $\boldsymbol{U} = (u, v, w)$ (Chassignet, Le Sommer, and Wallcraft, 2019). Ces champs sont quadridimensionnels, c'est-à-dire qu'ils évoluent dans le temps t et dans un espace tridimensionnel x, y, z. Pour accéder à un sous-échantillon discret de ces quantités, les dernières décennies ont vu une augmentation des systèmes d'observation des océans, soit à partir de mesures in-situ, par exemple grâce à des flotteurs ou des bouées (Roemmich et al., 2009), soit par télédétection (Esaias et al., 1998). Ces quantités sont accessibles soit par des mesures in-situ, par exemple grâce à des flotteurs ou des bouées (Roemmich et al., 2009), soit par télédétection. Le développement des systèmes satellitaires tout au long du 20ème siècle a permis d'augmenter le nombre d'observations océaniques. De nos jours, $\boldsymbol{T}$, $\boldsymbol{S}$ et la concentration en chlorophylle peuvent être observés par des satellites, à une distance de $1\text{km} \times 1\text{day}$ à la surface de la mer. Cela donne un sous-échantillonnage avec une résolution spatiale d'environ un kilomètre et une résolution temporelle d'environ un jour. Notez que les composantes horizontales de la vitesse u et v ne sont pas directement accessibles. Elles sont plutôt déduites d'observations à distance de la hauteur de la surface de la mer (SSH), à une résolution de $50\text{km} \times 1\text{week}$. Malgré de grands progrès, les systèmes d'observation fournissent des informations limitées et incomplètes. En particulier pour l'océan profond, c'est-à-dire en dessous d'une profondeur d'eau de plus de 200m, seules quelques rares observations in situ sont disponibles (Levin et al., 2019).

L'OCÉAN EN ÉQUATIONS L'océan échange des flux de chaleur, d'eau douce, de sel et de quantité de mouvement, par exemple par le biais de la tension du vent dans le cas de l'interface atmosphère-océan, avec la terre solide, les marges continentales, la glace de mer et l'atmosphère. De tels processus peuvent être décrits avec une bonne approximation par les équations primitives, c'est-à-dire un ensemble d'équations différentielles partielles non linéaires. Sous les hypothèses décrites dans Madec et al. (2017), celles-ci comprennent le bilan de quantité de mouvement, l'équilibre

hydrostatique, l'équation d'incompressibilité, les équations de conservation de la chaleur et du sel. Dans cet ordre, elles s'écrivent comme suit :

$$\frac{\partial \boldsymbol{U}}{\partial t} + (\boldsymbol{U} \cdot \nabla)\boldsymbol{U} + g'\nabla h = \gamma \wedge \boldsymbol{U} + D^{\boldsymbol{U}} + F^{\boldsymbol{U}}$$
$$\frac{\partial p}{\partial z} + \rho g = 0$$
$$\nabla \cdot \boldsymbol{U} = 0$$
$$\frac{\partial \boldsymbol{T}}{\partial t} = -\nabla \cdot (\boldsymbol{T}\boldsymbol{U}) + D^{\boldsymbol{T}} + F^{\boldsymbol{T}}$$
$$\frac{\partial \boldsymbol{S}}{\partial t} = -\nabla \cdot (\boldsymbol{S}\boldsymbol{U}) + D^{\boldsymbol{S}} + F^{\boldsymbol{S}}$$

où γ est le paramètre de Coriolis, h la profondeur de la couche de surface obtenue à partir des observations de la hauteur de la surface de la mer (SSH), g' la gravité réduite qui prend en compte la stratification en densité de l'océan de telle sorte que $g' \approx g.10^{-3}$, p est la pression, ρ est la densité, $D^{\boldsymbol{T}/\boldsymbol{U}/\boldsymbol{S}}$ se réfère aux processus à petite échelle et $F^{\boldsymbol{T}/\boldsymbol{U}/\boldsymbol{S}}$ équilibre les forçages de surface, c'est-à-dire les échanges à la surface d'énergie cinétique, de chaleur et de salinité. La circulation océanique, c'est-à-dire la dynamique des champs de vitesse des courants, est maintenant modélisée de façon réaliste dans des modèles structurés tridimensionnels tels que NEMO (Madec et al., 2017), en s'appuyant sur la résolution numérique des équations primitives ci-dessus. Dans le cadre de cette étude, nous travaillons sur des données issues de tels modèles. Cela nous libère des considérations inhérentes aux observations, et des limitations telles que la couverture nuageuse tout en nous offrant un cadre réaliste. Par ailleurs, nous travaillons sur des données de surface, c'est-à-dire que nous ne considérons que la surface bidimensionnelle de l'océan générée par (x, y), ci-après dénommée T, $U = (u, v)$ et F pour respectivement la température, la vitesse et les forçages. Nous considérons comme variables d'intérêt la température T et les champs de vitesse des courants de surface $U = (u, v)$, et donc uniquement la dynamique sur T et U c'est-à-dire que nous considérons l'advection de la température par les champs de vitesse des courants de surface. En particulier, nous ne représentons pas les vitesses verticales, responsables des mouvements entre la surface de l'océan et les strates inférieures. Dans un cadre bidimensionnel, $\nabla \cdot (TU)$ désigne l'advection d'une quantité scalaire T par un champ de vitesse $U = (u, v)$ et s'écrit comme suit : $\nabla \cdot (TU) = \frac{\partial T}{\partial x} u + \frac{\partial T}{\partial y} v$. Nous travaillons d'abord sur une représentation simplifiée de la dynamique océanique, en nous appuyant sur des hypothèses simplificatrices, que nous passons en revue dans la suite.

DONNÉES SYNTHÉTIQUES Pour les mouvements lents (c'est-à-dire de temps caractéristique supérieur à un jour et de dimension spatiale supérieure à 20km) la diffusion est omise et l'incompressibilité est supposée, c'est-à-dire que les termes turbulents sont nuls : $(U \cdot \nabla)U = 0$. Alors que T est observé par les satellites, U n'est pas connu. Cependant, la hauteur de la surface de la mer (SSH) pourrait être utilisée pour calculer des estimations grossières de U. En effet, sous des hypothèses telles que la stationnarité ($\frac{\partial U}{\partial t} = 0$), l'incompressibilité ($(U \cdot \nabla)U = 0$), les forçages peuvent être omis. Dans ce cas, l'équation sur U peut être réécrite en :

$$\gamma \wedge U = -g'\nabla h$$

Dans ce cas, la SSH h peut être considérée comme une fonction de courant, c'est-à-dire une fonction dont les dérivées expriment les composantes de la vitesse. Lorsqu'elle est projetée sur les axes x et y, l'équation géostrophique devient

$$\gamma v = -g' \frac{\partial h}{\partial x}, \qquad \gamma u = -g' \frac{\partial h}{\partial y}$$

Notez que ces équations ne tiennent pas à des échelles fines car les hypothèses de stationnarité et d'incompressibilité ne tiennent qu'à grande échelle. Nous étudions tout d'abord un ensemble de données générées à partir d'hypothèses simplifiées, que nous appelons Adv+F (pour advection + forçage). Nous ne nous basons pas sur les vrais U et F, mais nous les construisons. Leur calcul est décrit ci-dessous. Nous générons des données en suivant l'équation de traçage:

$$\frac{\partial T}{\partial t} = -\nabla \cdot (TU) + F$$

où les calculs de U et F sont dérivés ci-après. Notons que les équations de transport décrivent un large éventail de phénomènes physiques tels que la concentration chimique, la dynamique des fluides ou les propriétés des matériaux. Dans cette thèse, nous interprétons cette équation de transport comme l'évolution de la température T advectée par un champ de vitesse U dépendant du temps et soumis à un forçage F.

DONNÉES RÉALISTES ISSUES D'OGCM Après avoir travaillé sur des données idéales, nous voulons tester les modèles développés sur des données plus proches de la réalité. Cependant, pour éviter de faire face à de nombreuses incertitudes, nous ne travaillons pas sur des observations réelles. Pour étudier la capacité de l'apprentissage profond à reconstruire des champs de vitesse de courant de surface à fine échelle à partir des SST, nous utilisons les données de la simulation NATL60, basée sur le code NEMO (Ajayi et al., 2020). Il s'agit de la première simulation à l'échelle kilométrique (résolution de 1/60°) de la circulation océanique dans l'Atlantique Nord qui prend en compte la complexité des côtes et des reliefs sous-marins ainsi que la grande variabilité des conditions atmosphériques de surface. Les observations satellitaires ont une résolution de 1/4 de degré. Nous utilisons NATL60, un ensemble de données provenant d'une simulation complète d'un modèle océanique réaliste basé sur le moteur océanique NEMO exécuté à une résolution kilométrique sur le bassin de l'Atlantique Nord. Les données ont été fournies par l'équipe de recherche MEOM, du laboratoire IGE de l'Université Grenoble Alpes.

D.2.2 *Notations*

A un instant t, nous observons la SST $T_t \in \mathcal{T} \subseteq \mathbb{R}^p$. Cette température est influencée par la vitesse d'écoulement de surface non observée, c'est-à-dire un champ vectoriel, $U_t \in \mathcal{U} \subseteq \mathbb{R}^{2q}$. Pour simplifier la notation, nous désignons les variables observées et non observées par $X_t = (T_t, U_t) \in \mathcal{T} \times \mathcal{U} \subseteq \mathbb{R}^{p+2q}$: Soit $X_t \in \Omega \subseteq \mathbb{R}^{p+q}$ un état physique partiellement observé, écrit comme $X_t = (T_t, U_t) \in \mathcal{T} \times \mathcal{U}$ où $T_t \in \mathcal{T} \subseteq \mathbb{R}^p$ est la SST observée et $U_t \in \mathcal{U} \subseteq \mathbb{R}^q$ est la vitesse du courant de surface non observée. Nous considérons le système dynamique dont l'état au temps t est noté $X_t = X(t)$:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = f(X_t, t)$$

Toutes les quantités, observées ou à estimer, sont échantillonnées régulièrement sur une grille spatio-temporelle : à chaque pas de temps t, le champ de vitesse variable dans le temps U_t s'écrit

$U_t = (u_t, v_t)$, et u_t, v_t, T_t et le terme de forçage F_t sont tous de taille 64×64. Nous réécrivons f comme $f = (f_T, f_U)$ agissant sur T et U respectivement :

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} T_t \\ U_t \end{pmatrix} = \begin{pmatrix} f_T(X_t) \\ f_U(X_t) \end{pmatrix}$$

f_T et f_U peuvent être interprétés comme suit: f_T représente la dynamique des T observés et f_U représente la dynamique des U non observés. Bien que f_U ne soit pas connu, nous montrons que U suit une ODE, c'est-à-dire qu'il existe f_U tel que $\frac{\mathrm{d}U}{\mathrm{d}t} = f_U(U, t)$. Dans le contexte des données NATL6o réalistes, $f_T(X) = -\nabla.(TU) + D^T + F^T$ et $f_U(U) = -(U.\nabla)U + \gamma \wedge U - g'\nabla h + D^U + F^U$.

HYPOTHÈSE DYNAMIQUE Nous étudions des modèles hybrides, c'est-à-dire que l'on suppose disponible une connaissance partielle de la dynamique du T_t observé :

$$\frac{\mathrm{d}T_t}{\mathrm{d}t} = f_T(X_t) = f_T^p(X_t) + f_T^d(X_t)$$

où f_T^p représente la partie physique de f_T. f_T^d représente la dynamique qui n'est pas englobée par f_T^p. Plus formellement, $f_T^p \in \mathcal{H}_p$ est un opérateur connu avec des paramètres inconnus $\theta^\star$, et $f_T^d \in \mathcal{H}_d$ est la dynamique résiduelle inconnue. Notons que les paramètres inconnus $\theta^\star$ sont en fait les champs de vitesse U. Les expressions $\mathcal{H}_p$ et $\mathcal{H}_d$ désignent des espaces de fonctions.

MODÉLISATION HYBRIDE DES OCÉANS Dans le cas de l'équation sur T, F^T représente les termes de forçage de surface. D^T représente les paramétrisations de la physique à petite échelle : il représente la physique à l'échelle de la sous-grille, c'est-à-dire les processus physiques importants à petite échelle qui se produisent à des échelles de longueur qui ne peuvent pas être résolues de manière adéquate sur une maille de calcul. Par exemple, les mouvements turbulents ne sont jamais résolus explicitement, même partiellement. Au lieu de cela, ils sont paramétrés (Madec et al., 2017). Même si l'estimation de F^T et D^T est très importante pour les simulations à long terme, la compréhension des paramétrisations à l'échelle de la sous-maille est encore incomplète, et les champs de forçage sont encore mal connus (Chassignet, Le Sommer, and Wallcraft, 2019). Ainsi, nous ne considérerons l'advection que comme une partie de la connaissance physique préalable, et nous viserons à apprendre à la fois les flux d'échanges F^T et les paramétrisations D^T en tant que résidus. La deuxième difficulté est le manque actuel de connaissance des termes sources et puits, incluant non seulement le forçage thermique mais aussi les vitesses d'entraînement ou l'advection verticale, que nous négligeons pour le moment (Rio and Santoleri, 2018). Par ailleurs, tout au long de ce travail, nous supposerons que f_U n'est pas connu. Nos hypothèses de modélisation correspondent alors à :

$$\begin{aligned} f_T^p(T, U) &= -\nabla.(TU) \\ f_T^d &= D^T + F^T \\ f_U &\text{ is unknown.} \end{aligned}$$

que l'on peut réécrire

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \begin{pmatrix} f_T^p(X_t) + f_T^d(X_t) \\ f_U(X_t) \end{pmatrix}$$

En résumé, pour prédire avec précision la dynamique de T, nous cherchons à apprendre f_T^d tout en estimant avec précision U. Cette description correspond à nos hypothèses de modélisation.

Dit autrement, f_T est une fonction cible inconnue. Notre objectif est d'apprendre une estimation de f_T sur la base de nos connaissances disponibles, constituées d'hypothèses préalables sur f_T^p et d'observations. Le problème d'apprentissage est décrit précisément ci-dessous.

D.2.3 *Objectifs*

PROBLÈME D'APPRENTISSAGE Notre objectif est de prédire les trajectoires de T, c'est-à-dire de modéliser l'évolution de la partie observable suivant $\frac{\mathrm{d}T_t}{\mathrm{d}t} = f_T(X_t)$ avec un modèle hybride. Nous approximons f_T par une fonction $h_T \in \mathcal{H}$ apprise à partir des données observées, où $\mathcal{H}$ est un espace d'hypothèses. Nous supposons $h_T = h_T^p + h_T^d$ où $h_T^p \in \mathcal{H}_p$, c'est-à-dire que le modèle physique appartient au même espace d'hypothèses que f_T^p : il a la même forme paramétrique. Ses paramètres sont notés θ_p. Nous prenons comme antériorité physique sur la dynamique : $h_T^p(T, \theta_p) = -\nabla.(T\theta_p)$. Nous cherchons à apprendre les paramètres θ_p de h_T^p, c'est-à-dire les U non observés. Ainsi, nous avons $h_T^p(., \theta^*) = f_T^p$. La forme libre h_T^d vise à apprendre f_T^d. $h_T^d \in \mathcal{H}_d$ est représenté par une fonctionnelle de forme libre avec des paramètres θ_d, par exemple un réseau neuronal. Enfin, le problème d'apprentissage consiste à estimer à partir de données les paramètres de h_T^p de manière à ce qu'ils correspondent aux vrais paramètres physiques et de h_T^d pour approximer au mieux la dynamique inconnue f.

OBJECTIF D'APPRENTISSAGE INTUITIF À cet égard, un objectif d'apprentissage intuitif consiste à appliquer $\frac{\mathrm{d}T_t}{\mathrm{d}t} = h_T(T_t)$, c'est-à-dire à minimiser la distance entre $h_T = h_T^p + h_T^d$ et f_T, comme suit :

$$\min_{h_T \in \mathcal{H}} \mathbb{E}_{s \sim p_S} \|h_T(s) - f_T(s)\|_2$$

où p_S est la distribution de l'état X qui tient compte des états initiaux variables. Chaque s définit un échantillon d'apprentissage. Une telle approche ne fournit aucune garantie physique sur notre modèle. En effet, la minimisation de cet objectif avec $h_T = h_T^p + h_T^d$ permet de prédire des trajectoires précises mais peut avoir un nombre infini de solutions. Par exemple, h_T^d peut contourner l'hypothèse physique h_T^p. Ainsi, l'interprétabilité n'est pas garantie. Notre objectif n'est pas seulement de prédire les trajectoires exactes de T, mais aussi de nous assurer que nous apprenons une décomposition physiquement significative $h_T = h_T^p + h_T^d$, c'est-à-dire de surmonter le caractère mal posé. Comment s'assurer que les états appris X sont physiquement significatifs ? C'est-à-dire : comment s'assurer que h_p capture toute la physique incluse dans f_p ? Nous pouvons affiner nos tâches d'apprentissage en deux objectifs spécifiques : l'identification du système, c'est-à-dire l'estimation des paramètres du modèle physique (les champs de vitesse des courants) à partir des observations (la SST), et la prédiction, c'est-à-dire la récupération des trajectoires associées à la dynamique (de la vitesse et de la SST). Les deux sont essentiels pour les modèles hybrides MB/ML de systèmes dynamiques. Alors que la prédiction vise une extrapolation robuste, l'identification tient compte de l'interprétabilité physique du modèle MB/ML. Alors que la résolution de ces deux problèmes à l'aide d'une formulation basée sur un modèle admet des solutions numériques bien connues, par exemple en utilisant la méthode de l'adjoint (Courtier, Thépaut, and Hollingsworth, 1994; Le Dimet and Talagrand, 1986), la combinaison de modèles physiques et d'apprentissage profond reste un domaine de recherche ouvert. Dans ce contexte, les applications ML se concentrent principalement sur la tâche de prédiction, au détriment de l'identification du système. En effet, Ayed et al. (2020) montrent que sans aucune connaissance préalable, les estimations récupérées des états d'un système dynamique ne sont pas physiquement plausibles malgré des prédictions précises. De plus,

comme le noteẽxtciteYin2021, l'apprentissage d'une décomposition linéaire MB/ML avec la seule supervision sur les trajectoires du système est mal posé et admet un nombre infini de décompositions. De telles observations soulignent la nécessité d'incorporer des contraintes motivées physiquement dans l'apprentissage de modèles hybrides, par exemple par des pénalités de régularisation. Plusieurs travaux proposent déjà des contraintes additionnelles pour guider le modèle vers des solutions physiques (Jia, Willard, Karpatne, Read, Zwart, Michael S Steinbach, et al., 2019; Linial et al., 2021; Yin, Le Guen, et al., 2021).
Dans cette thèse, nous proposons des raffinements de cet objectif, en utilisant les connaissances physiques pour dériver de nouvelles contraintes.

DISTANCE AVEC LES FLUX Dans la pratique, f_T est inconnu. Pour s'entraîner, on utilise donc les trajectoires associées à la dynamique. Nous minimisons la distance entre les flux ODE ϕ_h et ϕ_f définis par h et f, $\delta(\phi_h, \phi_f)$, sur toutes les conditions initiales X_0 :

$$\delta(\phi_h, \phi_f) = \mathbb{E}_{X_0} \int_{t_0}^{t} \left\| \phi_h(\tau, X_0) - \phi_f(\tau, X_0) \right\|_2 \mathrm{d}\tau$$

Nous avons $\delta(\phi_h, \phi_f) = 0 \implies \phi_h = \phi_f \implies h = f$. Les gradients de $\delta(\phi_h, \phi_f)$ par rapport aux paramètres de h peuvent être estimés analytiquement en utilisant la méthode adjointe (R. T. Q. Chen et al., 2018) ou en utilisant des solveurs explicites, par exemple Rk45, et en calculant les gradients grâce à la rétropropagation, voir Onken and Ruthotto (2020).
Pour le calcul, nous nous appuyons sur un échantillonnage temporel de X : nos ensembles de données sont composés de n séquences d'observations de longueur N, $X^i = (X^i_{t_0}, \dots, X^i_{t_0+N\Delta t})$, où chaque séquence X^i correspond à une condition initiale $X^i_{t_0}$. Nous échantillonnons ensuite l'espace des conditions initiales $X^i_{t_0}$ pour calculer une approximation de Monte-Carlo de $\delta(\phi_h, \phi_f)$. Soit ODESolve la fonction intégrant tout état initial arbitraire X_{t_0} jusqu'au temps t avec la dynamique h, de sorte que $X_t = \text{ODESolve}(X_{t_0}, h, t)$. L'estimation de $\delta(\phi_h, \phi_f)$ s'écrit alors comme :

$$\delta(\phi_h, \phi_f) \approx \frac{1}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \left\| \text{ODESolve}(X^i_{t_0}, h, t_0 + j\Delta t) - X^i_{t_0+j\Delta t} \right\|_2$$

En d'autres termes, nous nous appuyons sur les trajectoires associées à la dynamique :

$$\text{ODESolve}(X^i_{t_0}, h, t_0 + j\Delta t) = X^i_{t_0} + \int_{t_0}^{t_0+j\Delta t} h(X^i_t)\, \mathrm{d}t$$

Notez qu'en pratique, nous n'apprenons pas la dynamique réelle h_T mais son intégration numérique.

D.3 RÉGULARIZATION DYNAMIQUE POUR L'APPRENTISSAGE DE MODÈLES HYBRIDES

Nous proposons une première approche pour surmonter le caractère mal posé induit par le rapprochement des modèles numériques d'équations aux dérivées partielles et de l'apprentissage profond. Nous proposons de régulariser l'apprentissage en se basant sur la seule minimisation de la différence entre notre modèle h_T et la dynamique réelle f_T. Pour garantir que h^p_T est physiquement plausible, nous suggérons d'intégrer des connaissances préalables dans le processus d'apprentissage. Dans notre cas, identifier les paramètres θ_p de h^p_T revient à résoudre un problème inverse (D. L. T. Anderson and Willebrand, 1989). En effet, la SST est un traceur

océanique, c'est-à-dire qu'elle peut être utilisée pour suivre les courants et déduire la circulation océanique à grande échelle (Bigg and Killworth, 1988; England and Maier-Reimer, 2001). Ainsi, comme dans Ayed et al. (2020) and Bezenac, Pajot, and Gallinari (2019), nous proposons d'apprendre θ_p à partir des observations passées de T. Cependant, n'ayant pas d'observations de U, on ne peut se baser que sur la prédiction des futurs T pour l'évaluation. Pour contraindre davantage l'apprentissage, nous proposons de tirer parti des connaissances dynamiques antérieures en introduisant une régularisation dynamique sur les états non observés. Nous supposons que U est la solution d'une ODE. Notre proposition est donc d'imposer à U de suivre une dynamique décrite par une équation différentielle. Notons que cette dynamique est inconnue et que nous l'approximons avec un réseau de neurones, en utilisant leur interprétation comme discrétisation numérique des équations différentielles (He et al., 2016; Lu et al., 2018).
Enfin, nous passons de notre problème initial, à savoir l'estimation de U et l'apprentissage de la dynamique du seul T, à une nouvelle formulation : l'apprentissage de la dynamique de l'état complet $X = (T, U)$. Notre proposition revient à apprendre un modèle prédictif d'espace d'état pour estimer l'état complet X et, en considérant la dynamique de la variable observée et sa dépendance connue à des variables inconnues, pour prédire X sur des pas de temps futurs. Nous montrons que l'espace latent d'un tel modèle figure les champs de vitesse U et est donc physiquement significatif comme souhaité.

OPTIMISATION Nous voulons estimer avec précision la dynamique de la variable observée T, mais aussi modéliser la dynamique intrinsèque de la variable non observée U. Nous avons accès à des observations partielles, c'est-à-dire à T, jusqu'à t_0 et voulons prévoir l'état complet de t_0 jusqu'au pas de temps final t_f. Nous considérons l'objectif suivant :

$$\min_{G_\theta, h_T^d, h_U} \left\| \hat{U}_{t_f} - \left(\hat{U}_{t_0} + \int_{t_0}^{t_f} h_U(X_t)\, \mathrm{d}t \right) \right\|_2 \text{ subject to } \frac{\mathrm{d}T_t}{\mathrm{d}t} = (h_T^p + h_T^d)(\hat{X}_t)$$

Malheureusement, n'ayant pas accès à la vraie U, nous ne pouvons compter que sur les estimations données par G_θ. Pour l'apprentissage, nous introduisons deux fonctions de coût: nous pénalisons les erreurs de prévision de l'état observé, et nous forçons la variable non observée U à obéir à une dynamique apprise h_U. Dans la suite, nous présentons ces pertes et leur implémentation.

D.4 MODÈLES PHYSICO-STATISTIQUES CONTRAINTS POUR L'IDENTIFICATION ET LA PRÉDICTION DE SYSTÈMES DYNAMIQUES

Nous nous intéressons non seulement à l'apprentissage de la vitesse U physiquement fondée et des trajectoires précises de la température T, mais aussi à la résolution du problème mal posé induit par l'apprentissage de modèles hybrides ML/MB. Nous ne faisons aucune hypothèse ni sur les champs de vitesse U ni sur sa dynamique associée f_U. Rappelons que nous considérons l'état partiellement observé $X_t = (T_t, U_t)$, où la température T est observée et les champs de vitesse U ne sont pas observés. Nous cherchons à modéliser la dynamique f_T de T avec un modèle hybride h_T. Notre proposition consiste à reformuler le problème d'apprentissage

$$\min_{h_T} \mathbb{E}_{s \sim p_S} \| h_T(s) - f_T(s) \|_2 \quad \text{with} \quad h_T = h_T^p + h_T^d$$

en introduisant une borne supérieure sur l'erreur de prédiction d'un modèle physico-statistique. Cela nous permet de contrôler la contribution des composantes physique et statistique à la

prédiction globale. En d'autres termes, nous cherchons à retrouver la qualité de la décomposition et son interprétabilité : alors que h_T^p doit rendre compte du modèle physique, h_T^d représente le résidu non englobé par h_T^p. À partir de cette limite supérieure, nous élaborons un cadre de principe qui généralise les tentatives précédentes de régularisation des modèles hybrides. En particulier, notre proposition va plus loin que Yin, Le Guen, et al. (2021), qui assurent l'unicité dans la décomposition en contraignant la norme de la composante ML. Pour compléter la connaissance dynamique préalable par une composante pilotée par les données et assurer l'interprétabilité de la décomposition. Nous élaborons un cadre de principe qui généralise les tentatives précédentes de régularisation des modèles hybrides. Nous proposons également un nouvel algorithme d'optimisation alternatif pour apprendre des modèles hybrides, pour lequel nous fournissons une analyse de la convergence sur un cas simplifié. Comme notre objectif ultime est de résoudre des problèmes du monde réel, c'est-à-dire les données NATL60, nous proposons une extension de notre cadre pour incorporer des données auxiliaires et nous rapprocher d'un scénario complexe du monde réel. Enfin, nous soulignons que la méthode proposée dans ce chapitre est très générale et peut être appliquée à de nombreuses données physiques.

OPTIMISATION Pour assurer l'identifiabilité, nous dérivons des régularisations sur h_T^p et h_T^d découlant du contrôle d'une borne supérieure de $\mathrm{d}(h_T, f_T)$. En particulier, minimiser $\mathrm{d}(h_T^p, f_T^p)$ nous permettrait d'interpréter avec précision h_T^p comme la vraie f_T^p, et h_T^d comme la dynamique résiduelle f_T^d. Cependant, comme nous n'avons pas accès aux paramètres de f_T^p, le calcul de $\mathrm{d}(h_T^p, f_T^p)$ n'est pas faisable. Nous considérons alors deux situations possibles. Dans la première, la seule information disponible sur le système physique est la forme paramétrique de f_T^p (ou de manière équivalente de h_T^p), la formation ne repose donc que sur les trajectoires observées. Dans le second, on considère les informations auxiliaires disponibles sur f_T^p qui seront utilisées pour minimiser la distance entre h_T^p et f_T^p. Bien que le premier paramètre soit le plus général, l'antériorité physique sur laquelle il repose est souvent insuffisante pour traiter efficacement les situations du monde réel. Le second paramètre utilise des antériorités plus informatives et correspond mieux aux cas réels.

Contrôle de la composante ML et de l'hypothèse MB

Nous proposons une approche générale pour contraindre l'apprentissage de modèles hybrides lorsqu'on a uniquement accès à la forme fonctionnelle de h_T^p. Dans ce cas, pour rendre h_T^p responsable de nos phénomènes observés, une solution est de minimiser $\mathrm{d}(h_T^p, f_T)$. En suivant l'inégalité triangulaire, nous lions les deux erreurs $\mathrm{d}(h_T, f_T)$ et $\mathrm{d}(h_T^p, f_T)$:

$$\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^d, 0) + \mathrm{d}(h_T^p, f_T)$$

Nous voulons que le modèle physico-statistique $h_T = h_T^p + h_T^d$ fournisse des prévisions de haute qualité. Pour ce faire, h_T^p doit expliquer autant que possible les phénomènes observés de manière à capturer la physique dans l'inconnue f_T^p et que h_T^d doit se concentrer sur le résidu de la dynamique qui ne peut être expliqué par la composante physique h_T^p. La minimisation de la seule borne supérieure ne garantit pas un tel objectif, car h_T^d est seulement pénalisé par $\mathrm{d}(h_T^d, 0)$ et n'est pas optimisé pour contribuer aux prévisions. Cependant, la seule minimisation du côté droit de la borne supérieure est insuffisante car h_T^d n'est pas responsable dans les prédictions. Nous proposons donc de minimiser $\mathrm{d}(h_T, f_T)$ tout en contrôlant à la fois $\mathrm{d}(h_T^d, 0)$ et $\mathrm{d}(h_T^p, f_T)$. Un tel contrôle de la borne supérieure revient à équilibrer la contribution des composantes ML et MB.

Introduction de données auxiliaires pour matcher les hypothèses physiques

Nous supposons ici que l'on accède à une version grossière de f_T^p, notée $f_T^{p,pr} \in \mathcal{H}_k$. Plus précisément, nous supposons que $f_T^{p,pr}$ provient d'une dynamique f_T^{pr}, qui est un modèle de première approximation de la véritable dynamique f_T. f_T^{pr} obéit à l'hypothèse de décomposition additive, de sorte que f_T^{pr} et $f_T^{p,pr}$ vérifient $f_T^{pr} = f_T^{p,pr} + f_T^{d,pr}$. Notre objectif est d'adapter notre cadre pour incorporer de telles informations auxiliaires, en faisant entrer la régularisation induite par $f_T^{p,pr}$ dans le cadre du contrôle d'une borne supérieure. Cela nous permet d'étendre notre proposition à la résolution de problèmes physiques du monde réel, encore largement inexplorés par la communauté ML. Avec des calculs similaires, nous avons :

$$\mathrm{d}(h_T, f_T) \leq \mathrm{d}(h_T^d, 0) + \mathrm{d}(h_T^p, f_T^{p,pr}) + \Gamma$$

où $\Gamma = \mathrm{d}(f_T^{pr}, f_T^{p,pr}) + \mathrm{d}(f_T^{pr}, f_T)$ est une constante du problème qui ne peut être optimisée. En effet, elle ne dépend que de f_T^{pr}, $f_T^{p,pr}$ et f_T, variables qui échappent à notre contrôle. Ainsi, un pré-entraînement sur des données auxiliaires de dynamique f_T^{pr} revient à contrôler le terme $\mathrm{d}(h_T, f_T^{pr})$ dans la borne supérieure. Comme ci-dessus, nous proposons de minimiser $\mathrm{d}(h_T, f_T)$ tout en contrôlant à la fois $\mathrm{d}(h_T^d, 0)$ et $\mathrm{d}(h_T^p, f_T^{p,pr})$. À partir des limites supérieures, nous retrouvons le caractère bien posé de l'optimisation et dérivons un schéma d'apprentissage théorique.

RECOUVREMENT DE LA BONNE POSOLOGIE Nous reformulons l'apprentissage mal posé de $\min_{h_T^p, h_T^d \in \mathcal{H}_p \times \mathcal{H}_d} \mathrm{d}(h_T, f_T)$, en optimisant plutôt $\mathrm{d}(h_T, f_T)$ tout en contraignant les bornes supérieures. Définissons $\mathcal{S}_p$ et $\mathcal{S}_d$ comme suit

$$\mathcal{S}_p = \{h_T^p \in \mathcal{H}_p \mid \ell(h_T^p) \leq \mu_p\} \qquad \mathcal{S}_d = \{h_T^d \in \mathcal{H}_d \mid \mathrm{d}(h_T^d, 0) \leq \mu_d\}$$

où μ_p, μ_d sont deux scalaires positifs et $\ell(h_T^p) = \mathrm{d}(h_T^p, f_T)$ dans le cas de la première borne supérieure et $\ell(h_T^p) = \mathrm{d}(h_T^p, f_T^{p,pr})$ dans le cas de la deuxième. Notre proposition revient alors à optimiser $\mathrm{d}(h_T, f_T)$ sur la somme de Minkowski $\mathcal{S}_p + \mathcal{S}_d = \{\, h_T = h_T^p + h_T^d \mid h_T^p \in \mathcal{S}_p, h_T^d \in \mathcal{S}_d \,\}$:

$$\min_{h_T \in \mathcal{S}_p + \mathcal{S}_d} \mathrm{d}(h_T, f_T)$$

Ce cadre d'optimisation sous contrainte nous permet de retrouver la validité du problème d'optimisation sous la compacité relative de la famille de fonctions $\mathcal{H}_p$.

D.5 CONFRONTATION AUX DONNÉES RÉELLES

Nous nous concentrons sur les données NATL60, qui sont des simulations de données réelles. Même si de nombreuses incertitudes inhérentes aux données d'observation réelles ne sont pas présentes, par exemple les incertitudes dues à la couverture nuageuse ou aux appareils de mesure, ces données reflètent étroitement la complexité des observations réelles. Il convient de noter que la dynamique des océans est un phénomène spatial tridimensionnel, faisant intervenir de nombreuses variables à plusieurs échelles interdépendantes. Ainsi, alors que les données simulées utilisées sont bi-dimensionnelles, le phénomène étudié dans cette section est tri-dimensionnel. Pour des raisons de commodité, nous nous concentrons uniquement sur les champs de vitesse de surface, et ignorons les composantes verticales.

Nous cherchons à confronter des modèles pilotés par les données à des données de type réel. Dans cette optique, nous ne prétendons pas résoudre un problème du domaine de l'océanographie.

Nous cherchons plutôt à détecter les limites de nos modèles théoriques, lorsqu'ils sont confrontés à des dynamiques complexes. Ainsi, nous adaptons les méthodologies présentées aux simulations de données réelles. Nous analysons les performances du modèle introduit précédemment sur NATL60. Nous présentons et discutons ensuite les différentes tentatives d'adaptation de notre modèle à la complexité du NATL60. Enfin, des directions de recherche sont proposées pour améliorer les résultats et ouvrir la voie au développement de nouveaux modèles.

BIBLIOGRAPHY

Abbe, Cleveland (1901). "The physical basis of long-range weather forecasts". In: *Monthly Weather Review* 29.12, pp. 551–561.

Agapiou, Athos (2017). "Remote sensing heritage in a petabyte-scale: satellite data and heritage Earth Engine© applications". In: *International Journal of Digital Earth* 10.1, pp. 85–102.

Ajayi, Adekunle et al. (2020). "Spatial and temporal variability of the North Atlantic eddy field from two kilometric-resolution ocean models". In: *Journal of Geophysical Research: Oceans* 125.5, e2019JC015827.

Alber, Mark et al. (2019). "Integrating machine learning and multiscale modeling—perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences". In: *NPJ digital medicine* 2.1, pp. 1–11.

Alvarez, Mauricio A, David Luengo, and Neil D Lawrence (2013). "Linear latent force models using Gaussian processes". In: *IEEE transactions on pattern analysis and machine intelligence* 35.11, pp. 2693–2705.

"Tracer Inverse Problems" (1989). In: *Oceanic Circulation Models: Combining Data and Dynamics*. Ed. by David L. T. Anderson and Jürgen Willebrand. Dordrecht: Springer Netherlands, pp. 1–77. ISBN: 978-94-009-1013-3. DOI: 10.1007/978-94-009-1013-3_1. URL: https://doi.org/10.1007/978-94-009-1013-3_1.

Ardizzone, Lynton et al. (2019). "Analyzing Inverse Problems with Invertible Neural Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rJed6j0cKX.

Ayed, Ibrahim et al. (2020). "Learning the Spatio-Temporal Dynamics of Physical Processes from Partial Observations". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3232–3236.

Barros, Saulo R. M. et al. (1995). "The IFS model: A parallel production weather code". In: *Parallel Computing* 21.10, pp. 1621–1638.

Bauer, Peter, Alan Thorpe, and Gilbert Brunet (2015). "The quiet revolution of numerical weather prediction". In: *Nature* 525.7567, pp. 47–55.

Belbute-Peres, Filipe de Avila, Thomas Economon, and Zico Kolter (2020). "Combining differentiable PDE solvers and graph neural networks for fluid flow prediction". In: *International Conference on Machine Learning*. PMLR, pp. 2402–2411.

Bergen, Karianne J et al. (2019). "Machine learning for data-driven discovery in solid Earth geoscience". In: *Science* 363.6433, eaau0323.

Bezenac, Emmanuel de, Arthur Pajot, and Patrick Gallinari (2019). "Deep learning for physical processes: Incorporating prior scientific knowledge". In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.12, p. 124009.

Bi, Kaifeng et al. (2022). "Pangu-Weather: A 3D High-Resolution Model for Fast and Accurate Global Weather Forecast". In: *arXiv preprint arXiv:2211.02556*.

Bigg, GR and PD Killworth (1988). "Conservative tracers and the ocean circulation". In: *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 325.1583, pp. 177–187.

Bihouix, Philippe (2014). *L'Âge des low tech. Vers une civilisation techniquement soutenable: Vers une civilisation techniquement soutenable*. Média Diffusion.

Bihouix, Philippe (2019). *Le bonheur était pour demain: les rêveries d'un ingénieur solitaire*. Éditions du Seuil.

Bihouix, Philippe and Benoit De Guillebon (2021). "Quel futur pour les métaux?" In: *Quel futur pour les métaux?* EDP sciences.

Boffetta, Guido et al. (2001). "Detecting barriers to transport: a review of different techniques". In: *Physica D: Nonlinear Phenomena* 159.1-2, pp. 58–70.

Bolton, Thomas and Laure Zanna (2019). "Applications of deep learning to ocean data inference and subgrid parameterization". In: *Journal of Advances in Modeling Earth Systems* 11.1, pp. 376–399.

Bowman, John C, Mohammad Ali Yassaei, and Anup Basu (2015). "A fully Lagrangian advection scheme". In: *Journal of Scientific Computing* 64.1, pp. 151–177.

Boyd, Stephen, Stephen P Boyd, and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge university press.

Brandstetter, Johannes, Daniel Worrall, and Max Welling (2022). "Message passing neural PDE solvers". In: *arXiv preprint arXiv:2202.03376*.

Bryan, Kirk (1969). "A numerical method for the study of the circulation of the world ocean". In: *Journal of Computational Physics* 4.3, pp. 347–376.

Butcher, John Charles (2016). *Numerical methods for ordinary differential equations*. John Wiley & Sons.

Cai, Shengze et al. (2022). "Physics-informed neural networks (PINNs) for fluid mechanics: A review". In: *Acta Mechanica Sinica*, pp. 1–12.

Chantry, Matthew et al. (2021). "Opportunities and challenges for machine learning in weather and climate modelling: hard, medium and soft AI". In: *Philosophical Transactions of the Royal Society A* 379.2194, p. 20200083.

Chassignet, Eric P, Julien Le Sommer, and Alan J Wallcraft (2019). "General circulation models". In: *Encyclopedia of Ocean Sciences, 3rd edn., edited by: Cochran, KJ, Bokuniewicz, HJ, and Yager, PL* 5, pp. 486–490.

Chen, Ricky T. Q. et al. (2018). "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems 31*. Ed. by Samy Bengio et al. Curran Associates, Inc., pp. 6571–6583.

Collobert, Ronan and Jason Weston (2008). "A unified architecture for natural language processing: Deep neural networks with multitask learning". In: *Proceedings of the 25th international conference on Machine learning*, pp. 160–167.

Courtier, PHILIPPE, J-N Thépaut, and Anthony Hollingsworth (1994). "A strategy for operational implementation of 4D-Var, using an incremental approach". In: *Quarterly Journal of the Royal Meteorological Society* 120.519, pp. 1367–1387.

Cranmer, Miles et al. (2020). "Lagrangian Neural Networks". In: *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*.

Crutchfield, James P and BS McNamara (1987). "Equations of motion from a data series". In: *Complex systems* 1, pp. 417–452.

De Boer, MA and K Lammertsma (2013). "Scarcity of rare earth elements". In: *ChemSusChem* 6.11, pp. 2045–2055.

Déchelle, Marie et al. (2020). "Bridging Dynamical Models and Deep Networks to Solve Forward and Inverse Problems". In: *NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning*.

Demailly, Jean-Pierre (2016). *Analyse numérique et équations différentielles-4eme Ed*. EDP sciences.

Dohan, Kathleen and Nikolai Maximenko (2010). "Monitoring ocean currents with satellite sensors". In: *Oceanography* 23.4, pp. 94–103.

Dona*, Jérémie, Marie Déchelle*, Marina Levy, et al. (2021). "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction". In: *International Conference on Learning Representations*.

Droniou, Jérôme (Apr. 2001). "Intégration et Espaces de Sobolev à Valeurs Vectorielles." working paper or preprint. URL: https://hal.archives-ouvertes.fr/hal-01382368.

Dueben, Peter D and Peter Bauer (2018). "Challenges and design choices for global weather and climate models based on machine learning". In: *Geoscientific Model Development* 11.10, pp. 3999–4009.

Dupont, Emilien, Arnaud Doucet, and Yee Whye Teh (2019). "Augmented neural odes". In: *Advances in Neural Information Processing Systems* 32.

Edenhofer, Ottmar et al. (2011). "IPCC special report on renewable energy sources and climate change mitigation". In: *Prepared By Working Group III of the Intergovernmental Panel on Climate Change, Cambridge University Press, Cambridge, UK*.

England, Matthew H and Ernst Maier-Reimer (2001). "Using chemical tracers to assess ocean models". In: *Reviews of Geophysics* 39.1, pp. 29–70.

Esaias, Wayne E et al. (1998). "An overview of MODIS capabilities for ocean science observations". In: *IEEE Transactions on Geoscience and Remote Sensing* 36.4, pp. 1250–1265.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017). "Model-agnostic meta-learning for fast adaptation of deep networks". In: *International conference on machine learning*. PMLR, pp. 1126–1135.

Forssell, U. and P. Lindskog (1997). "Combining Semi-Physical and Neural Network Modeling: An Example ofIts Usefulness". In: *IFAC Proceedings Volumes* 30.11. IFAC Symposium on System Identification (SYSID'97), Kitakyushu, Fukuoka, Japan, 8-11 July 1997, pp. 767–770. ISSN: 1474-6670. DOI: https://doi.org/10.1016/S1474-6670(17)42938-7.

Frankignoul, Claude (1985). "Sea surface temperature anomalies, planetary waves, and air-sea feedback in the middle latitudes". In: *Reviews of geophysics* 23.4, pp. 357–390.

Frezat, Hugo et al. (2022). "A posteriori learning for quasi-geostrophic turbulence parametrization". In: *arXiv preprint arXiv:2204.03911*.

Funahashi, Ken-ichi and Yuichi Nakamura (1993). "Approximation of dynamical systems by continuous time recurrent neural networks". In: *Neural networks* 6.6, pp. 801–806.

Ganguly, AR et al. (2014). "Toward enhanced understanding and projections of climate extremes using physics-guided data mining techniques". In: *Nonlinear Processes in Geophysics* 21.4, pp. 777–795.

Geneva, Nicholas and Nicholas Zabaras (2020). "Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks". In: *Journal of Computational Physics* 403, p. 109056. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2019.109056.

Gholami, Amir, Kurt Keutzer, and George Biros (2019). "Anode: Unconditionally accurate memory-efficient gradients for neural odes". In: *arXiv preprint arXiv:1902.10298*.

Gomez, Aidan N et al. (2017). "The reversible residual network: Backpropagation without storing activations". In: *Advances in neural information processing systems* 30.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Gouk, Henry et al. (2021). "Regularisation of neural networks by enforcing lipschitz continuity". In: *Machine Learning* 110.2, pp. 393–416.

Greydanus, Samuel, Misko Dzamba, and Jason Yosinski (2019). "Hamiltonian neural networks". In: *Advances in Neural Information Processing Systems*, pp. 15353–15363.

Groenke, Brian, Luke Madaus, and Claire Monteleoni (2020). "ClimAlign: Unsupervised statistical downscaling of climate variables via normalizing flows". In: *Proceedings of the 10th International Conference on Climate Informatics*, pp. 60–66.

Haber, Eldad and Lars Ruthotto (2017). "Stable architectures for deep neural networks". In: *Inverse problems* 34.1, p. 014004.

Hamdi, Samir, William E Schiesser, and Graham W Griffiths (2007). "Method of lines". In: *Scholarpedia* 2.7, p. 2859.

He, Kaiming et al. (2016). "Identity mappings in deep residual networks". In: *European conference on computer vision*. Springer, pp. 630–645.

Henderson, Peter et al. (2020). "Towards the systematic reporting of the energy and carbon footprints of machine learning". In: *Journal of Machine Learning Research* 21.248, pp. 1–43.

Hewitt, Helene et al. (2022). "The small scales of the ocean may hold the key to surprises". In: *Nature Climate Change* 12.6, pp. 496–499.

Isola, Phillip et al. (2017a). "Image-to-Image Translation with Conditional Adversarial Networks". In: *CVPR*.

— (2017b). "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134.

Jaderberg, Max et al. (2015). "Spatial Transformer Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2015/file/33ceb07bf4eeb3da587e268d663aba1a-Paper.pdf.

Jia, Xiaowei, Jared Willard, Anuj Karpatne, Jordan Read, Jacob Zwart, Michael Steinbach, et al. (2019). "Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles". In: *Proceedings of the 2019 SIAM International Conference on Data Mining*. SIAM, pp. 558–566.

Jia, Xiaowei, Jared Willard, Anuj Karpatne, Jordan Read, Jacob Zwart, Michael S Steinbach, et al. (2019). "Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles". English (US). In: *SIAM International Conference on Data Mining, SDM 2019*. SIAM International Conference on Data Mining, SDM 2019. Society for Industrial and Applied Mathematics Publications, pp. 558–566. DOI: 10.1137/1.9781611975673.63.

Jia, Xiaowei, Jacob Zwart, et al. (2021). "Physics-guided recurrent graph model for predicting flow and temperature in river networks". In: *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*. SIAM, pp. 612–620.

Jin, Xiaowei et al. (2021). "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations". In: *Journal of Computational Physics* 426, p. 109951.

Jordan, Michael I and Tom M Mitchell (2015). "Machine learning: Trends, perspectives, and prospects". In: *Science* 349.6245, pp. 255–260.

Karkar, Skander et al. (2020). *A Principle of Least Action for the Training of Neural Networks*.

Karpatne, Anuj et al. (2017). "Theory-guided data science: A new paradigm for scientific discovery from data". In: *IEEE Transactions on knowledge and data engineering* 29.10, pp. 2318–2331.

Kashinath, K et al. (2021). "Physics-informed machine learning: case studies for weather and climate modelling". In: *Philosophical Transactions of the Royal Society A* 379.2194, p. 20200093.

Kingma, Diederik P and Max Welling (2013). "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114*.

Kirchmeyer, Matthieu et al. (2022). "Generalizing to New Physical Systems via Context-Informed Dynamics Model". In: *arXiv preprint arXiv:2202.01889*.

Kochkov, Dmitrii et al. (2021). "Machine learning–accelerated computational fluid dynamics". In: *Proceedings of the National Academy of Sciences* 118.21, e2101784118.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25.

Lagaris, Isaac E, Aristidis Likas, and Dimitrios I Fotiadis (1998). "Artificial neural networks for solving ordinary and partial differential equations". In: *IEEE transactions on neural networks* 9.5, pp. 987–1000.

Lam, Remi et al. (2022). "GraphCast: Learning skillful medium-range global weather forecasting". In: *arXiv preprint arXiv:2212.12794*.

Lamont, Tarron, Raymond G Barlow, and Robert JW Brewin (2018). "Variations in Remotely-Sensed Phytoplankton Size Structure of a Cyclonic Eddy in the Southwest Indian Ocean". In: *Remote Sensing* 10.7, p. 1143.

Lange, Kenneth, Joong-Ho Won, and Jason Xu (June 2019). "Projection onto Minkowski Sums with Application to Constrained Learning". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 3642–3651.

Laplace, Pierre Simon (1799). *Traité de mécanique céleste*. Vol. 1. de l'Imprimerie de Crapelet.

Larsson, Gustav, Michael Maire, and Gregory Shakhnarovich (2016). "Fractalnet: Ultra-deep neural networks without residuals". In: *arXiv preprint arXiv:1605.07648*.

Le Dimet, François-Xavier and Olivier Talagrand (1986). "Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects". In: *Tellus A: Dynamic Meteorology and Oceanography* 38.2, pp. 97–110.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep learning". In: *nature* 521.7553, pp. 436–444.

Lee, Hyuk and In Seok Kang (1990). "Neural algorithm for solving differential equations". In: *Journal of Computational Physics* 91.1, pp. 110–131.

Lee, Seungjun, Haesang Yang, and Woojae Seong (2021). "Identifying Physical Law of Hamiltonian Systems via Meta-Learning". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=45NZvF1UHam.

Leung, Michael KK et al. (2014). "Deep learning of the tissue-regulated splicing code". In: *Bioinformatics* 30.12, pp. i121–i129.

Levin, Lisa A et al. (2019). "Global observing needs in the deep ocean". In: *Frontiers in Marine Science* 6, p. 241.

Lévy, Marina, Patrice Klein, and Anne-Marie Treguier (2001). "Impact of sub-mesoscale physics on production and subduction of phytoplankton in an oligotrophic regime". In: *Journal of marine research* 59.4, pp. 535–565.

Lewis, Adrian S. and Jérôme Malick (2008). "Alternating Projections on Manifolds". In: *Mathematics of Operations Research* 33.1, pp. 216–234. DOI: 10.1287/moor.1070.0291.

Li, Zhen and Zuoqiang Shi (2017). "Deep Residual Learning and PDEs on Manifold". In: *CoRR* abs/1708.05115. arXiv: 1708.05115. URL: http://arxiv.org/abs/1708.05115.

Ling, Julia, Andrew Kurzawski, and Jeremy Templeton (2016). "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance". In: *Journal of Fluid Mechanics* 807, pp. 155–166.

Linial, Ori et al. (2021). "Generative ODE modeling with known unknowns". In: *Proceedings of the Conference on Health, Inference, and Learning*, pp. 79–94.

Long, Yun, Xueyuan She, and Saibal Mukhopadhyay (2018). "HybridNet: integrating model-based and data-driven learning to predict evolution of dynamical systems". In: *Conference on Robot Learning*. PMLR, pp. 551–560.

Long, Zichao, Yiping Lu, and Bi Dong (2019). "PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network". In: *Journal of Computational Physics* 399, p. 108925.

Long, Zichao, Yiping Lu, and Bin Dong (2019). "PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network". In: *Journal of Computational Physics* 399, p. 108925.

Long, Zichao, Yiping Lu, Xianzhong Ma, et al. (July 2018a). "PDE-Net: Learning PDEs from Data". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, pp. 3208–3216.

— (2018b). "Pde-net: Learning pdes from data". In: *International Conference on Machine Learning*, pp. 3208–3216.

Lu, Yiping et al. (2018). "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations". In: *International Conference on Machine Learning*. PMLR, pp. 3276–3285.

Lynch, Peter (2008). "The origins of computer weather prediction and climate modeling". In: *Journal of computational physics* 227.7, pp. 3431–3444.

Madec, Gurvan et al. (2017). "NEMO ocean engine". In.

Masson-Delmotte, Valérie et al. (2018). *Global warming of 1.5 C. An IPCC Special Report on the impacts of global warming*.

Mathieu, Michael, Camille Couprie, and Yann LeCun (2015). "Deep multi-scale video prediction beyond mean square error". In: *arXiv preprint arXiv:1511.05440*.

Mehta, Viraj et al. (2020). *Neural Dynamical Systems: Balancing Structure and Flexibility in Physical Prediction*. arXiv: 2006.12682 [cs.LG].

Mirza, Mehdi and Simon Osindero (2014). "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784*.

Monteleoni, Claire et al. (2013). *Climate informatics*. Tech. rep. CRC Press.

Navier, CLMH (1823). "Mémoire sur les lois du mouvement des fluides". In: *Mémoires de l'Académie Royale des Sciences de l'Institut de France* 6.1823, pp. 389–440.

Neumann, John von (1950). *Functional Operators (AM-22), Volume 2: The Geometry of Orthogonal Spaces*. Princeton University Press. ISBN: 978-1-4008-8189-5. DOI: https://doi.org/10.1515/9781400881895.

NOAA (2021). *National Ocean Service website*. URL: https://oceanservice.noaa.gov/facts/gyre.html.

— (2022). *National Ocean Service website*. URL: https://oceanservice.noaa.gov/education/tutorial_currents/04currents1.html.

Onken, Derek and Lars Ruthotto (2020). "Discretize-optimize vs. optimize-discretize for time-series regression and continuous normalizing flows". In: *arXiv preprint arXiv:2005.13420*.

Paszke, Adam et al. (2019). "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems* 32.

Pfaff, Tobias et al. (2020). "Learning mesh-based simulation with graph networks". In: *arXiv preprint arXiv:2010.03409*.

Pontryagin, Lev Semenovich (1987). *Mathematical theory of optimal processes*. CRC press.

Pörtner, Hans-Otto et al. (2022). "Climate change 2022: Impacts, adaptation and vulnerability". In: *IPCC Sixth Assessment Report*, pp. 37–118.

Psichogios, Dimitris C. and Lyle H. Ungar (1992). "A hybrid neural network-first principles approach to process modeling". In: *AIChE Journal* 38.10, pp. 1499–1511. DOI: https://doi.org/10.1002/aic.690381003. eprint: https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.690381003. URL: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003.

Raissi, Maziar (2018). "Deep hidden physics models: Deep learning of nonlinear partial differential equations". In: *The Journal of Machine Learning Research* 19.1, pp. 932–955.

Raissi, Maziar, Hessam Babaee, and Peyman Givi (Dec. 2019). "Deep learning of turbulent scalar mixing". In: *Phys. Rev. Fluids* 4 (12), p. 124501. DOI: 10.1103/PhysRevFluids.4.124501. URL: https://link.aps.org/doi/10.1103/PhysRevFluids.4.124501.

Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 378, pp. 686–707.

Reichstein, Markus et al. (2019). "Deep learning and process understanding for data-driven Earth system science". In: *Nature* 566.7743, pp. 195–204.

Rico-Martinez, R, JS Anderson, and IG Kevrekidis (1994). "Continuous-time nonlinear signal processing: a neural network based approach for gray box identification". In: *Proceedings of IEEE Workshop on Neural Networks for Signal Processing*. IEEE, pp. 596–605.

Rio, M-H and R Santoleri (2018). "Improved global surface currents from the merging of altimetry and sea surface temperature data". In: *Remote sensing of Environment* 216, pp. 770–785.

Roemmich, Dean et al. (2009). "The Argo Program: Observing the global ocean with profiling floats". In: *Oceanography* 22.2, pp. 34–43.

Rolnick, David et al. (2022). "Tackling climate change with machine learning". In: *ACM Computing Surveys (CSUR)* 55.2, pp. 1–96.

Romm, Joseph (2022). *Climate change: What everyone needs to know*. Oxford University Press.

Ruthotto, Lars and Eldad Haber (2020). "Deep neural networks motivated by partial differential equations". In: *Journal of Mathematical Imaging and Vision* 62.3, pp. 352–364.

Sabatier, Pierre C (2000). "Past and future of inverse problems". In: *Journal of Mathematical Physics* 41.6, pp. 4082–4124.

Saemundsson, Steindor et al. (2020). "Variational integrator networks for physically structured embeddings". In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 3078–3087.

Saha, Priyabrata, Saurabh Dash, and Saibal Mukhopadhyay (2020). "PhICnet: Physics-incorporated convolutional recurrent neural networks for modeling dynamical systems". In: *arXiv preprint arXiv:2004.06243*.

San, Omer and Romit Maulik (Apr. 2018). "Machine learning closures for model order reduction of thermal fluids". In: *Applied Mathematical Modelling* 60. DOI: 10.1016/j.apm.2018.03.037.

Schultz, Martin G et al. (2021). "Can deep learning beat numerical weather prediction?" In: *Philosophical Transactions of the Royal Society A* 379.2194, p. 20200097.

Shi, Xingjian et al. (2015). "Convolutional LSTM network: A machine learning approach for precipitation nowcasting". In: *Advances in neural information processing systems* 28.

Silver, David et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, pp. 484–489.

Sinha, Anirban and Ryan Abernathey (2021). "Estimating Ocean Surface Currents With Machine Learning". In: *Frontiers in Marine Science* 8, p. 672477.

Sirignano, Justin and Konstantinos Spiliopoulos (2018). "DGM: A deep learning algorithm for solving partial differential equations". In: *Journal of computational physics* 375, pp. 1339–1364.

Sitzmann, Vincent et al. (2020). "Implicit neural representations with periodic activation functions". In: *Advances in Neural Information Processing Systems* 33, pp. 7462–7473.

Stevens, Rick et al. (2020). *AI for Science: Report on the Department of Energy (DOE) Town Halls on Artificial Intelligence (AI) for Science*. Tech. rep. Argonne National Lab.(ANL), Argonne, IL (United States).

Stewart, Russell and Stefano Ermon (2017). "Label-free supervision of neural networks with physics and domain knowledge". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 31.

Stockhause, Martina and Michael Lautenschlager (2017). "CMIP6 data citation of evolving data". In: *Data Science Journal* 16.

Stokes, Georges Gabriel (1880). "On the theories of the internal friction of fluids in motion, and of the equilibrium and motion of elastic solids". In: *Transactions of the Cambridge Philosophical Society* 8.

Szegedy, Christian et al. (2017). "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-first AAAI conference on artificial intelligence*.

Tait, Daniel J and Theodoros Damoulas (2020). "Variational Autoencoding of PDE Inverse Problems". In: *arXiv preprint arXiv:2006.15641*.

Thompson, Michael L and Mark A Kramer (1994). "Modeling chemical processes using prior knowledge and neural networks". In: *AIChE Journal* 40.8, pp. 1328–1340.

Tompson, Jonathan et al. (Aug. 2017). "Accelerating Eulerian Fluid Simulation With Convolutional Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 3424–3433.

Toth, Peter et al. (2020). "Hamiltonian Generative Networks". In: *International Conference on Learning Representations*.

Um, Kiwon et al. (2020). "Solver-in-the-Loop: Learning from Differentiable Physics to Interact with Iterative PDE-Solvers". In: *Neural Information Processing Systems (NeurIPS)*. Vol. 33, pp. 6111–6122.

Vallis, Geoffrey K (2011). "Climate and the Oceans". In: *Climate and the Oceans*. Princeton University Press.

Verma, Siddhartha, Yuan Xuan, and Guillaume Blanquart (2014). "An improved bounded semi-Lagrangian scheme for the turbulent transport of passive scalars". In: *Journal of Computational Physics* 272, pp. 1–22.

Vidal, Olivier (2018). "Ressources minérales, progrès technologique et croissance". In: *Temporalités. Revue de sciences sociales et humaines* 28.

Wan, Zhong Yi et al. (2018). "Data-assisted reduced-order modeling of extreme events in complex dynamical systems". In: *PloS one* 13.5, e0197704.

Wandel, Nils, Michael Weinmann, and Reinhard Klein (2021). "Learning Incompressible Fluid Dynamics from Scratch - Towards Fast, Differentiable Fluid Models that Generalize". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=KUDUoRsEphu.

Wang, Jindong et al. (2022). "Generalizing to unseen domains: A survey on domain generalization". In: *IEEE Transactions on Knowledge and Data Engineering*.

Watson-Parris, Duncan (2021). "Machine learning for weather and climate are worlds apart". In: *Philosophical Transactions of the Royal Society A* 379.2194, p. 20200098.

Weinan, E (2017). "A proposal on machine learning via dynamical systems". In: *Communications in Mathematics and Statistics* 1.5, pp. 1–11.

Willard, Jared et al. (2020). "Integrating physics-based modeling with machine learning: A survey". In: *arXiv preprint arXiv:2003.04919*.

Wu, Yonghui et al. (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144*.

Yin, Yuan, Ibrahim Ayed, et al. (2021). "LEADS: Learning dynamical systems that generalize across environments". In: *Advances in Neural Information Processing Systems* 34, pp. 7561–7573.

Yin, Yuan, Vincent Le Guen, et al. (2021). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *ICLR*.

Young, Chih-Chieh, Wen-Cheng Liu, and Ming-Chang Wu (2017). "A physically based and machine learning hybrid approach for accurate rainfall-runoff modeling during extreme typhoon events". In: *Applied Soft Computing* 53, pp. 205–216.

Zaeemzadeh, Alireza, Nazanin Rahnavard, and Mubarak Shah (2020). "Norm-preservation: Why residual networks can become extremely deep?" In: *IEEE transactions on pattern analysis and machine intelligence* 43.11, pp. 3980–3990.

Zanna, Laure and Thomas Bolton (2021). "Deep learning of unresolved turbulent ocean processes in climate models". In: *Deep Learning for the Earth Sciences: A Comprehensive Approach to Remote Sensing, Climate Science, and Geosciences*, pp. 298–306.

Zhang, Chiyuan et al. (2021). "Understanding deep learning (still) requires rethinking generalization". In: *Communications of the ACM* 64.3, pp. 107–115.

Zhang, Xingcheng et al. (2017). "Polynet: A pursuit of structural diversity in very deep networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 718–726.

Zhou, Jie et al. (2020). "Graph neural networks: A review of methods and applications". In: *AI Open* 1, pp. 57–81.

Zhuang, Juntang et al. (2020). "Adaptive checkpoint adjoint method for gradient estimation in neural ode". In: *International Conference on Machine Learning*. PMLR, pp. 11639–11649.