Statistical learning of physical dynamics

Jérémie Donà

SCIENCES SORBONNE UNIVERSITÉ

THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ

Spécialité **Informatique**

École Doctorale Informatique, Télécommunications et Électronique (Paris)

Statistical Learning of Physical Dynamics

Apprentissage Statistique de Dynamiques Physiques

Présentée par

Jérémie Donà

Dirigée par

Patrick Gallinari, Gérard Biau

Pour obtenir le grade de

DOCTEUR de SORBONNE UNIVERSITÉ

Présentée et soutenue publiquement le 25 mai 2022

Devant le jury composé de :

Marc SCHOENOER *Professeur, INRIA – Equipe Tau*	Rapporteur
Amaury HABRARD *Professeur, Université Jean Monnet – Laboratoire Hubert Curien*	Rapporteur
Anna KORBA *Professeure Assistante, ENSAE ParisTech – CREST*	Examinatrice
Nicolas BOUSQUET *Professeur Associé, Sorbonne Université – LPSM*	Examinateur
Gérard BIAU *Professeur, Sorbonne Université – LPSM*	Co-directeur de thèse
Patrick GALLINARI *Professeur, Sorbonne Université – ISIR, MLIA*	Directeur de Thèse

ABSTRACT

The modeling of natural processes relies on a physical description that prescribes the changes in the state of the studied system. The use of domain specific knowledge about the system allows the translation of physical principles into models, which are then validated by experimental data. With its successes in many domain like text generation and image classification, machine learning, and in particular deep learning, has become a powerful tool for the modeling of physical processes, thanks to the significant increase in the amount of data available from sensors. Nonetheless, statistical learning of physical processes by a sole data-driven approach suffers from several limitations such as interpretation difficulties, stability during the learning phase and reduced generalization capabilities.

A major objective of this work is to provide tools in order to perform data-driven learning of physical processes and more generally spatio-temporal systems. In particular, we will study spatio-temporal phenomena which dynamics obey a differential equation. More precisely, we focus on incorporating domain and physical knowledge in learning algorithms. Depending on the application, an approximation of the differential equation at stake is often accessible; the complement remains to be estimated by a neural network. This leads us to study hybrid physical-statistical systems for the modeling of physical processes. Thus, we will identify the problems related to the formulation of hybrid dynamics learning and then propose a framework including constraints adapted to deep networks to improve the interpretability and the performance of the learned algorithms.

Conversely, physics and the theory of dynamical systems have provided numerous tools to improve and better understand statistical models. However, neural networks, although efficient for a large number of tasks, remain qualified as "black boxes" because they are not interpretable. The black box behavior of neural networks is particularly true for complex task such as text generation or video predictions. Thus, we will attempt to open the black box and propose more interpretable neural network architectures with increased generalization performances for the modeling of spatio-temporal systems.

RÉSUMÉ

La modélisation de processus naturels repose en large partie sur une description physique qui prescrit les changements dans l'état observé du système. L'utilisation de connaissances spécifiques relatives au système permet la traduction de principes physiques en modèles, ensuite validés par des données expérimentales. Fort de succès dans de nombreux domaines comme la génération de texte et la classification d'images, l'apprentissage machine et en particulier profond, est devenu un outil puissant pour la modélisation de processus physiques grâce à l'augmentation significative de la quantité de données disponibles provenant de capteurs. Toutefois, l'apprentissage statistique de processus physiques par une approche uniquement guidée par les données souffre de plusieurs limites comme les difficultés d'interprétation, l'instabilité lors de la phase d'apprentissage et les capacités de généralisation réduites.

Un objectif majeur de ce travail réside dans la construction d'outils permettant la prédiction de systèmes physiques, ou plus généralement spatio-temporels. En particulier, nous étudierons les systèmes spatio-temporels dont l'évolution des observations obéit à une équation différentielle. Plus précisément, il sera question d'incorporer de la connaissance physique à priori dans les algorithmes d'apprentissage. En effet, en fonction du domaine d'application, une première estimation, peut-être grossière, de la dynamique du système considéré est souvent accessible ; le complément restant à être estimé à partir de réseaux de neurones. Cela nous pousse donc à étudier les systèmes hybrides physiques-statistiques pour la modélisation de processus physiques. Ainsi, nous identifierons les problèmes liés à l'apprentissage de dynamiques hybrides puis proposerons un cadre et des contraintes adaptés aux réseaux profonds afin d'améliorer l'interprétabilité et la performance des algorithmes ainsi appris.

Réciproquement, la physique et l'étude des systèmes dynamiques ont proposé nombre d'outils pour améliorer le fonctionnement et la compréhension de modèles statistiques. Toutefois, les réseaux de neurones, bien qu'efficaces pour un grand nombre de tâches, restent qualifiés de "boîte-noire" car non interprétables. Le comportement de "boîte-noire" des réseaux neuronaux est particulièrement vrai pour les tâches complexes telles que la génération de textes ou les prédictions vidéo. Ainsi, nous tenterons d'ouvrir la boîte noire et de proposer des architectures de réseaux de neurones plus interprétables et aux capacités de généralisation accrues pour la prédiction de systèmes spatio-temporels.

REMERCIEMENTS

À présent au terme de cette thèse, je réalise que ce fût une aventure formidable et riche de rencontres. Ainsi, ceux qui m'ont aidé, soutenu et encouragé sont nombreux et il me serait impossible de tous les nommer ici.

Je remercie en premier lieu les membres de mon jury ayant accepté d'évaluer mon travail : Anna Korba, Nicolas Bousquet ainsi que Amauray Habrard et Marc Schoenauer qui ont accepté le rôle de rapporteur.

Je tiens aussi à exprimer ma reconnaissance envers mes directeurs de thèse, Gérard Biau et Patrick Gallinari pour cette opportunité. Ils ont su guider mes travaux et m'apporter leurs conseils et leur expérience; cette thèse n'aurait pas été possible sans leur aide et soutien. Je remercie les chercheurs de l'équipe, notamment Sylvain et Olivier, qui ont fait progresser ma réflexion par leurs encouragements. Enfin, je tiens à remercier Christophe et Nadine qui fournissent aux doctorants une aide irremplaçable. Merci aussi à tous les membres du MLIA, permanents ou non qui durant cette thèse en période de restriction sanitaire furent un appui sans faille. Un grand merci pêle-mêle à tous mes co-auteurs : Emmanuel, Ibrahim, Jean-Yves, Marie, Matthieu, Vincent, Yuan, avec qui j'ai grandement apprécié réfléchir et avancer sur nos recherches, et grâce à qui j'ai mûri professionnellement et humainement. Merci à Agnès, Arthur, Clément, Edouard, Etienne, Mickaël, Thomas et tous les doctorants que j'ai eu la chance de côtoyer pour les cafés, les babyfoots endiablés et les moments de convivialité au laboratoire et en dehors.

J'adresse enfin mes remerciements à mes proches et amis, dont l'affection m'a aidé à vivre pleinement et avec sérénité cette expérience, malgré l'angoissante (et récurrente) question : "Tu soutiens quand ?". Un grand merci à Alain, Amine, Anna, Antoine, Antonin, Arnault, Behrang, Clément, Dersy, Guillaume, Jean, Julien, Maëlle, Nicolas, Marie, Max, Paul, Minh-Hanh et les Chollet-Ta, et tous ceux que j'oublie pour ces précieux moments de vie partagés, les discussions animées, les vacances et week-ends, qui m'ont rappelé qu'il n'y a pas que la thèse dans la vie. À Nicolas, Gladys, Maxime, Lucas, Jonathan merci d'avoir illuminé cette étrange année 2021 au cours de sessions inoubliables. Merci à Nhan et Minh pour leur aide et recommandations.

Un immense merci à Mai-Thi, qui en plus de son indéfectible affection quotidienne, m'a donné l'occasion de m'épanouir scientifiquement, de m'enrichir émotionnellement et de m'ouvrir sur le monde. Son soutien est inestimable.

Mes dernières pensées se tournent vers ma famille. Mes grands parents, Ginette et Filiberto, sont une source d'inspiration et de réconfort inépuisable. Merci à Elena et Sante, qui ont grandement contribué à ce que je suis, j'aurais aimé qu'ils puissent aussi voir ce jour. Merci à ma soeur Sara, et ses enfants Andrea et Nora pour leur présence chaleureuse. Enfin, merci à mes parents, infaillibles piliers durant toutes ces années, je leur dois tout.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

BPTT	BackPropagation Through Time
ConvNet	Convolutional Neural Network
DL	Deep Learning
DNN	Deep Neural Network
ELBO	Evidence Lower Bound
EM	Expectation-Maximization
GAN	Generative Adversarial Networks
GD	Gradient Descent
GNN	Graph Neural Network
GPU	Graphics Processing Unit
GRU	Gated Reccurent Unit
LSTM	Long Short Term Memory
ML	Machine Learning
MLP	Multi-Layer Perceptron
NLP	Natural Language Processing
NN	Neural Network
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
ResNet	Residual Network
RNN	Recurrent Neural Network
SDE	Stochastic Differential Equation
SGD	Stochastic Gradient Descent
SMC	Sequential Monte Carlo
SST	Sea Surface Temperature
VAE	Variational Auto-Encoder

Part I

INTRODUCTION

CHAPTER 1

CONTEXT

Thanks to the increase in the availability of adapted computational power such as Graphics Processing Unit (GPU) and of data sources, along with the advances in optimized software, the use of Deep Neural Network (DNN) for the Machine Learning (ML) community has become standard in various tasks such as classification and regression. Indeed, Deep Learning (DL) performances scale better than most traditional approaches such as Support Vector Machines, or Linear Regression. In that perspective, traditional classification (ImageNet, CIFAR, MNIST) or regression tasks are well-handled by DL based approaches. A common explanation for such gain in performances is the capacity of DNN to extract meaningful features tailored for the downstream task as soon as the amount of available data is large enough. Successes on traditional machine learning tasks pave the way to the application of DL to more challenging and complex problems. From this standpoint emerging topics such as video prediction, image inpainting, natural language processing etc. have seen significant progresses with the introduction of DNN architectures.

Unlike recent machine learning tasks, the modeling of physical processes is nearly as old as statistics. Indeed, from animal and population census to astronomical ephemeris, statistical knowledge is gathered to confront a model to empirical evidences. Over the last few years, the number of sensors and satellite missions measuring and monitoring Earth climate drastically increased. This increase in data accessibility provides a great opportunity for the machine learning community to test the applicability of DL methods to atmospheric and climate data.

The collected data serve several goals. First of all, these satellite observations enable a fine monitoring of various quantities of crucial interest, e.g. the Sea Surface Temperature (SST), that control the variations of Earth Climate. Also, on a more recent basis, thanks for example to Sentinel satellite mission, greenhouse gazes such as NOx and CO_2 are now monitored. Besides the measurement of climate variables, these observations evidence complex ocean and atmospheric dynamics. Theses dynamics are an intense subject of research in various fields. For instance, climatology and oceanography build dynamical models aiming to explain at best Earth Climate. These models are crucial to understand the root

causes and the dynamics of climate change. Besides practical fields, theoretical fields are also interested in such dynamics, studying for example the smoothness, the uniqueness or the stability of Navier-Stokes equations.

On the first hand theoreticians are confronted to non-linearity, smoothness and dimensional issues. Practitioners on the other hand often face several difficulties when dealing with real-life data as incomplete observations, exogenous and unobserved terms (e.g. forcing terms that account for changes in the radiative heat balance) or complex and unknown interactions (e.g. ocean-atmosphere, surface-deep ocean layers). More generally, ML is a promising alternative for scientific problems whose underlying processes are not fully understood, or when the computational cost to run physical simulation is prohibitive. Then, DL methods, able to extract complex features from data, provide prior-free methods to learn the observed phenomenon. Indeed, a strength of such prior-free and data-driven methods is that they do not rely on any physical principle. The physical principle is learned by the algorithm and depends on the considered task.

Besides climate and Earth sciences, several other natural sciences have seen the introduction of ML and DL methods to help the modeling. For example Chemistry is now assisted by ML methods for drug and molecular discovery. ML has also helped material science in the prediction of macroscopic properties from microscopic characteristics. Finally, ML algorithms have been proven helpful in medicine since cancer detection in medical images defines a classical computer vision task, for which DL excels.

However, DL approaches have mainly shown remarkable results either on problems of limited complexity or on complex problems at the cost of gathering a large amount of data. Modeling Physical processes raises new problems and challenges for the ML community.

In particular, the modeling of spatio-temporal dynamics is challenging. Indeed, spatio-temporal modeling involves the learning of the interactions between high dimensional variables at various spatial and temporal ranges. These challenges pave the way to a new area of research: the statistical learning of spatio-temporal phenomena.

CHAPTER 2

OBJECTIVE AND CONTRIBUTIONS

The objective of this thesis is to establish bridges between physical modeling of spatio-temporal phenomena and machine learning. After introducing elementary notations, we present the major challenges encountered when modeling dynamical systems with DL. Then, we introduce our research works and link them to the evidenced learning difficulties.

2.1 Elementary Notations for Dynamical Systems

To set the framework of this thesis, we begin by providing notations and describing the addressed learning tasks. Studying spatio-temporal systems, we focus on modeling the system's dynamics.

Such a modeling in practice amounts to a forecasting task that can be formulated as follows: from past observations of the studied phenomenon, we want to be able to predict future ones. In general, we denote the observed state X, where at each time t, $X_t \in \mathbb{R}^d$. Similar to time series models, we want to predict X at $t+1$ from the observation of X at $t-k$, ..., t. With f_ψ a parametric model, with the parameters ψ to be optimized, we write $\widehat{X}_{t+1}$ the estimated value of X at $t+1$ as:

$$\widehat{X}_{t+1} = f_\psi(X_{t-k}, ..., X_t) \tag{2.1}$$

A particular case, yet broadly used in various natural sciences, is when the dynamics of the system state X follows a differential equation, so that we can write:

$$\frac{dX(t)}{dt} = f(X, t) \tag{2.2}$$

Equation (2.2) describes the (continuous) variations of the state X with the time. It is a very general formulation that can account for both Ordinary Differential Equation (ODE) and Partial Differential Equation (PDE) (and in that case f is a differential operator). Therefore such a dynamical model encompasses a wide range of possible phenomena spanning population dynamics, Newtonian mechanics, fluid dynamics ... Then, the prediction task amounts to approximating f with

a learned function, or more precisely at learning how to accurately integrate a given initial condition. This task is also known as a *forward problem*.

When dealing with dynamical systems, several other tasks may emerge besides forecasting, e.g. the estimation of physical quantities from data. Such a task is often referred to as learning an *inverse* problem. For instance, we consider in this work the estimation of the parameters of ODE or PDE from data. The latter task is related to system identification, which aims at estimating the parameters governing the dynamics of an observed system. This research topic, originating from the dynamical system research field, has now emerged in the scientific ML community. Finally, another prototypical inverse modeling task emerge from the study of dynamical systems: complete state estimation. The task is to recover the complete state of a system from limited measurements. Let $Z_t = (X_t, Y_t)$ be our system of interest and let Z_t be partially observed, i.e. suppose that X_t is observed while Y_t is not. The objective is the estimation of the unobserved Y given the measurements of the observed X_t. This task is at stake in oceanography where two fields are intricated: an observed scalar tracer field X_t (that accounts for instance for the ocean temperature, or more generally that describes the concentration of a physical quantity), and a vector velocity field Y_t describing the transport of X_t in time. Because, it is in practice much easier to observe the tracer quantity X_t, inverse models are calibrated to obtain velocity fields Y_t given the observations of X_t.

Besides forward and inverse modeling, natural sciences raise several challenges for the ML community. Of crucial importance for physicists is *uncertainty quantification*, that instead of learning a function f mapping inputs x to output y, aims at characterizing the distribution $p(y|x)$, enabling a finer analysis than just a pointwise regression. Also, the *automatic discovery of governing equations* is a longstanding topic of statistics that aims at estimating the differential operator acting on a system. Finally, *data generation* that aims at sampling from a distribution, finds application for example in chemistry or in drug design, where given a chemical property x, the goal is to find potential molecule y satisfying such a property, i.e. one want to be able to sample in $p(y|x)$.

2.2 The Difficulties of Machine-Learning for Dynamical Systems

Despite theoretical results showing the approximation capacities of Neural Network (NN), the training of a NN on dynamical systems data can be unstable and difficult, leading to inaccurate estimation and prediction. In sections 2.2.1

to 2.2.3 we describe the major challenges in machine learning for dynamical systems.

2.2.1 Generalization

Consider for example the learning of a forward model, approximating the dynamics of the data. In this setting, several difficulties arise. The major one is the conservation of physical properties. For example, one can think of the preservation of the total energy in mechanics. Other crucial issues include the stability of the model for long term prediction and robustness to out-of-distribution data. Indeed, if the learned model is inaccurate for long-term prediction, it could mean that the learned statistical model misses a critical physical law underlying the observed phenomenon. Likewise, the non-robustness to out-of-distribution data means that the learned predictor is unable to generalize; thus, the model is only able to interpolate between the observed data points. When aiming to learn a model, the latter point is crucial to assess whether the physics of the studied system is actually learned. To sum up, when learning the dynamics of a system the two following difficulties may arise:

1. generalization for long-term prediction.

2. generalization on novel initial conditions or out-of-distribution data.

2.2.2 Incorporating Physical Prior

Another crucial area of research in the machine learning community addresses the inclusion of prior knowledge in the design of NN architectures. This task has motivated several topics of machine learning research. For instance Convolutional Neural Network (ConvNet) and their equivariance to translation improved significantly computer vision results on a wide variety of tasks such as classification. Recently, equivariant-NN are an active research topic with applications for example in molecular synthesis. Besides these general algebraic approaches, the inclusion of prior (physical) knowledge over a dynamical system is still largely unexplored. This topic is crucial since most purely NN approaches fail to provide an acceptable physical prediction. Hence, the inclusion of prior physical knowledge and the physical plausibility of an output are crucial for the learning of dynamical systems.

2.2.3 Interpretability

Finally, by construction, a NN extracts complex relationships between the input variables, and make the resulting NN difficultly interpretable. This uninterpretability characterizes a model for which the input to output mapping is not understandable. In other words, one cannot explicit the changes in the inputs causing a change in the outputs. Then, NN are often referred to as black-box models. This prevents the use of NN in several industrial applications where the interpretability is a prerequisite.

Our goal in this thesis, is to propose methods addressing these highlighted issues. The remaining of this chapter is dedicated to a brief review of our propositions. In section 2.3, we present an instance of a weak-prior on the structure of a NN prediction system to increase interpretability and forecast performances. Section 2.4 introduces the incorporation of prior physical knowledge in a NN with a particular focus on differential system and show how it can help generalization. Finally, in section 2.5, we describe a new method for learning from multiple environments, adapted to dynamical system data, to improve generalization.

2.3 Disentanglement for Spatio-Temporal Systems

We ask ourselves whether strong inductive biases, inspired by differential systems, can guide the design of DL algorithms. Several DL-based models for spatio-temporal systems exist, for example deploying Long Short Term Memory (LSTM) or more complex RNN schemes. However, such models are not interpretable due to the models being black-box and are prone to over-fitting, struggling to extrapolate in time. Therefore, we follow a recent trend in machine learning that aims at separating factors of variations in the prediction. This property is referred to as *disentanglement*. We propose to perform disentanglement to model a spatio-temporal phenomenon with a focus on a prediction task. We show in chapter 6 that the proposed algorithm increases interpretability and provides accurate long-term prediction, hence addressing two of the issues raised above.

2.3.1 Disentanglement in ML

Let D be a NN that reconstructs a signal X from a latent space $\mathcal{Z}$, i.e. for $z \in \mathcal{Z} : X = D(z)$. Disentanglement is a property of the tuple $(\mathcal{Z}, D)$, so that when one coordinate z_i in z varies, only one observable attribute in the decoded signal varies. In other words, consider $\tilde{z} \in \mathcal{Z}$, so that $\forall i \neq i_0 \, \tilde{z}_i = z_i$ and $\tilde{z}_{i_0} \neq$

z_{i_0}, then the decoded signals $X = D(z)$, and $\tilde{X} = D(\tilde{z})$, vary by exactly one characteristic (associated to the coordinate i_0). Such a behavior is often obtained by enforcing a factorized probabilistic prior in the latent space $\mathcal{Z}$. Note that a factorized probabilistic prior over the latent space $\mathcal{Z}$, for instance via variational auto-encoding, induces that the coordinates of $z \in \mathcal{Z}$ are independent and it is often used in practice.

In these models, interpretability is increased thanks to the fact that the practitioner has a finer understanding on how each variable of the input z impacts the prediction $D(z)$.

2.3.2 Disentanglement in Spatio-Temporal Data

Disentanglement finds extension for time-varying phenomena: spatio-temporal disentanglement that also aims at separating factors of variation in a latent space (associated with a decoder network D). We focus on the dissociation of the dynamics and visual aspects of the data. This definition encompasses various proposition such as foreground-background decomposition or the building of structured frame representations. More precisely, we focus on constructing an algorithm providing us with a separation of content and motion information.

Finally, besides interpretability, we make the hypothesis that separating static from dynamical information enables us to increase prediction performance. Indeed, adapted inductive biases have been shown to facilitate learning and generalization.

Learning Task

We address both a representation learning and a structured prediction task as we aim not only at providing accurate prediction but also separating dynamic from static content in a sequence. The learning task is twofold as we want to learn:

1. a dynamic and a static representation from a sequence of observations.
2. a forward model of the observed system from the learned representations.

More formally, let X be our observed system of interest, we aim at extracting a static (time invariant) representation S and a dynamic representation T. For $X_{t-\tau:t} = (X_{t-\tau}, ..., X_t)$, it writes as

$$S,\ T_t = E_S(X_{t-\tau_S:t}),\ E_T(X_{t-\tau_T:t}), \tag{2.3}$$

where E_S and E_T are encoder networks. Then, from the extracted representations S and T, we aim at learning a decoder D, that forwards X in time as:

$$\widehat{X}_{t+1} = D(S, T_t) \tag{2.4}$$

This structured learning presents several advantages. First, learning directly the dynamics of a spatio-temporal phenomenon in the pixel space may be hard because of long-range interactions and high dimension. Also, the above formulation decomposes the initial prediction task, into simpler sub-problems. Finally, it provides the practitioners with a finer understanding of the model by separating motion from static information.

2.3.3 Contribution

In our proposition, we aim at providing an effective framework, yielding adapted inductive biases for spatio-temporal disentanglement. The proposed method, inspired by the separation of variable, a resolution method of PDE, was published as a conference paper:

Jérémie Donà, Jean-Yves Franceschi, Sylvain Lamprier, and Patrick Gallinari (May 2021). "PDE-Driven Spatiotemporal Disentanglement". In: *The Ninth International Conference on Learning Representations*. International Conference on Representation Learning. Vienne, Austria. URL: https://hal.archives-ouvertes.fr/hal-02911067.

2.4 Hybrid Modeling, Bridging Physical and Statistical Models

Hybrid modeling aims at completing prior physical knowledge with a statistical component. In particular, we focus on spatio-temporal dynamics defined by differential equations. This task receives an increasing focus in the machine learning community. Indeed, as previously evoked, several natural processes such as in oceanography or other climate science, involve unknown interactions that need to be taken into account to build an accurate dynamical model. More specifically, we aim at enhancing a physical model with a statistical component. In that perspective, we are looking for a well-posed formulation that enables to bridge a physical hypothesis with a data-driven component. Two main advantages result from using a hybrid model. First of all, hybrid models are more easily interpretable compared to black-box models. Also, because physics based models rely on first

principles (conservation of energy, mass ...) their prediction performances do not suffer from out-of-distribution samples; Thus, we expect hybrid models to generalize better than purely data-driven method. This motivates the need of bridging physics based models and data driven algorithms. The learning of hybrid models then defines a twofold task. First of all, we want to be able to predict accurately the observed phenomenon. Moreover, because the physical hypothesis may depend on unknown parameters, we want to estimate them precisely. Therefore, this problem amounts to the resolution of a *forward* and an *inverse* problem.

Before detailing our contribution, we begin by providing further elements on the addressed tasks.

2.4.1 A Short Introduction to Hybrid Models

Hybrid models use physical assumptions about the dynamics f, as defined in eq. (2.2), in the design of the learning algorithm. Incorporating prior knowledge or physical assumption in the learning of eq. (2.2) for hybrid modeling can take various forms. A first one makes use of a closed-form solution to eq. (2.2) as in (Bézenac et al. 2018b). This closed-form solution depends on parameters. Thus, the objective of hybrid modeling is to find an adapted way to learn the parameters that fit the observed dynamics. Another assumption on f is to assume an additive decomposition of f as $f = f_k + f_u$, where one of the component is known, (Rico-Martinez et al. 1994).

In particular, we tackle in this work the latter : the learning of *partially known dynamics*. Several natural phenomena are well-described by physicist using domain specific knowledge. However, for complex dynamics, some part of the dynamics might not be known to the practitioner. This may be due to various factors, such as idealized assumptions, computational constraints prescribing a fine grain modeling unknown external factors, forces and source etc... NN are then interesting candidates to complete the partial knowledge over the system dynamics. Therefore, our goal is to augment the domain specific models with ML. We rely in this part on the hypothesis of an additive decomposition for the overall dynamics.

That is, we denote f_k the prior (domain specific) knowledge the practitioner has about the dynamics f, and f_u the complementary dynamics to model f (that can be unknown). We can re-write eq. (2.2) using the additive decomposition hypothesis as:

$$\frac{dX_t}{dt} = f(X_t, t) = f_k(X_t, t) + f_u(X_t, t), \tag{2.5}$$

2.4.2 Learning Problem

Prediction:

The resolution of the forward problem amounts to integrate any state X at time t_0 up to t_1, using an approximation of f. Given the domain specific knowledge f_k, we aim at learning h_u approximating f_u so that:

$$X_{t_1} = X_{t_0} + \int_{t_0}^{t_1} f(X(t), t)dt \approx X_{t_0} + \int_{t_0}^{t_1} (f_k + h_u)(X, t)dt$$

Because, we aim at solving the forward problem, i.e. approximating f, few practical difficulties may occur when dealing with the above integration. The first concerns the complexity and the relative importance of f_u. For instance f_u can be non-smooth hence harder to learn. Second, depending on the prediction horizon, the prediction error may become large leading to exploding gradients making the optimization difficult.

Parameter Identification:

More critically, f_k may depend on parameters θ that are unknown. That is, we know the form of f_k but ignore the values of its parameters. We write the true $f_k(X, t)$ as $f_k(X, t, \theta^\star)$, and denote $\mathcal{H}_k$ the space of functions associated to f_k, with $\mathcal{H}_k = \{h_k : \mathbb{R}^p \mapsto \mathbb{R}^p | \exists\ \theta,\ \text{s.t}\ h_k(X, t) = f_k(X, t, \theta)\ \}$.

In this case, our goal is to learn (θ, h_u) so that:

$$X_{t_1} = X_{t_0} + \int_{t_0}^{t_1} f(X, t)dt \approx X_{t_0} + \int_{t_0}^{t_1} (h_k(X, t, \theta) + h_u(X, t))dt, \tag{2.6}$$

With h_u a free-form model, e.g. a NN. In this setting, we are facing an ill-posedness issue: it may exist an infinite number of decompositions (θ, h_u) providing a good approximation to eq. (2.6). Therefore, we concentrate on solving the ill-posedness to recover both accurate trajectories of the state X and a physically sound estimation of the parameters θ attached to the physical hypothesis. The estimation of θ is referred to as an inverse problem. Note that, the geophysical community developed *data-assimilation* methods, see (Moore 1991), using for instance a prediction model to solve an inverse problem, thus intricating the estimation of physical parameters and the ability to predict the observed state of the system.

2.4.3 Contributions

We address the simultaneous resolution of both the forward problem (to model accurately the observed dynamics) and the inverse problem (to estimate parameters of the associated ODE or PDE) with adapted learning constraints. This axis of research has lead to two publications studying the augmentation of physical models with deep learning:

1. Yuan Yin, Vincent Le Guen, Jérémie Donà, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari (2021b). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *The Ninth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=kmG8vRXTFv.

2. Jérémie Donà, Marie Déchelle, Patrick Gallinari, and Marina Lévy (2022). "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction". In: *The Tenth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=gbe1zHyA73

2.5 Learning Neural Dynamics from Several Environments

Another crucial application of machine learning arises when the data to learn from originate from several environments. This learning setting has gained momentum for classification tasks, with frameworks such as Meta-Learning or Multi-Task Learning.

This setting is also encountered when learning dynamical systems. Indeed, it is common to observe several trajectories of the same phenomenon, with variations in the underlying dynamics. These changes in the dynamics can be caused by a change in the values of the parameters of the underlying ODE, or a change in the initial or border condition leading sometimes to radically different trajectories. This setting then defines an essential task in the learning of dynamical systems tightly linked with generalization.

2.5.1 Task And Contribution

Suppose we observe a set of trajectories $\mathcal{D}^e$, where $e \in \mathcal{E}$ characterizes the context, i.e. we have for some example i of environment e, $\mathcal{D}_i^e = \{X_{t_0}^e, X_{t_1}^e, ..., X_{t_n}^e\}$ that verifies:

$$\frac{dX^e}{dt} = f^e(X^e) \tag{2.7}$$

so that $\forall e \in \mathcal{E}$, $f^e \in \mathcal{F}$, and $\mathcal{F}$ is a function space. Since the trajectories depict the same phenomenon, a particular example of learning from multiple environments occurs when f^{e_i} and f^{e_j} share the same parametric form but with different parameters.

Let g_θ be a neural network, with $\theta \in \Theta \subseteq \mathbb{R}^n$. A common approach to solve the prediction associated to eq. (2.7) is:

$$\min_\theta \sum_{e \in \mathcal{E}} \mathcal{L}(g_\theta(x_t^e), f^e(X_t^e)), \tag{2.8}$$

This setting can be thought of as finding one function g_θ suitable for all trajectories, hence often referred to as a *One-for-all* learning setting. This approach is prone to under-fitting and often misses the specificities of each trajectories. Another approach consists in fitting one NN g_{θ^e} for each environment solving:

$$\min_{\theta_e} \mathcal{L}(g_{\theta^e}(X_t^e), f^e(X_t^e)) \quad \forall e \in \mathcal{E}, \tag{2.9}$$

Since it learns one function per environment this approach does not leverage the common behavior of all environments, is prone to over-fitting and is computationally expensive.

Contribution We address the drawbacks of the above approaches and develop a novel algorithm to learn from several environments. To do so, we leverage the fact the trajectories depict a common dynamics, and assume the existence parameters $\theta_c \in \Theta$ from which a deviation of limited complexity in the parameter space enables us to find for all environment e a parameter θ^e adapted to the trajectories in $\mathcal{D}^e$ using an hyper-network approach. This work is currently under review:

Matthieu Kirchmeyer, Yuan Yin, Jérémie Donà, Nicolas Baskiotis, Alain Rakotomamonjy, and Patrick Gallinari (Jan. 2022). "Generalizing to New Physical Systems via Context-Informed Dynamics Model". working paper or preprint. URL: https://hal.archives-ouvertes.fr/hal-03547546

Part II

RELATED WORKS

CHAPTER 3

MODELING DYNAMICAL SYSTEMS

Our work lies between several areas of research. Instead of proposing an exhaustive review of all evoked research areas, we give a contextual overview enabling us to provide the reader with the necessary elements to understand and grasp the essential notions, the background and the novelty of our proposed approaches.

This section is organized as follows. In section 3.1 and section 3.2, we begin by reviewing classical approaches to model times series from a statistical standpoint. In section 3.3, we show how machine learning, and in particular DL developed tools to capture and model the temporal dependency in the data. Because our work heavily relies on ODE and PDE, we present their treatment through a statistical approach in a dedicated part, chapter 4.

3.1 Time Series Modeling

Despite being a longstanding tool for natural sciences, the statistical analysis of time series is relatively young. Indeed, econometrics tools investigating the estimation of predictive time series models date from the 1970s.

Consider an observed variable $(X_t)_{t\in[|1:n|]}$. Unless said otherwise, we consider X_t to be scalar for the rest of the subsection. We start by considering econometric modeling of time series. Such models treat the observed time series as a stochastic process, i.e. as a (ordered) sequence of observations in a probabilistic space and rely on two main assumptions: stationarity and ergodicity.

Definition 3.1 (Stationarity)**.** Let $(X_t)_t$ a stochastic process. $(X_t)_t$ is said to be stationary in a weak sense if:

1. It has constant mean, i.e. $\forall s, \mathbb{E}(X_t) = \mathbb{E}(X_{t+s})$
2. Its auto-covariance function defined by $K(t,s) = \mathbb{E}[(X_t - \mathbb{E}(X_t))(X_s - \mathbb{E}(X_s))]$, depends only on the difference between the time steps t and s, i.e. : $K(t,s) = K(t-s, 0)$
3. The second order moments are finite for all times, i.e. : $\mathbb{E}(|X_t|^2) < \infty$.

The above definition relaxes strong-stationarity that states that for all $s, t_1, ..., t_n$, the joint distribution of $(X_{t_1}, ..., X_{t_n})$ and $(X_{t_1+s}, ..., X_{t_n+s})$ are similar. Stationarity implies that the main statistical properties of the sequence $(X_{t_1}, ..., X_{t_n})$ are conserved by considering a lag in the series.

Unlike stationarity characterizing statistical properties of the time series, ergodicity establishes a link with the actually observed dynamical systems.

Definition 3.2 (Ergodicity)**.** Let $(X_t)_t$ be a stochastic process. $(X_t)_t$ is ergodic (in the first order), if its mean converges towards its expected value, i.e. if $\lim_{T\to+\infty} \frac{1}{T}\int_0^T X(t)dt = \mathbb{E}(X_t)$

This definition finally says that the mean of the random process corresponds to the mean of the observations, provided that we have enough observations. The stationarity of the process is a key property of a time series, that enables to make prediction, ergodicity is a technical condition not often discussed in ML. Statistical modeling literature flourishes with time series models. In the following, we briefly review the most classical ones.

3.1.1 Autoregressive Models (AR)

Econometrics proposes to investigate the case where X_t depends linearly on its previous value, modulo a random perturbation of zero average and time independent variance. This kind of model is referred to as *auto-regressive*. Assuming 0-mean for the observed process X_t, an auto-regressive model of order $p \in \mathbb{N}$, denoted AR(p) writes as:

$$X_t = \sum_{i=1}^{p} a_i X_{t-i} + \epsilon_t \tag{3.1}$$

where ϵ_t is a white noise.

We can write eq. (3.1) equivalently as $(1 - \sum_{i=1}^{p} a_i L)X_t = \epsilon_t$ where L is a lag operator. Defining the characteristic polynomial by $\chi(x) = (1 - \sum_{i=1}^{p} a_i x^i)$. The process modeled by eq. (3.1) is stationary if and only if the roots of $\chi(x)$ are strictly outside the unit circle.

The estimation of the coefficients $(a_i)_{i\in[|1,p|]}$ can be conducted for example using ordinary least squares or maximum likelihood.

3.1.2 Moving Average Models (MA)

Unlike AR models, moving average models of orders q denoted MA(q), involve exogenous i.i.d. shocks denoted ϵ, impacting durably the time series as:

$$X_t = \epsilon_t + \sum_{i=1}^{p} a_i \epsilon_{t-i} \tag{3.2}$$

Notably X_t is stationary as the sum of stationary processes. Of interest for modeling and estimation, Moving Average models are said invertible when they can be re-written as an $AR(\infty)$ model. In general the estimation of the parameters of eq. (3.2) is conducted via maximum likelihood.

Extensions Moving average models are interesting since according to Wold theorem (Fuller 1976), any covariance stationary process X_t, can be represented using a MA(∞) (whose coefficients are square summable), plus a deterministic term. The estimation of the deterministic term has led to a refinement denoted ARMA(p,q) that combines an AR(p) and a MA(q) models. To cope with potential non-stationarity, ARIMA(p,i,q) models were developed; instead of modeling the original time series X_t models the difference of order i of X_t: $\Delta^i X$. The differentiation process eliminates potential polynomial trends in the time series enabling to recover a stationary time series.

When X_t is a vector of observation, Vector Autoregressive (VAR) models generalize AR models by modeling a vectorial times series instead of modeling a scalar times series. They enable to incorporate the influence of other variables in the modeling process.

The above methods are variants of the linear regression model, hence assuming constant variance of the error. This assumption has been criticized: inspired by financial time series, (G)ARCH models (Engle 1982) deals with the time dependent intrinsic volatility (i.e. standard deviation) of the time series by modeling the heteroscedasticity. With m a regression function of parameters β, a classical model is:

$$\begin{aligned} &Y_t = m(X_t; \beta) + \epsilon_t \\ &\mathbb{E}[\epsilon_t | \epsilon_{t-1}] = 0 \text{ and } \mathbb{V}(\epsilon_t | \epsilon_{t-1}) = \sigma_t^2 \\ &\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i} \end{aligned}$$

3.2 State-Space Models and Kalman Filter

State-space models are crucial examples of how probabilistic modeling can help modeling observed dynamical systems. By definition, states-space models separate the dynamical model from the observation model. The dynamical model operates on an unobserved state X, whereas an observation model links the (unobserved) state X to the actual measurements z. In Bayesian statistics given observations $(z_1, .., z_t)$ common tasks are:

1. smoothing: for $n < t$ estimate $p(X_n|z_1, ...z_t, \theta)$
2. filtering: estimate $p(X_t|z_1, ...z_t, \theta)$
3. prediction: estimate $p(X_{t+1}|z_1, ..., z_t)$

In general state-space models have the following form:

$$X_t = f(X_{t-1}) + \epsilon$$
$$z_t = g(X_t) + \eta$$

Where f, g are respectively the dynamical model and the measurement (or emission) function, and ϵ, η are i.i.d. noises.

Perhaps the most famous example of state-space model is the linear and Gaussian case that leads to the Kalman filter. Given a linear dynamical and measurement model, the Kalman filter proposes a solution to the filtering problem, along with a simple method for prediction. Indeed, it introduces two essential equations: a linear stochastic difference equation (i.e. a linear dynamical model) and a measurement equation:

$$X_k = FX_{k-1} + Bu_k + w_{k-1} \tag{3.3}$$
$$z_k = HX_k + v_k \tag{3.4}$$

Where X_k is the signal of interest with dynamics given by eq. (3.3), u_k is a control vector, F is a transition matrix, B and H are respectively a control and a measurement matrix, and both (w_k, v_k) represent process and measurement noise, so that $p(w) \sim \mathcal{N}(0, Q)$, and $p(v) \sim \mathcal{N}(0, R)$ (with Q, R covariance matrices).

The objective of the Kalman filter is to provide estimate for X_k given: the initial state X_0, the information about the system F, B, H, Q, R and observation $z_1, ..., z_k$. Note that in practice, Q and R are not known and are parameters to be tuned, whereas F, B and H are known matrices. The Kalman filter procedure introduces P, accounting for the predicted error covariance. Kalman filtering alternates a two steps procedure:

1. Prediction Step: update the estimated prediction and the covariance of the error:

$$\text{State} \quad \hat{X}_k^- = F\hat{X}_{k-1}^+ + Bu_{k-1} \tag{3.5}$$
$$\text{Predicted Error Covariance} \quad P_k^- = FP_{k-1}^+F^T + Q \tag{3.6}$$

2. Update Step: compute:

$$\text{Residual} \quad \tilde{y}_k = z_k - H\hat{X}_k^- \tag{3.7}$$
$$\text{Kalman Gain} \quad K_k^- = P_k^- H^t(R + HP_k^- H^t)^{-1} \tag{3.8}$$
$$\text{Update state estimate} \quad \hat{X}_k^+ = \hat{X}_k^- + K_k\tilde{y}_k \tag{3.9}$$
$$\text{Predicted Error Covariance} \quad P_k^+ = (I - K_kH)P_k^- \tag{3.10}$$

By design, such a procedure minimizes the estimated error covariance, and is in that sense optimal. Initially designed for inference and smoothing in the context of vehicle motion, a major limitation of the Kalman filter is the knowledge of the transition matrix F. To alleviate this drawback, several techniques exists to learn the transition matrix F, such as the Expectation-Maximization (EM) algorithm, by maximizing the conditional likelihood of the parameters given the observations or iterative least squares.

Kalman filter finds extension in non-linear settings via the formalism of the extended Kalman filter. For more complex observation function and dynamics, e.g. non linear dynamics and measurements, inference smoothing and prediction can be conducted using non linear Monte Carlo methods as for example the particle filter and Sequential Monte Carlo (SMC), see (Doucet et al. 2000).

3.3 Recurrent Neural Networks

The time-series models of section 3.1 model directly the observations. On the other hand, state-space models as described in section 3.2 model the dynamic of an unobserved state. Traditional models and particle filter approaches become intractable when the Markov assumption is relaxed or when the dynamics is unknown. NN and RNN are gifted with universal approximation capacities. Simply put, RNN learns a representation of the data in a latent space to simplify the modeling of the dynamics. As a reminder, a short overview of NN architecture is available in appendix A.

Besides, the accuracy of NN predictions for long horizons depend strongly on the capacity of the model in producing accurate outputs at each time-step, otherwise error may accumulate leading to aberrant or unrealistic predictions.

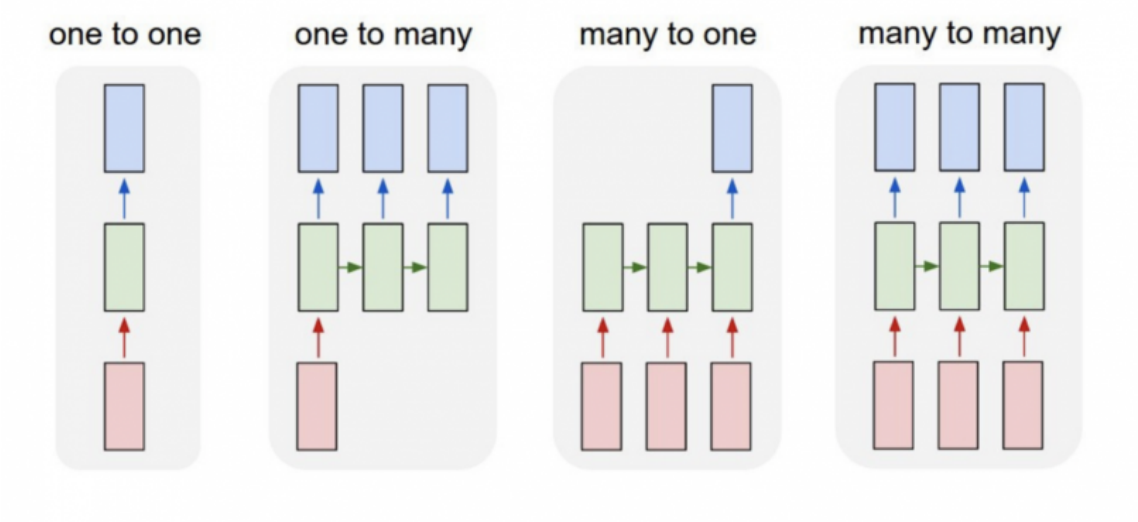


Figure 3.1. – Because the inputs $X_{1:T}$ are treated sequentially, RNN can handle a wide variety of temporal tasks. Schema taken from: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

3.3.1 Introduction and Notation

A way to bypass this limitation is to model the dynamics in a smaller dimensional latent space in which the data are well-represented. More specifically, as soon as sequential data are involved, RNN are used to extract dynamical information from the input sequence. Similarly to classical NN, see appendix A, the main objective objective for RNN is the accurate modeling of an output $y \in \mathbb{R}^d$ given a sequence of output $X_{1:t} = X_1, ..., X_t$, thus to maximize:

$$\log p_\theta(y|X_{1:t}) \tag{3.11}$$

In the case of sequence modeling, i.e. when the task is to predict $y_{1:t}$, where $y_k \in \mathbb{R}^d$ eq. (3.11) rewrites as:

$$\log p_\theta(y_{1:t}|X_{1:t}) = \sum_{k=1}^{t} \log p_\theta(y_k|X_{1:k}) \tag{3.12}$$

Thanks to their ability to handle data sequentially, RNN can handle several tasks as illustrated in fig. 3.1. In that perspective, RNN have found several application fields such as in Natural Language Processing (NLP) for text generation and translation, multimodal learning with for instance image captioning, spatio-temporal prediction, image inpainting by modeling sequentially the likelihood of the pixels (Oord et al. 2016).

For modeling the conditional distribution $p_\theta(y_t|X_1, ..., X_t)$, a RNN learns a vectorial (or tensorial) representation denoted h_t which current value at time t depends on h_{t-1}. The classical architecture of a RNN obeys the following equation: $h_t = f_\theta(X_t, h_{t-1})$ and its conventional implementation denoted vanilla-RNN is:

$$h_t = \tanh(W_x.X_t + W_h.h_{t-1} + b) \tag{3.13}$$
$$\hat{y}_t = g_\psi(h_t) \tag{3.14}$$

Where W_x, W_h are weight matrices and g_ψ is a NN mapping the learned representation h_t to the output space. Note that

However, eq. (3.13) has several drawbacks as its training relies on the BackPropagation Through Time (BPTT) algorithm. Indeed, let $\hat{y}_t$ be the estimated output of the RNN, according to the chain rule the derivative of the loss $\mathcal{L}$ with respect to the matrix W_h writes as:

$$\frac{\partial \mathcal{L}}{\partial \hat{y}_t} \propto \sum_{k=1}^{t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W_h}$$

A term can cause trouble: $\frac{\partial h_t}{\partial h_k}$:

$$\frac{\partial h_t}{\partial h_k} = \prod_{j=k}^{j=t-1} \frac{\partial h_{j+1}}{\partial h_j} = \prod_{j=k}^{j=t-1} (1 - tanh^2(W_x.x_t + W_h.h_{j-1} + b))W_h \tag{3.15}$$

As noted in (Y. Bengio et al. 1993), two main problems emerge from the above derivation. First, the term $1 - tanh^2$ tends to quickly saturate at 0 when the output deviates from 0. The second problem that emerges from eq. (3.15) is the multiplication by W_h $(k - t)$-times. Therefore, if the norm of W_h is high, this term increases geometrically, leading to an instability in the training called *exploding* gradients. Conversely, if the norm of W_h is low, this term decreases geometrically towards 0, leading to *vanishing* gradients problem. In either case, the corresponding gradient will not help modeling the conditional $p_\theta(y_t|x_{1:t})$.

Several solutions have been proposed to learn vanilla- RNN, as for example truncating terms in eq. (3.15) (Williams and Peng 1990) or value clipping as (Pascanu et al. 2013). Besides the stability of the backward pass, another area of research concerns NN-architectures.

3.3.2 Models and Architectures

Several architectures propose to enhance traditional RNN notably for the learning of long term dependencies.

GRU

The idea behind the GRU is to enable a better flow of information through the recurrent cell, that would also prevent gradient related issues. To do so, a GRU-cell manages two gates that enables to deal with the data flow throughout the sequence:

1. *reset* gate denoted r
2. *update* gate denoted z

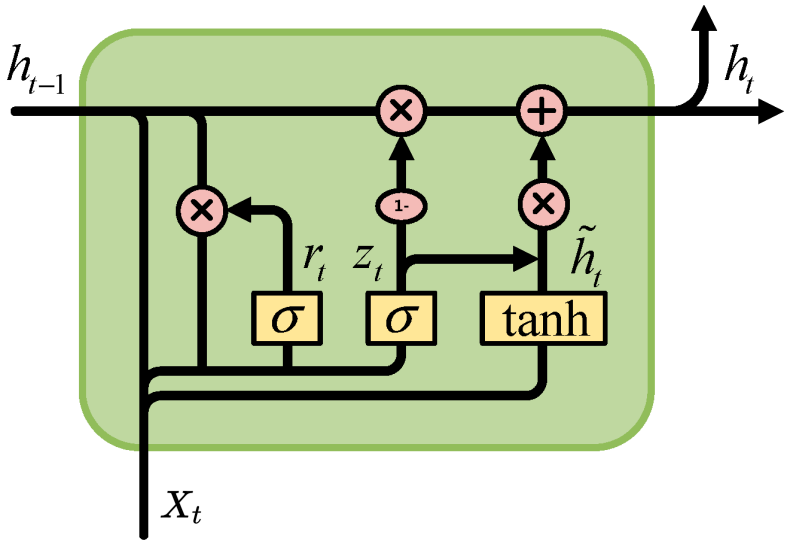


Figure 3.2. – Schematic Overview of the computations in a GRU-cell. Illustration taken from (Pan et al. 2018)

Given an input X_t and a previous state h_{t-1} the *reset* and *update* state follow:

$$r_t = \sigma(W_{rx}X_t + W_{rh}h_{t-1} + b_r) \tag{3.16}$$
$$z_t = \sigma(W_{hx}X_t + W_{zh}h_{t_1} + b_z) \tag{3.17}$$

z_t, r_t and h have similar dimension and $W_{rx}, W_{rh}, W_{hx}, W_{zh}$ are weight matrices of appropriate dimension.

An intermediate state $\tilde{h}_t$ is introduced, following:

$$\tilde{h}_t = tanh(W_{hx}X_t + W_{hh}(r_t \odot h_{t-1}) + b_h) \tag{3.18}$$

$\tilde{h}_t$ combines an innovation from the new incoming data X_t via $W_{hx}X_t$ and the possibility to forget all past information of the previous state h_{t-1} via $r_t \odot h_{t-1}$. Finally this intermediate state is combined with the previous state following:

$$h_t = z_t\tilde{h}_t + (1 - z_t)h_{t-1} \tag{3.19}$$

Equation (3.19) shows that z_t is learned to optimally mix innovation i.e. $\tilde{h}_t$ and the previous state h_{t-1}. These equations can be summarized schematically as in fig. 3.2

GRU, introduced in (Cho et al. 2014), combines the gradient advantages of LSTM, as will be detailed in the next session or see (Rehmer and Kroll 2020), while alleviating slightly the computational cost of LSTM since LSTM requires an additional matrix multiplication compared to GRU.

Long-Short Term Memory Networks

Originally proposed in (Hochreiter and Schmidhuber 1997), LSTM aims at addressing both shortcomings of RNN, i.e. modeling long-term dependency and training instabilities. Similarly to what is done in GRU cells, LSTM introduces several gating mechanisms: an input gate i_t and a forget gate denoted f_t control the information flow in the LSTM cell, along with a cell state denoted c_t. With σ the sigmoid activation, $\odot$ the element wise product, and $[a, b]$ the concatenation of a and b, the LSTM equations are given by eqs. (3.20) to (3.24), and a schematic illustration is provided in fig. 3.3.

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i) \tag{3.20}$$
$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \tag{3.21}$$
$$o_t = \sigma(W_o[h_{t-1}, X_t] + b_o) \tag{3.22}$$
$$\tilde{c}_t = tanh(W_c[h_{t-1}, X_t] + b_c) \tag{3.23}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{3.24}$$
$$h_t = o_t \odot tanh(c_t) \tag{3.25}$$

In that case, we have: $\frac{\partial c_t}{\partial c_{t-1}} = f_t$, therefore since f_t is the output of a sigmoid layer it is between $[0, 1]$, and the gradients neither geometrically increase nor decrease as in the vanilla RNN case. Finally, exhaustive experimental comparison concluded to that GRU and LSTM obtain comparable results on sequence modeling tasks.

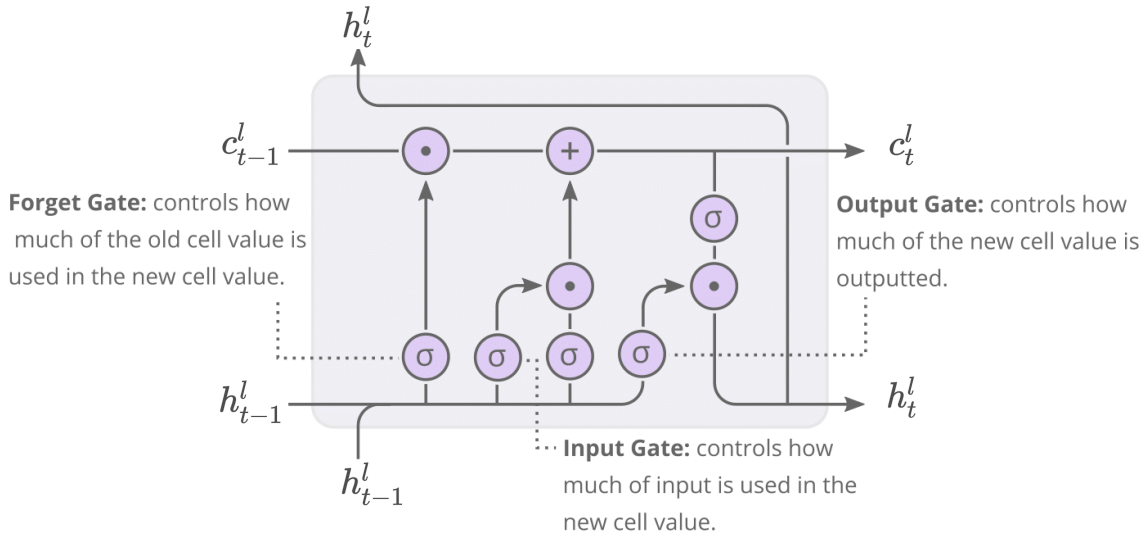


Figure 3.3. – Schematic illustration of (stacked) LSTM cells. The forget gate and input gate enables a better gradient flow preventing gradient issues during learning https://distill.pub/2019/memorization-in-rnns/

RNN Refinements

RNN architectures can be refined: RNN cells can be stacked, the output h_t of the first RNN being the input of the following RNN-cell as illustrated in fig. 3.3, so on and so forth. Also, the sequence can be processed using a positive and a negative time axis, i.e. processing the sequence in two ways (Schuster and Paliwal 1997): one from $X_1, ..., X_t$ the other from $X_t, ..., X_1$. The intuition behind the latter is that processing the sequence both ways will provide a better grasp of the context.

Attention Based Networks

Both LSTM and GRU are now considered traditional deep learning approaches to sequence modeling. Recently, a novel approach: *attention*, brought a significant improvement in both NLP and computer vision tasks. Crucially, it does not imply recurrent computations for sequence modelling, breaking a significant bottleneck of traditional RNN approaches.

Introduced in (Bahdanau et al. 2015), the attention mechanism enables to focus on the desired part of a sequence. In its original formulation, it comprises 3 main parts:

1. A RNN-Encoder that encodes the input sequence $(X_i)_{i\in[|1,t|]}$ as $(h_i)_{i\in[|1,t|]}$.
2. A RNN-Decoder Network: let $y_{i\in[|1,t|]}$ the sequence to be reconstructed, the task is formulated as maximizing: $p(y_k|y_{k-1}, s_k, c)$ parameterized as

$g(y_{k-1}, s_k, c)$, where s_k is the state of a RNN, and c is an context vector. s_i is given by $s_i = f(y_{i-1}, s_{i-1}, c_i)$.

3. The construction of the context vector or attention mechanism focuses on how to compute the c_i to facilitate learning. The core idea is to find which representations in the sequence of h_j mostly influence the output to be emitted at time i.

   The attention proposed in (Bahdanau et al. 2015) computes the c_i as a weighted average of all the hidden representations h_j. The weights, denoted α, express the dependence between the hidden representation h_j and the word to be emitted at time i, represented by s_{i-1}, and is usually parameterized by a Multi-Layer Perceptron (MLP) taking as input the concatenation $[h_j, s_{i-1}]$.

   Finally, we write $c_i = \sum_{j=1}^{j=t} \alpha_{i,j} h_j$. And for all i, $\alpha_{i,.}$ is a probability distribution over the integer $[|1, ..., t|]$, so that c_i is a weighted average of the h_j. Practically, $\alpha_{i,j}$ is the output of a neural network $a(h_j, s_{i-1})$.

Other attention mechanisms exist, see (Luong et al. 2015). For instance, the *self-attention* computes the fitness weights α between the observations of the input sequence $X_{1:t}$. Combined with *Positional Encoding*, a learned representation of the position of the input within the sequence, *self-attention* led to the Transformer architecture, see (Vaswani et al. 2017). Since their introduction, Transformer-based architectures have succeeded in increasing performances on several NLP tasks (Ott et al. 2018; Dai et al. 2019). Besides NLP tasks, Transformer architectures have spread to computer vision for image classification tasks (Dosovitskiy et al. 2021). Recently, the use of transformers for spatio-temporal tasks as video and dynamical system prediction (Weissenborn et al. 2020; Geneva and Zabaras 2021) shows promising results.

3.3.3 RNN for Dynamical Systems

Let aside traditional applications to language models, a significant research track focuses on the application and adaption of RNN to the modeling of dynamical systems. Direct applications of RNN to dynamical systems date back to (Funahashi and Nakamura 1993) in which the authors prove that any finite trajectory dynamical system can be realized by a RNN. Besides the application to model univariate or multivariate time series (Cai et al. 2019; Che et al. 2018; Hewamalage et al. 2021), this theoretical motivation inspired several papers to use directly RNN to learn the dynamics of physical systems (X. Jia et al. 2021; Wan et al. 2018).

A milestone in this field is the introduction of ConvLSTM by (Shi et al. 2015): with the objective of short time forecasting, the authors introduced a novel re-

current cell combining the behavior of LSTM with the spatialized treatment of convolutional neural networks. Notably this architecture designs a latent embedding that does not linearize the data and treat them as images, hence preserves and enhances local structures. (Y. Wang et al. 2019b) proposed a ConvLSTM architecture designed for non stationary spatio-temporal data using *Memory in Memory* blocks that differentiate the time series, just like in an ARIMA model, making it stationary.

Spatio-Temporal prediction is a challenging task due to the variance in the changes that can occur and to the difficulty to generate likely images. Even sophisticated RNN architectures as (Y. Wang et al. 2018) struggle with accurate frame prediction. Yet, RNN-encoders are still used for this task together with generative models (Babaeizadeh et al. 2018; A. X. Lee et al. 2019; Denton and Fergus 2018).

CHAPTER 4

LEARNING ODE AND PDE FROM DATA

The task of learning the dynamics of a system from data is a longstanding problem in the machine learning community and is one of the reasons for the development of RNN. Note that other research fields, such as system identification and data assimilation have also addressed the task of learning ODE parameters from data. As a starting point, consider an ODE describing the evolution of a quantity $X \in \mathbb{R}^d$ through time t

$$\frac{dX_t}{dt} = f(X, t) \tag{4.1}$$

Several problems in machine learning are reformulated as learning a function f that satisfies eq. (4.1). Learning eq. (4.1) amounts to approximating f through a parametric class of function, $\mathcal{F}(\theta)$, large enough to represent a wide variety of functions f.

This chapter is organized as follows, in section 4.1 we provide brief and intuitive elements for the understanding of ODE and PDE based models. Afterwards, we provide details over 3 main learning problems:

1. Prior-Free Forecasting: this approach leverages trajectories of observations $X_{t_0}, ..., X_{t_n}$ for learning a parametric function $f : X_{t-\tau:t} \mapsto X_{t+1}$. Assuming a dynamics following eq. (4.1), a simple discretization writes as:

$$X_{t+1} = X_t + dt f_\theta(X_t, t), \tag{4.2}$$

   with dt the integration step-size. section 4.2 is dedicated to explain Residual Network (ResNet) (K. He et al. 2016) defined by eq. (4.2) along with its continuous version introduced in (R. T. Q. Chen et al. 2018).

2. ODE-PDE resolution, see section 4.3. Unlike the previous approach that relies on data trajectories, this learning task, in its simplest form, assumes that f and the initial condition X_0 are known and propose to learn the solution to eq. (4.1).

3. Hybrid and grey-box modeling, see section 4.4. As detailed in section 2.2.2, one core machine learning issue lies in the incorporation of prior knowledge

in learning algorithms. A prototypical example, illustrated in section 2.4, is the learning of a residual dynamics:

$$\frac{dX_t}{dt} = f(X_t, t) = \underbrace{f_k(X_t, t)}_{prior\ knowledge} + \underbrace{f_u(X_t, t)}_{neuralnet} \tag{4.3}$$

4.1 Brief Introduction to ODE and PDE models

We present introductory elements about ordinary differential equations in section 4.1.1 and partial differential equations in section 4.1.2.

4.1.1 Ordinary Differential Equations

The existence and uniqueness of the solution to eq. (4.1) given an initial condition X_0 and the Lipschitz behavior of f is ensured by the Cauchy-Lipschitz lemma (Cartan and Kouneiher 1967). For less restrictive assumptions, the uniqueness is not guaranteed, as illustrated by the Cauchy-Peano-Arzelà theorem (Cartan and Kouneiher 1967).

Given f, expressing the solution X to eq. (4.1) can be cumbersome, or even infeasible, making the differential formulation of eq. (4.1) impractical. In general, given eq. (4.1) and an initial condition X_0, one major task for dynamical systems practitioners and physicists is the forecast of the quantity X. Thus, developing integration methods for ODE is a longstanding problem in numerical analysis. Their quality is expressed through three main characteristics: Convergence, Consistency, and Stability, see (Arnold 2015). Informally, *convergence* refers to the fact that the numerical solution converges towards the true solution when the stepsize (or integration step) goes to 0. *Consistency* refers to the adequacy of the true solution to the discretized one. Finally, *stability* refers to the bounded behavior of the numerical scheme with respect to the initial condition. As an example, consider the Euler method that discretized eq. (4.1) writing $\frac{dX}{dt}$ as $\frac{X_{t+h}-Xt}{h}$. Its local-truncation error, i.e. its one step ahead prediction error is proportional to (h^2). On the other side, the Grönwall lemma provides us with an upper bound for the global-truncation error (GTE) (i.e. prediction from t_0 up to t): $C(L, ||X''||) \times h(\exp^{L(t-t_0)} - 1)$, where C is a constant that depends on L, the Lipschitz constant of f; and on the norm of the second order derivative of X. For an exhaustive discussion on the subject, we refer the reader to (Butcher and Goodwin 2008).

4.1.2 Partial Differential Equation

While ODE describes the evolution of scalar and vectorial quantities with time, potentially in a very complex manner, PDE introduces a dependency of the target solution X with respect to a vectorial variable or coordinate $x \in \mathbb{R}^l$ so that X writes as $X(x)$. However, in practice, time is separated from spatial coordinates so one writes $X(t, x)$. If X is a scalar field, i.e. $X : \mathbb{R}^+ \times \mathbb{R}^l \mapsto \mathbb{R}$, PDE refers to the fact that the values of X varies both with time t and space x. Therefore the variations of X involve a differential operator with respect to x. If x is a one-dimensional space coordinate, a partial differential equation can often write as:

$$\frac{\partial X(x,t)}{\partial t} = \mathcal{N}(X, \frac{\partial X}{\partial x}, \frac{\partial^2 X}{\partial x^2}, ...), \tag{4.4}$$

where $\mathcal{N}$ is a generic function. General conditions to retrieve existence and uniqueness to eq. (4.4) do not exist as such. Indeed, each PDE problem has its own specificities requiring an ad-hoc treatment. However, the Lax-Milgram theorem, a general tool from linear algebra extending Riez representation lemma, enables to recover the well-posedness of eq. (4.4) for a wide class of PDE, provided adapted initial and boundary conditions. For exhaustive considerations on this matter, see (Renardy and Rogers 2006). Regarding numerical simulations of PDE, several solutions exist depending on the considered equation: Finite Difference Method (FDM), Finite Element Methods (FEM), Finite Volume Methods (FVM). While the FDM discretizes the PDE both in space and time then solve the associated equation on a grid; finite elements and finite volumes methods rather divide the space on which the solution is computed using "elements" (or cells). Then, the target solution is either approximated using a basis of functions in the case of FEM, or rely on conservation of a quantity in the case of FVM see (Ames 1977).

Besides prediction of the system states, several learning tasks emerge when relying on eqs. (4.1) and (4.4). For example, in parametric differential equation solving, the parametric form of f is known, but its coefficient are to be estimated. The system identification literature tackled this subject providing methods to estimate the parameters θ so that $\frac{dX}{dt} = f_\theta(X, t)$, see (Ljung 1998). Another classical example, common in natural sciences such as oceanography, is the identification of some unobserved component in X, as in (Bérériat and Herlin 2015). This task can be solved using data assimilation methods (Carrassi et al. 2018).

4.2 Statistical Learning of ODE and PDE

In this section we describe the adjoint method in section 4.2.2 and details on ResNet and other NN architectures inspired by ODE integration schemes in sections 4.2.1 and 4.2.3

4.2.1 Residual Networks

In its simplest form, residual networks (K. He et al. 2016) writes as:

$$X_{t+1} = f_\theta(X_t) + X_t, \tag{4.5}$$

where θ are learnable parameters.

Its original implementation presents two convolutional weights layers and a ReLU activation as illustrated in fig. 4.1. Also, since the introduction of ResNet

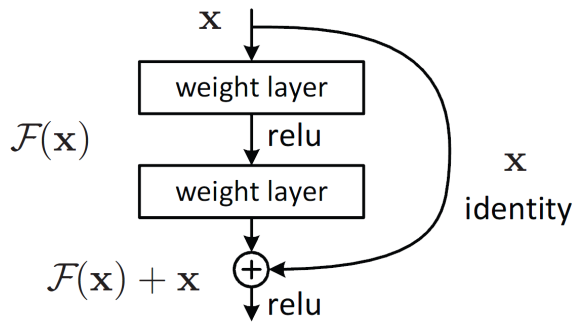


Figure 4.1. – Schematic illustration of a Residual layer. Illustration taken from (K. He et al. 2016)

by K. He et al. (2016) and the remarkable results for classical computer vision tasks such as classification and object detection, the use of the dynamical system formalism has thrived in the DL community. Note that, it is not the analogy with numerical scheme that initially motivated the use of ResNet for ML tasks but rather its stability during training, solving vanishing gradient problems and enabling very deep architectures (Zaeemzadeh et al. 2020).

4.2.2 Neural-ODE and the Continuous Adjoint Method

Following the path opened by ResNet, (R. T. Q. Chen et al. 2018) proposed Neural-ODE, a continuous version of ResNet that addresses the following learning problem:

$$\begin{aligned} &\min_{z} J(z(T), y) \\ &\text{s.t} : dz/dt = f(z(t), t, \theta)\ z(0) = x \end{aligned} \tag{4.6}$$

where $J : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ is some cost function, and y is typically a data point. For instance, in a dynamical system perspective, y corresponds to the observed output of the system state at time T.

Solving eq. (4.6) actually mobilizes a method called the *Adjoint Method*. The adjoint is a theoretical method for computing gradients. It finds two instantiations: a continuous mode that computes the gradient of $\partial f/\partial\theta$ analytically, and a discrete mode that computes the gradient based on explicit discretization, similarly to the backpropagation algorithm. The core idea behind the adjoint method is to avoid cumbersome and expensive estimation of differential terms.

The adjoint method recovers the gradient of J with respect to θ solving eq. (4.6) and writing the Lagrangian $\mathcal{L}$, with λ the Lagrange multiplier function:

$$\mathcal{L}(z(T), \theta, \lambda) = J(z(T), y) + \int_0^T \lambda(t)[\frac{dz}{dt} - f(z(t), t, \theta)]dt \tag{4.7}$$

We now take the derivative of $\mathcal{L}$ w.r.t z and λ. The partial derivative of $\mathcal{L}$ w.r.t to λ amounts to the ODE equality: $dz/dt = f(z, t, \theta)$. Obtaining the derivative of $\mathcal{L}$ w.r.t to $z(t)$ directly is cumbersome. However, recall the following lemma:

Proposition 4.1. *Let $f : \mathbb{R}^d \mapsto \mathbb{R}$ be a differentiable function at point P. Then, we have: $\frac{df}{dt}(P + tv)|_{t=0} = \nabla_p f.v$ This generalizes not only to lines but also to curves : $\frac{df}{dt}(c(t)) = \nabla_P f.c'(t)$*

We use Proposition 4.1 and we consider a small variation $z(t) + \epsilon\overline{z(t)}$, for some arbitrary function $\overline{z}$ we write:

$$\mathcal{L}(z(t)+\epsilon\overline{z(t)}, \theta, \lambda) = J(z(t)+\epsilon\overline{z(t)}) + \int_0^T \lambda(t)^T[\frac{dz(t) + \epsilon\overline{z(t)}}{dt} - f(z(t)+\epsilon\overline{z(t)}, t, \theta)]dt \tag{4.8}$$

We have for all t, we have: $\frac{\partial \mathcal{L}}{\partial z(t)} = \frac{d\mathcal{L}}{d\epsilon}|_{\epsilon=0}$. Thus:

$$\begin{aligned}
\frac{d\mathcal{L}}{d\epsilon}|_{\epsilon=0} &= \frac{\partial J}{\partial z}^\top \overline{z(T)} + \int_0^T \lambda(t)^T [\frac{d\overline{z}(T)}{dt} - \frac{\partial f(z(t),t,\theta)}{\partial z(t)} \overline{z(t)}] dt \\
&= \frac{\partial J}{\partial z(T)} \overline{z(T)} + \int_0^T \lambda(t)^\top [\frac{d\overline{z(t)}}{dt} - \frac{\partial f}{\partial z(t)} \overline{z(t)}] dt \\
&= \frac{\partial J}{\partial z(T)}^\top \overline{z(T)} + \int_0^T \lambda(t)^\top [\frac{d\overline{z(t)}}{dt} + \frac{\partial f}{\partial z(t)} \overline{z(t)}] + \lambda'(t)^\top \overline{z(t)} - \lambda'(t)^\top \overline{z(t)} dt \\
&= \frac{\partial J}{\partial z(T)}^\top \overline{z(T)} + [\lambda \overline{z(t)}]_0^T - \int_0^T \lambda'(t)^\top \overline{z(t)} + \lambda(t) \frac{\partial f}{\partial z(t)} \overline{z(t)}] \\
&= [\frac{\partial J}{\partial z(T)} + \lambda(T)]^\top \overline{z(T)} - \lambda(0)\overline{z(0)} - \int_0^T \overline{z(t)} [\lambda'(t) + \frac{\partial f}{\partial z(t)}^\top \lambda(t)] dt
\end{aligned}$$

Moreover, because the initial condition is fixed, $\overline{z(0)} = 0$. Thus, having computed all gradients, we can write the K.K.T. necessary condition for optimality for $z(t)$:

$$\frac{\partial J}{\partial z} + \lambda(T) = 0 \tag{4.9}$$

$$\lambda'(t) + \frac{\partial f}{\partial z}^\top \lambda(t) = 0 \tag{4.10}$$

We now derive the last term: $\frac{\partial \mathcal{L}}{\partial \theta}$, that writes as:

$$\frac{\partial \mathcal{L}}{\partial \theta} = -\int_0^T \lambda(t)^\top \frac{\partial f}{\partial \theta} \tag{4.11}$$

In this setting, the variable λ is called the negative adjoint variable and Equations (4.9) to (4.11) are sufficient to define the adjoint method as:

1. Solve z forward in time and compute the loss $J(z(T), y)$ and according to eq. (4.9): $\lambda(T) = -\frac{\partial J}{\partial z}$
2. Using eq. (4.10), solve $\lambda(t)$ backward in time (from $T \mapsto 0$) .
3. Using eq. (4.11), compute the gradient w.r.t θ.

Note that backward integration defined by eq. (4.10) can be conducted with similar computations than the forward integration for z.

The adjoint method for the optimization of Neural-ODE models presents several advantages. One interesting property of continuous adjoint based optimizers is the possibility to use non-differentiable solvers in the forward resolution steps as the gradients w.r.t to θ are analytically computed. Indeed, $\frac{\partial \mathcal{L}}{\partial \theta}$ depends only on the terminal value $z(T)$, not on the whole trajectory of $z(t)$. Moreover, it allows efficient memory use.

This approach is often called *optimize-then-discretize* as a reference to the analytical gradient computation scheme, the discretization occurring only for the numerical evaluation of the gradients. On the other-hand, the *discretize-then-optimize* approach discretizes the time integration in the ODE defined by eq. (4.6), then performs optimization based the actual computations, for instance using automatic differentiation. This setting amounts to the backpropagation algorithm as the gradients w.r.t θ are estimated based on the explicit computations of the NN. This approach can be computationally more costly but can lead to substantial gain in performances (Onken and Ruthotto 2020a). Indeed, the original adjoint for NN method proposed by (R. T. Q. Chen et al. 2018), involves both a forward (for loss computation) and a backward (for gradient estimation) resolution of $z(t)$. The discrepancy between the forward and the backward resolution could lead to numerical errors in the gradient estimation. In that perspective, Zhuang et al. (2020) and Gholaminejad et al. (2019) proposed to keep in memory the forward resolution of z (i.e. from $z(t=0) = x \mapsto z(T)$), and perform the backward integration of λ (from $T \mapsto 0$) between the timesteps defined by the forward resolution of z, guaranteeing the adequacy between the forward and the backward resolution while avoiding redundant computations.

4.2.3 ResNet and Neural-ODE Applied to Dynamical System Modeling

ResNet interpreted as an Euler step and Neural-ODE have provided a regain of interest for neural prediction of dynamical systems. It is nonetheless a long-standing issue in the community since Runge-Kutta computations rules in fact define recurrent computations that was used in early works such as (Y.-J. Wang and C.-T. Lin 1998). For instance, given $z(t)$, h and f, RK(2) prediction scheme writes as:

$$\begin{aligned}\hat{z}(t+h/2) &= z(t) + h/2 f(t, z(t)) \\ \hat{z'}(t+h/2) &= f(t+h/2, z_{t+h/2}) \\ \hat{z}(t+h) &= z(t) + h\hat{z'}(t+h/2)\end{aligned}$$

The application of Neural-ODE and the adjoint frameworks for dynamical systems is easy to understand as soon as in eq. (4.6) $J(z(t+1), y) = \|z(t+1) - x_{t+1}\|$ and $z(t) = x(t)$. It then amounts to a prediction tasks under the constraints that the estimated state z obeys an ODE defined by f. In that perspective (K. He et al. 2016; R. T. Q. Chen et al. 2018) paved the way to ML developments for dynamical systems estimation and prediction, for instance, when the system state is partially

observed (Ayed et al. 2019), or when the time-sampling of the observation is irregular (Rubanova et al. 2019).

4.2.4 ResNet and Neural-ODE Extensions

Residual Networks analogy with Euler discretization paved the way to more complex and expressive integration scheme based on NN, for instance by aggregating several data representations within one Residual Block, e.g. (Ouala et al. 2019; Xie et al. 2017; Y. Lu et al. 2018). Besides expressivity, (Haber and Ruthotto 2017) propose adapted constraints for ResNet for increasing training stability.

Since the (re)introduction of the adjoint method for the optimization of the Neural-ODE algorithm several refinements were introduced, palliating the drawback of the original method. Indeed, consider a classification problem of two different input images of the same class: Cauchy-Lipschitz ensures that to one $z(0)$ matches only one $z(T)$; However, in a classification task, to two initial data $z^i(0)$ and $z^j(0)$ may be associate to the same output $z(T)$ (the class label). Thus, a direct application of Neural-ODE for classification tasks is irrelevant. To palliate such a drawback, ANODE Dupont et al. (2019) augments the state of Neural-ODE to enable two different inputs to map the same output, enriching the space of possible dynamics then enabling to map two different inputs to the same output. (Finlay et al. 2020) also evidence the potential complexity of the integration path of vanilla Neural-ODE and implements optimal-transport inspired regularizations, penalizing both the norm of the learned f and its gradients $\nabla_z f$ along the integration paths.

While Neural-ODE and ResNet provide an interesting learning setting, the learned function remains nonetheless difficultly interpretable. 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 physics principles (Ayed et al. 2020). To palliate such observations, one can enforce physical properties by considering additional penalty besides the sole prediction loss. Of particular interest in natural sciences, Neural-solver (section 4.3) *Hybrid and Grey box* (section 4.4) approaches aim at opening the black box using stronger inductive biases to guide the model towards physically acceptable solutions. The first one, described in section 4.3, tackles directly the learning of the solution to a given PDE or ODE. The second one; described in section 4.4, models the evolution of a system state through time.

4.3 Solving Differential Equations with Neural Networks

As previously stated a significant part of the literature tackles the learning of the solution to a given ODE or PDE problem by directly parameterizing the solution by a NN.

A first research track investigated in the literature aims at learning the solution to a PDE using a NN without data. These approaches consider available initial and border conditions, but also the differential equation along with its parameters. This task amounts to solve the known differential equation given the border condition g and an initial condition u_0. We write u this (unknown) function depending on space $x \in \Omega \subseteq \mathbb{R}^d$ and time t, solution on the spatial domain Ω to the following PDE :

$$u_t = \mathcal{N}(x, u_x, u_{xx}, ...) \; \forall \, x \in \Omega \tag{4.12}$$

$$u(t = 0, x) = u_0(x), \quad \forall x \text{ and } u(t, x) = g(t, x), \forall x \in \partial\Omega, \forall t \tag{4.13}$$

where in this setting u_x denotes the partial derivative of u with respect to x and $\mathcal{N}$ is a function. Given samples in Ω for the initial condition u_0, and on $\partial\Omega$ for the border condition, denoted g, and knowing $\mathcal{N}$, the *Physics-Informed-Neural-Network* (PINN) approach developed in (Maziar Raissi et al. 2019a) and concurrently in (Sirignano and Spiliopoulos 2018), parameterizes u by a NN denoted u^θ, taking explicitly as input the spatial coordinates x and time t, denoted u^θ that aims at minimizing the following loss:

$$\begin{aligned}\mathcal{L}_{PINN} = \sum_{(x) \in \text{Initial condition}} \|u^\theta(0, x) - u_0(x)\| + \sum_{(x,t) \in \text{Border condition}} \|u^\theta(t, x) - g(t, x)\| \\ + \sum_{(x,t)} \|u^\theta_t(t, x) - \mathcal{N}(x, u^\theta_x(t, x), ...,)\|\end{aligned} \tag{4.14}$$

Where each term respectively accounts for the respect of the initial, border and differential conditions. Note that thanks to automatic differentiation the differential terms u^θ_t and u^θ_x are relatively cheap to compute.

A second learning task aims at learning jointly the solution u_θ and its differential equation $\mathcal{N}$. In this setting, one considers access to observation of the solution

sampled at interior domain points : (t_i, x_i) denoted u^i. Given a maximal order of differentiation of u with respect to x, $\mathcal{N}$ can be learned by minimizing:

$$\mathcal{L} = \sum_i \|u^\theta(t^i, x^i) - u^i\| + \lambda \|u_t^\theta(t_i, x_i) - \mathcal{N}(x_i, u_x^\theta(t_i, x_i), u_{xx}^\theta(t_i, x_i))\|, \tag{4.15}$$

In the case of eq. (4.15), $\mathcal{N}$ is unknown and can parameterized by a NN or learned using sparse dictionnary (Brunton et al. 2016).

These approaches, developed in (Maziar Raissi 2018), have been proven useful but on simple problems such as Burgers equation. Combining both approaches enables the resolution of inverse problems, such as recovering velocity fields from observations a scalar field (Maziar Raissi et al. 2020).

In practice, the aforementioned methodologies suffer from several issues (Krishnapriyan et al. 2021; S. Wang et al. 2021b). The most evidenced one is the bias towards low frequency, i.e. the model has difficulties to represents high frequencies (both spatial and temporal) in its predictions. Another issue, is that PINN models suffer from mode collapse, eventually learning the null solution except near the initial condition. Finally, difficulty in time and space extrapolation arise even for simple problems. The main reason for such difficulties have been proven not to be the expressivity of the neural-networks (Krishnapriyan et al. 2021), but rather comes from either the differential penalty that generates ill-conditioned gradients, or the $\|.\|_2$ as evidenced in a Neural Tangent Kernel analysis. These considerations led to potential solutions palliating the existing drawbacks of such meshless approaches: for instance curriculum training, i.e. learning the solution for physical parameters of increasing complexity (somehow similar to numerical continuation) or the use Fourier features (Krishnapriyan et al. 2021; S. Wang et al. 2021a).

4.4 Hybrid and Grey-Box Integration Methods

The previous approaches require neither a uniform sampling nor a fixed mesh but rather learn the solution to a PDE. On the contrary, several methods modeling data resulting from either in-situ observations or physical model simulations, are subject to a more structured and rigid sampling. They can then leverage specific architectures of NN such as convolution or Graph Neural-Network. Such constraints are common in numerical simulations of physical phenomena because they involve a discretization in space and time such as FEM, FVM or the Galerkin method.

Specific convolutional architectures has been developed to make deep auto-regressive models efficient to extrapolate in time (Geneva and Zabaras 2020). How-

ever, stronger inductive biases can be imposed on NN to mimic spatial-differential operator. For instance, Sobolev filters define differential operations. With appropriate constraints, convolutional filters can approximate differential operations, generalizing Sobolev filters and can be trained within a ResNet as in (Long et al. 2018; Long et al. 2019). Such a proposition defines a strong inductive bias and increase interpretability in residual models by structurally enforcing a PDE on the model. Indeed, given a differentiation filter one can easily identify the differentiation axis.

Closer to our work, early studies propose to mix a physical-hypothesis and data-driven component in order to generalize better on simulated chemical reaction data (Rico-Martinez et al. 1994; Thompson and Kramer 1994; Psichogios and Ungar 1992). Interestingly, all these works rely on some knowledge about the phenomena at stake and learn a NN in places where time-evolving parameters are unknown for instance as defined in the decomposition of eq. (4.3). Instead of chemical reactions, mechanistic principles often rely on the conservation of the Hamiltonian, providing practitioners with time-dependent equations on a reduced sets of variables. Recently, Greydanus et al. (2019) and S. Lee et al. (2021) 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 seminal researches of (Rico-Martinez et al. 1994; Thompson and Kramer 1994; Psichogios and Ungar 1992) have found recent echoes in the ML community and several works now aim at augmenting a dynamical assumption (originating from physics) with a learned data driven components. For instance, (Mehta et al. 2020; San and Maulik 2018; Young et al. 2017; Saha et al. 2020b) augment a physical models with a learned NN to compensate for inaccurate or unobserved physics. This can also be done within a Variational Auto-Encoder (VAE)-framework such as in (Linial et al. 2021; Tait and Damoulas 2020; Saemundsson et al. 2020). An interesting side-effect of latter VAE-based approaches is the possibility to sample from a latent space, that can either account for unobserved states or unknown parameters, which can be of great help for physicists for uncertainty quantification. However, for now such models only address low dimensional dynamical systems. These works differ mainly in the application and implementation as the principle remains either the refining of a physical prediction (i.e. learning a model that takes as input the physical prediction to better fit the data), or the additive decomposition between a physical assumption and a NN. Both are in practice equivalent since the learned residual dynamics in the first case can be recovered by simply subtracting the known physical input from the estimated output of the model. Lastly, inspired by transfer learning, (X. Jia et al. 2019) learns the profile

temperature in a lake (real data), by first pre-training the NN on simulated data obeying a simpler dynamics yet close in principle to the real one. The idea in this work is to leverage simpler data to guide the learned model towards acceptable solutions since the direct learning violates physical principles. Such a learning procedure can be thought as similar to curriculum learning in PINN frameworks.

Besides early and recent works on low dimensional systems, computational fluid dynamics is nowadays a crucial subject of interest for the community. In that perspective, several works have focused on transport phenomena, well-represented by the PDE with an advection component. For instance, (Bézenac et al. 2018a) use a closed-form solution of an advection-diffusion PDE in order to learn the associated velocity field. Other works such as (Tompson et al. 2017a; Wandel et al. 2021a) tackle the generalization of deep learning approaches and combine numerical solvers and NN in order to learn fluid dynamical models.

While the above works rely on regular grid sampling, thus operating on images, a novel line of research has emerged to address non uniform sampling, common in fluid simulation. Such data are better represented under the form of graphs. In order to learn on such structured data, Graph Neural Network (GNN) are a generalization of MLP and define operations either on the full-graph, or on local neighbors of each node in the graph defining graph convolutional neural networks. For a thorough review on existing methods and application see (J. Zhou et al. 2020). In that perspective, several works adopt the GNN framework. Particularly adapted for fluid simulations, this paradigm has been proven effective in order to predict a high resolution data from a coarse resolution model originating from a computational fluid dynamic simulator, alleviating significantly the computational cost for generating high resolution data. Specific GNN architectures adapted to the prediction of both the system state and the discretization are also an intense subject of study (Pfaff et al. 2021).

CHAPTER 5

GENERATIVE MODELS

Particularly useful to model complex data, generative models aim at sampling from a complex distribution p_X. Besides being a particular active field in the ML community, this task has several real world applications for natural sciences. For instance, weather forecast and analysis rely on the analog methods (Lorenz 1969), that constructs scenarii from already observed initial conditions that resemble our current observation. Generative methods can serve as samplers on the space of analogs, or even bypass the analog methods directly sampling the whole scenarii. Another application of Generative methods is the uncertainty quantification since some methods treat the data probabilistically, it provides us with quantities that can vouch for uncertainty. While traditional methods lack expressivity to model the variety of natural data, several techniques involving DL have been developed in order to perform this task.

This chapter is organized as follows, before describing the variational autoencoder and its application to dynamical system in section 5.2, we provide some details on the autoencoder in section 5.1 Finally, we describe recent generative methods, the GAN in section 5.3 and the normalizing flows in section 5.4 and how they can help model dynamical system.

5.1 Autoencoder

Autoencoders, originating from (Rumelhart and McClelland 1987) and further analyzed in (G. E. Hinton and R. R. Salakhutdinov 2006), propose a two-folded networks to enable dimensional reduction in the data. Let x be our data of interest, the auto-encoder is defined as two jointly trained neural networks (f_θ, g_ψ) as:

$$z = f_\theta(x)$$
$$\tilde{x} = g_\psi(z)$$

Such a network is trained using gradient descent minimizing the loss: $\mathcal{L} = \|x - \tilde{x}\|$. Strictly speaking, the autoencoder is not a generative model since it does not offer the possibility to sample from the data.

However, a simple refinement, the denoising autoencoder presents such a property. The denoising auto-encoder follows the same training rule as the conventional autoencoder except that the datum x is altered by some noise (it can be Gaussian additive, pixel-block etc.). (Yoshua Bengio et al. 2013c) propose a Markov Chain algorithm to enable the sampling from a denoising autoencoder. This algorithm paves the way to Stochastic Generative Networks popularized by the Variational Autoencoder and Generative Adversarial Networks.

5.1.1 Extension and Application to Dynamical Systems

The field of representation learning has greatly benefited from the advances and expressivity of the NN to embed high dimensional observations onto a smaller dimensional space (or manifold) in which the dynamics is supposed to be simpler or at least learnable.

Perhaps one a the most critical example is the Koopman operator. Koopman analysis of ODE proves, provided the adequate assumptions, the existence of a space (of potentially infinite dimension) in which the dynamics is linear. In that setting, the goal of the encoder network is to learn a space (maybe of infinite dimension) in which the system obeys a linear dynamics. The decoder is learned to project back into the space of observations. This property has been exploited in several ways. For instance, (Lusch et al. 2018) proposes a Koopman-operator based autoencoders involving 3 components: an encoder (ϕ), a matrix (K), a decoder (ϕ^{-1}), learned by minimizing a linear sum of 3 cost functions enforcing an autoencoding penalty ($\|\phi^{-1}(\phi(x)) - x\|$), a forecast penalty ($\|K^m\phi(X_t) - \phi(X_{t+m}\|$), and consistency penalty (the code of an observation at time $t+k$ must be equal to the code predicted from the observations at time t, i.e. $\|\phi^{-1}(K^m\phi(X_t)) - X_{t+m}]\|$). The above proposition assumes the possibility to truncate the infinite dimensional space into a finite dimensional one. (Takeishi et al. 2017) follows similar principles but relies on a sequence of observations as inputs. Non-linear latent dynamics can be learned by an ODE approximated using a NN for example helping to model irregularly sample time series (Rubanova et al. 2019).

5.2 The Variational AutoEncoder

With the aim to bridge representation power and learning capacities of autoencoders and the capacity to sample a prior distribution, (Diederik P. Kingma and Welling 2014a) introduced the VAE. The VAE framework has provided two major contributions. First it enables to learn the distribution of the data, given a

flexible enough parameterized prior distribution. It bypasses the complex task of directly learning to sample from the data distribution by instead considering the sampling from an (unobserved) latent space and learning the conditional $p(x|z)$. Second, it has proposed to estimate gradients from stochastic nodes through a simple and effective trick: *reparameterization*.

5.2.1 Posterior and Inference Models

Given observations $x \sim p^\star$, the objective of the VAE is to optimize the parameters θ of a distribution model p_θ to approximate $p^\star$. The data x is supposed to depend on a latent variable z, so that we have: $p_\theta(x) = \int_z p_\theta(x, z)dz$. The VAE framework addresses settings in which it is difficult or intractable to sample z given x or equivalently to get $p_\theta(x)$. Indeed, following Bayes rule, both intractabilities are related according to $p_\theta(z|x) = p_\theta(x, z)/p_\theta(x)$. The idea is to use amortized inference to learn both the posterior or recognition model $p_\theta(z|x)$ (denoted $q_\phi(z|x)$) and the inference model $p_\theta(x|z)$ using neural networks maximizing a workaround of the loglikelihood:

$$\begin{aligned}
\log p_\theta(x) &= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log p_\theta(x) \\
&= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x, z)}{p_\theta(z|x)} \\
&= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x, z)q_\phi(z|x)}{p_\theta(z|x)q_\phi(z|x)} \\
&= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x, z)}{q_\phi(z|x)} + \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log \frac{q_\phi(z|x)}{p_\theta(z|x)}
\end{aligned}$$

The first term, $\mathcal{L}_{\theta,\phi} = \mathbb{E}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x,z)}{q_\phi(z|x)}$ is called the ELBO (evidence lower bound) while the second term $\mathbb{E}_{z\sim q_\phi(z|x)} \log \frac{q_\phi(z|x)}{p_\theta(z|x)}$ defines a Kulblack-Leibler divergence: $KL(q_\phi(z|x), p(z|x))$. Because $KL(q_\phi(z|x), p(z|x)) > 0$, the ELBO $\mathcal{L}_{\theta,\phi} = \mathbb{E}_{q_\phi(z|x)}(\log p_\theta(x, z) - \log q_\phi(z|x))$ defines a lower bound of the original likelihood.

Practical optimization is conducted by employing another version of the Evidence Lower Bound (ELBO), writing $\mathcal{L}_{\phi,\theta}$ as:

$$\begin{aligned}
\mathcal{L}_{\phi,\theta} &= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x, z)}{q_\phi(z|x)} \\
&= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} \\
&= KL(q_\phi(z|x), p(z)) + \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log p_\theta(x|z) \qquad (5.1)
\end{aligned}$$

The first term in eq. (5.1) defines the distance between the (learned) conditional posterior $q_\phi(z|x)$ and the prior $p(z)$. When $q_\phi(z|x)$ and $p(z)$ are Gaussian, $KL(q_\phi(z|x), p(z))$ has an analytical form allowing for direct optimization. The second-term can be understood as: given z sampled according to $q_\phi(z|x)$, the goal of $p_\theta(x|z)$ is to provide an accurate estimate of x. Indeed with a Gaussian prior, $p_\theta(x|z)$ amounts to an ℓ_2 loss, i.e. a reconstruction loss.

5.2.2 The Reparameterization Trick for Gradient Estimation

In practice, one wants to use NN in order to parameterize $p_\theta(x|z)$ and $q_\phi(z|x)$. The gradients of $\mathcal{L}_{\phi,\theta}$ with respect to θ are not an issue here, as with the appropriate assumptions to permute ∇ and $\mathbb{E}$, we have:

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\phi,\theta} &= \nabla_\theta KL(q_\phi(z|x), p(z)) + \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log p_\theta(x|z) \\ &= \nabla_\theta \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \log p_\theta(x|z) \\ &= \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \nabla_\theta \log p_\theta(x|z)\end{aligned}$$

The difficulty in the optimization of eq. (5.1) lies in the fact that the parameters ϕ are variables of the sampled distribution $q_\phi(x|z)$. Therefore, we cannot permute ∇_ϕ and $\mathbb{E}$. The core idea of the reparameterization trick is not to directly sample z according to $z \sim q_\phi(z|x)$ but rather to write z as a differentiable transformation of ϵ, where ϵ is sampled from a fixed distribution, i.e. to write z as $z = g(\epsilon, x, \phi)$, where g is differentiable. That way we can write: $\mathbb{E}_{z\sim q_\phi(z|x)} \log p_\theta(x|z) = \mathbb{E}_\epsilon \log p_\theta(x|g(\epsilon, x, \phi))$. A common prior on $q_\phi(z|x)$ is the Gaussian one. Thus, in practice, given a data x, one learns a mean μ and a diagonal covariance vector σ using two NN respectively denoted f_μ and f_σ and in that case, with $\epsilon \sim \mathcal{N}(0, 1)$, $g(\epsilon, x, \phi) = f_\mu(x) + f_\sigma(x) \odot \epsilon$. We refer to (Diederik P. Kingma and Welling 2019) for an exhaustive presentation of the VAE framework and the properties of the gradients computed using the reparameterization trick.

5.2.3 Extension and Application to Dynamical Systems

Several variants of the original VAE framework, such as $\beta - $ VAE (Higgins et al. 2017) enable a finer control over the equilibrium in the losses defined by eq. (5.1). Adversarial autoencoders (Makhzani et al. 2016), propose another decomposition of $\log p_\theta(x)$ and learning a critic network that discriminates learned latent space $q_\phi(z|x)$ from the true prior $p(z)$.

The VAE-framework has also been leveraged to model time series as in (Gregor et al. 2019) and state-space models (Karl et al. 2017). Recent works learn either PDE parameters or a full dynamics using supervision over observed states (P. Y. Lu et al. 2020; Linial et al. 2021; Tait and Damoulas 2020; Saemundsson et al. 2020). Finally, because the variance of the latent code z is learned in the VAE-setting, the estimated variance of the predicted normal law can be assimilated as uncertainty quantification about the estimated code μ.

5.3 Generative Adversarial Networks

I. Goodfellow et al. (2014) developed Generative Adversarial Networks (GAN) introducing a novel way for data generation that originates from a simple principle. Consider a dataset $\mathcal{D}$, and a generation method G. The core idea of the adversarial generation is to learn G so as the samples it produces (denoted $\mathcal{D}_G$) are *indistinguishable* from samples of $\mathcal{D}$. Originally, the *indistinguishable* property is enforced using a critic network denoted D, often called discriminator that estimates the probability for a sample to come either from $\mathcal{D}$ or from the generated $\mathcal{D}_G$.

The learning problem is then framed as a classification task and the GAN aims at solving the following optimization problem: G must generate samples that "confuse" D while D must be able to recognize the samples originating from G. It is then a $\min - \max$ game formulated as:

$$\min_G \max_D V(G, D) = \mathop{\mathbb{E}}_{x \sim \mathcal{D}_G} \log(1 - D(x)) + \mathop{\mathbb{E}}_{x \sim \mathcal{D}} \log(D(x)) \tag{5.2}$$

In practice, G admits as entry a random (Normal) noise, so one does not sample $\mathcal{D}_G$ but rather $z \sim p_z$. The parameters of G and D are learned alternately following:

1. with $\mathcal{D}_G$ fixed, update D by performing a gradient in the direction $\nabla_D V(G, D)$.
2. with D fixed, update G by performing a gradient update in the direction $\nabla_G V(G, D)$.

Interestingly, the cost function defined by eq. (5.2) is shown to minimize a Jenson-Shannon divergence.

5.3.1 Extension and Application to Dynamical Systems

This learning framework has been experimentally proven useful for data generation and has seen a wide variety of refinements. The original GAN framework

has seen a wide variety of improvements. A major axis of improvement has been the investigation regarding the training loss leading to consider simpler distances such as least square (Mao et al. 2017). Considering the adversarial losses as a Wasserstein distance (from optimal transport theory) has provided practitioners with constraints to stabilize the training of GAN (Arjovsky et al. 2017; Gulrajani et al. 2017). Also beside unconditional data generation, the computer vision community has leveraged this learning framework for several tasks such as domain translation and style transfert (Isola et al. 2017b; J.-Y. Zhu et al. 2017), denoising (Bora et al. 2018), features and disentangled representation learning (Donahue et al. 2017; X. Chen et al. 2016b).

Closer to our subject of interest adversarial data generation has also been applied to video prediction (Mathieu et al. 2016a) for which classical regression methods fail at providing accurate and realistic output. Sequence generation of discrete tokens has also been trained using adversarial networks (Lantao Yu et al. 2017). Nonetheless, the generation of images or sequence that respects the underlying physics is still an open area of research. In that perspective, (Kashinath et al. 2019) propose a discriminator based on sample statistics, e.g. covariance matrix, for improved data generation of physical systems. To ensure a realistic timeline in the generated images, (Chu and Thuerey 2017) developed a GAN-based framework for fluid-flows prediction and generation that include a discriminator network that guides the generator towards accurate temporal changes.

5.4 Normalizing Flow

We now provide elements to understand a recent research track for unconditional generative models: *normalizing flows*. Normalizing flows propose to use a simple statistical property of transformation of measures. Let $z \in \mathbb{R}^d$ be a random variable of known density p_z and let $x = f(z)$ where f is an invertible transformation. According to the change of formula variables, we have [1]:

$$p_x(x) = p_z(f^{-1}(x))|\det J_{f^{-1}}(x)| \tag{5.3}$$

The application to density estimation and sampling is straightforward. Suppose, one accesses samples i.i.d. samples $x_1, .., x_n \sim p_{data}$ and let z a real random variables distributed according to p_z, where p_z is known, we have:

$$\log p(x_1, ..., x_n) = \sum_i \log p_{data}(x_i) = \sum_i \log(p_z(f^{-1}(x_i))) + \log|\det J_{f^{-1}}(x_i)| \tag{5.4}$$

1. Note that a more formal definition based on the change of measure can be conducted to derive an expression similar to eq. (5.3)

Crucial advantages of the use of normalizing flows appear in eq. (5.4). First of all, sampling from p_{data} is simplified. Indeed, if the flow f is known (or learned), $f(z)$ follows p_{data}. Moreover, thanks to maintained tractability, the likelihood of the observations are accessible. As soon as the defined flow is differentiable w.r.t to its parameters, f can be optimized using gradient descent so that the loglikelihood of the observation is maximized

Yet, Optimization difficulties arise in eq. (5.4). First of all the computation of the log-determinant in eq. (5.4) can be expensive. A simple way to dodge this computational difficulty is to design specifically f so that it has a diagonal (or triangular) Jacobian matrix, and thus so will f^{-1}. Also, the flow f must be invertible and at a reasonable computational cost. This condition is not ensured when dealing with NN, thus specific architectures have been defined in order to use normalizing flows for data generation.

The simplest class of normalizing flows, linear flows, can be considered, i.e. $x = Az + b$ which are computationally cheap for the inverse and the log-determinant computation. Naturally, the expressivity of linear flows is limited. Generalizing linear flows, planar transformation defined in (Rezende and Mohamed 2015) allows for more expressive data generation following: $f(z) = z + uh(w^T z + b)$. where h is a smooth non linearity and u is a vector. Sylvester transformations introduced in (Van Den Berg et al. 2018) generalize planar flows by considering U and w as matrices increasing expressivity. Other structural prior can be learned using carefully designed coupling layers, cutting the signal in half and combining both components using a transformation, producing a diagonal Jacobian facilitating the optimization of eq. (5.4) as (Dinh et al. 2015; Dinh et al. 2017). It gives for additive coupling layers from a signal $x \in \mathbb{R}^D$:

$$\begin{aligned} y_{1:d} &= x_{1:d} \\ y_{d+1:D} &= x_{d+1:D} + m(x_{1:d}) \end{aligned}$$

Adaptation to convolutional architectures have been developed in (Durk P Kingma and Dhariwal 2018). Also, flows can be combined in an autoregressive manner in order to increase the expressiveness of the generator NN, see (Durk P Kingma et al. 2016) We refer to (Kobyzev et al. 2021) for an exhaustive consideration on this subject.

A drawback of using flow is the dimension the computation. Indeed, to remain invertible, all computations must be of the dimension of the data, which can be prohibitive even for small images.

5.4.1 Extension and Application to Dynamical Systems

Normalizing flows are now tightly linked to dynamical systems through ODE, Residual Networks (K. He et al. 2016), and Langevin diffusion process.

Recall that ResNet are defined by: $X_t = X_{t-1} + f_\theta(X_{t-1})$, hence approximating a forward Euler-step. Coupling Layers can be designed in a ResNet fashion, for example as in (Gomez et al. 2017; Jacobsen et al. 2018);

$$y_1 = x_1 + f(x_2)$$
$$y_2 = x_2 + g(y_1)$$

Behrmann et al. (2019a) do not rely on the traditional splitting/coupling scheme but exhibit a fixed point iteration method to recover the inverse f^{-1} thanks to the invertibility retrieved by constraining the Lipschitz-norm of the learned residual blocks below 1. The log-determinant estimation is conducted using a stochastic approximation.

The continuous version of ResNet, i.e. NeuralODE defined by (R. T. Q. Chen et al. 2018), does not need structural constraints to ensure bijectivity. Indeed, by definition, the flow of an ODE is a bijection thanks to Cauchy Lipschitz Lemma: provided that f is Lipschitz, for any initial condition z_0, it exists one and only one solution to $\frac{dz}{dt} = f(z,t)$ with $z(t=0) = z_0$. (Grathwohl et al. 2019) exploits this property to define an instantaneous change of variables:

$$\frac{\partial \log p(z(t))}{\partial t} = -\mathrm{Tr}\left(\frac{\partial f(z(t),t)}{\partial z(t)}\right)$$

Hence bypassing the computational complexity of calculating the $\det$ of a Jacobian, for example using the identity: $\mathrm{Tr}(A) = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}(\epsilon^T A\epsilon)$.

A last dynamical system inspiration to construct normalizing flows originate from Langevin Stochastic Differential Equation (SDE), describing the speed v of a particle of mass m subject to friction (P. 1908):

$$m\frac{dv}{dt} = -\lambda v + \eta t$$

The above equation defines the density p of finding the particle at state x at time t through the Fokker-Planck equation that generally writes as:

$$\frac{\partial p(x,t)}{\partial t} = -\nabla_x \cdot (b(x,t)p(x,t)) + \sum_{i,j} \frac{\partial^2}{\partial x_i \partial x_j} D_{i,j}(x,t)p(x,t)$$

The core idea in this literature is to provide a framework for learning efficiently b and $D_{i,j}$ as in (C. Chen et al. 2018).

Besides, the use of dynamical model formalism and ODE to help modeling fixed densities, normalizing flows are also used to learn stochastic dynamics as in (Urain et al. 2020) and for time series prediction (R. Deng et al. 2020)

Part III

DISENTANGLING DYNAMICS

Chapter 6

PDE-DRIVEN SPATIOTEMPORAL DISENTANGLEMENT

Chapter abstract

In this chapter, we investigate the possibility to increase the interpretability in spatio-temporal prediction systems. Born in variational auto-encoding, a core idea in statistical learning to increase interpretability is to isolate independent factors of variations. Such a separation is denoted disentanglement. For dynamical systems, a specific form disentanglement takes place: spatio-temporal disentanglement that amounts to separating variations that accounts for the dynamics from variations in the content. We propose in this chapter a novel learning framework to learn to predict spatio-temporal systems in a disentangled manner. Specifically, we leverage the separation of variables, an analytical tool to solve PDE in order to derive a grounded algorithm.

The work in this chapter has led to the publication of a conference paper:

Jérémie Donà, Jean-Yves Franceschi, Sylvain Lamprier, and Patrick Gallinari (May 2021). "PDE-Driven Spatiotemporal Disentanglement". In: *The Ninth International Conference on Learning Representations*. International Conference on Representation Learning. Vienne, Austria. URL: https://hal.archives-ouvertes.fr/hal-02911067

6.1 Introduction

The interest of the machine learning community in physical phenomena has substantially grown for the last few years (Shi et al. 2015; Long et al. 2018; Greydanus et al. 2019). In particular, an increasing amount of works studies the challenging problem of modeling the evolution of dynamical systems, with applications in sensible domains like climate or health science, making the understanding of physical phenomena a key challenge in machine learning. To this end, the community has successfully leveraged the formalism of dynamical systems and their

associated differential formulation as powerful tools to specifically design efficient prediction models. In this work, we aim at studying this prediction problem with a principled and general approach, through the prism of Partial Differential Equations (PDEs), with a focus on learning spatiotemporal disentangled representations.

Prediction via spatiotemporal disentanglement was first studied in video prediction works, in order to separate static and dynamic information (Denton and Birodkar 2017) for prediction and interpretability purposes. Existing models are particularly complex, involving either adversarial losses or variational inference. Furthermore, their reliance on Recurrent Neural Networks (RNNs) hinders their ability to model spatiotemporal phenomena (Yıldız et al. 2019; Ayed et al. 2020; Franceschi et al. 2020). Our proposition addresses these shortcomings with a simplified and improved model by grounding spatiotemporal disentanglement in the PDE formalism.

Spatiotemporal phenomena obey physical laws such as the conservation of energy, that lead to describe the evolution of the system through PDEs. Practical examples include the conservation of energy for physical systems (Hamilton 1835), or the equation describing constant illumination in a scene (Horn and Schunck 1981) for videos that has had a longstanding impact in computer vision with optical flow methods (Dosovitskiy et al. 2015; Finn et al. 2016). We propose to model the evolution of partially observed spatiotemporal phenomena with unknown dynamics by leveraging a formal method for the analytical resolution of PDEs: the functional separation of variables (Miller 1988). Our framework formulates spatiotemporal disentanglement for prediction as learning a separable solution, where spatial and dynamic information are represented in separate variables. Besides offering a novel interpretation of spatiotemporal disentanglement, it confers simplicity and performance compared to existing methods: disentanglement is achieved through the sole combination of a prediction objective and regularization penalties, and the temporal dynamics is defined by a learned Ordinary Differential Equation (ODE). We experimentally demonstrate the applicability, disentanglement capacity and forecasting performance of the proposed model on various spatiotemporal phenomena involving standard physical processes and synthetic video datasets against prior state-of-the-art models.

6.2 Related Work

Our contribution deals with two main directions of research: spatiotemporal disentanglement and the coupling of neural networks and PDEs.

Spatiotemporal Disentanglement. Disentangling factors of variations is an essential representation learning problem (Yoshua Bengio et al. 2013a). Its cardinal formulation for static data has been extensively studied, with state-of-the-art solutions (Locatello et al. 2019) being essentially based on Variational Autoencoders (VAEs; Diederik P. Kingma and Welling 2014b; Rezende et al. 2014). As for sequential data, several disentanglement notions have been formulated, ranging from distinguishing objects in a video (Hsieh et al. 2018; Steenkiste et al. 2018) to separating and modeling multi-scale dynamics (Hsu et al. 2017; Yingzhen and Mandt 2018).

We focus in this work on the dissociation of the dynamics and visual aspects for spatiotemporal data. Even in this case, dissociation can take multiple forms. Examples in the video generation community include decoupling the foreground and background (Vondrick et al. 2016), constructing structured frame representations (Villegas et al. 2017b; Minderer et al. 2019; Z. Liu et al. 2019), extracting physical dynamics (Le Guen and Thome 2020), or latent modeling of dynamics in a state-space manner (Fraccaro et al. 2017; Franceschi et al. 2020). Closer to our work, (Denton and Birodkar 2017), (Villegas et al. 2017a) and (Hsieh et al. 2018) introduced in their video prediction models explicit latent disentanglement of static and dynamic information obtained using adversarial losses (I. Goodfellow et al. 2014) or VAEs. Disentanglement has also been introduced in more restrictive models relying on data-specific assumptions (Kosiorek et al. 2018; Jaques et al. 2020), and in video generation (Tulyakov et al. 2018). We aim in this work at grounding and improving spatiotemporal disentanglement with more adapted inductive biases by introducing a paradigm leveraging the functional separation of variables resolution method for PDEs.

Spatiotemporal Prediction and PDE-based Neural Network Models. An increasing number of works combining neural networks and differential equations for spatiotemporal forecasting have been produced for the last few years. Some of them show substantial improvements for the prediction of dynamical systems or videos compared to standard RNNs by defining the dynamics using learned ODEs (Rubanova et al. 2019; Yıldız et al. 2019; Ayed et al. 2020; Le Guen and Thome 2020), following (R. T. Q. Chen et al. 2018), or adapting them to stochastic data (Ryder et al. 2018; X. Li et al. 2020; Franceschi et al. 2020). Most PDE-based spatiotemporal models exploit some prior physical knowledge. It can induce the structure of the prediction function (Brunton et al. 2016; Avila Belbute-Peres et al. 2018) or specific cost functions, thereby improving model performances. For instance, (Bézenac et al. 2018a) shape their prediction function with an advection-diffusion mechanism, and (Long et al. 2018; Long et al. 2019) estimate PDEs and their solutions by learning convolutional filters proven to approximate differential operators. (Greydanus et al. 2019), (Z. Chen et al. 2020a) and (Toth et al. 2020)

introduce non-regression losses by taking advantage of Hamiltonian mechanics (Hamilton 1835), while (Tompson et al. 2017b) and (Maziar Raissi et al. 2020) combine physically inspired constraints and structural priors for fluid dynamic prediction. Our work deepens this literature by establishing a novel link between a resolution method for PDEs and spatiotemporal disentanglement, thereby introducing a data-agnostic model leveraging any static information in observed phenomena.

6.3 Background: Separation of Variables

Solving high-dimensional PDEs is a difficult analytical and numerical problem (Bungartz and Griebel 2004). Variable separation aims at simplifying it by decomposing the solution, e.g., as a simple combination of lower-dimensional functions, thus reducing the PDE to simpler differential equations.

6.3.1 Simple Case Study

Let us introduce this technique through a standard application, with proofs in appendix B.1.1, on the one-dimensional heat diffusion problem (Fourier 1822), consisting in a bar of length L, whose temperature at time t and position x is denoted by $u(x,t)$ and satisfies:

$$\frac{\partial u}{\partial t} = c^2 \frac{\partial^2 u}{\partial x^2}, \qquad u(0,t) = u(L,t) = 0, \qquad u(x,0) = f(x). \tag{6.1}$$

Suppose that a solution u is product-separable, i.e., it can be decomposed as: $u(x,t) = u_1(x) \cdot u_2(t)$. Combined with eq. (6.1), it leads to $c^2 u_1''(x)/u_1(x) = u_2'(t)/u_2(t)$. The left- and right-hand sides of this equation are respectively independent from t and x. Therefore, both sides are constant, and solving both resulting ODEs gives solutions of the form, with $\mu \in \mathbb{R}$ and $n \in \mathbb{N}$:

$$u(x,t) = \mu \sin\big(n\pi x/L\big) \times \exp\Big(-\big(cn\pi/L\big)^2 t\Big). \tag{6.2}$$

The superposition principle and the uniqueness of solutions under smoothness constraints allow then to build the set of solutions of eq. (6.1) with linear combinations of separable solutions (Le Dret and Lucquin 2016). Besides this simple example, separation of variables can be more elaborate.

6.3.2 Functional Separation of Variables

The functional separation of variables (Miller 1988) generalizes this method. Let u be a function obeying a given arbitrary PDE. The functional variable separation method amounts to finding a parameterization z, a functional U, an entangling function ξ, and representations ϕ and ψ such that:

$$z = \xi\big(\phi(x), \psi(t)\big), \qquad u(x,t) = U(z). \tag{6.3}$$

Trivial choices $\xi = u$ and identity function as U, ϕ and ψ ensure the validity of this reformulation. Finding suitable ϕ, ψ, U, and ξ with regards to the initial PDE can facilitate its resolution by inducing separate simpler PDEs on ϕ, ψ, and U. For instance, product-separability is retrieved with $U = \exp$. General results on the existence of separable solutions have been proven (Miller 1983), though their uniqueness depends on the initial conditions and the choice of functional separation (Polyanin 2020).

Functional separation of variables finds broad applications. It helps to solve refinements of the heat equation, such as generalizations with an advection term (see appendix B.1.2) or with complex diffusion and source terms forming a general transport equation (H. Jia et al. 2008). Besides the heat equation, functional separation of PDEs is also applicable in various physics fields like reaction-diffusion with non-linear sources or convection-diffusion phenomena (Polyanin 2019; Polyanin and Zhurov 2020), Hamiltonian physics (Benenti 1997), or even general relativity (Kalnins et al. 1992).

Reparameterizations such as eq. (6.3) implement a separation of spatial and temporal factors of variations, i.e., spatiotemporal disentanglement. We introduce in the following a learning framework based on this general method.

6.4 Proposed Method

We propose to model spatiotemporal phenomena using the functional variable separation formalism. We first describe our notations and then derive a principled model and constraints from this method.

6.4.1 Problem Formulation Through Separation of Variables

We consider a distribution $\mathcal{P}$ of observed spatiotemporal trajectories and corresponding observation samples $v = (v_{t_0}, v_{t_0+\Delta t}, \dots, v_{t_1})$, with $v_t \in \mathcal{V} \subseteq \mathbb{R}^m$ and

$t_1 = t_0 + \nu\Delta t$. Each sequence $v \sim \mathcal{P}$ corresponds to an observation of a dynamical phenomenon, assumed to be described by a hidden functional u_v (also denoted by u for the sake of simplicity) of space coordinates $x \in \mathcal{X} \subseteq \mathbb{R}^s$ and time $t \in \mathbb{R}$ that characterizes the trajectories. More precisely, u_v describes an unobserved continuous dynamics and v corresponds to instantaneous discrete spatial measurements associated to this dynamics. Therefore, we consider that v_t results from a time-independent function ζ of the mapping $u_v(\cdot, t)$. For example, v might consist in temperatures measured at some points of the sea surface, while u_v would be the complete ocean circulation model. In other words, v provides a partial information about u_v and is a projection of the full dynamics. We seek to learn a model which, when conditioned on prior observations, can predict future observations.

To this end, we posit that the state u of each observed trajectory v is driven by a hidden PDE, shared among all trajectories; we discuss this assumption in details in appendix B.3.1. Learning such a PDE and its solutions would then allow us to model observed trajectories v. However, directly learning solutions to high-dimensional unknown PDEs is a complex task (Bungartz and Griebel 2004; Sirignano and Spiliopoulos 2018). We aim in this work at simplifying this resolution. We propose to do so by relying on the functional separation of variables of eq. (6.3), in order to leverage a potential separability of the hidden PDE. Therefore, analogously to eq. (6.3), we propose to formulate the problem as learning observation-constrained ϕ, ψ and U, as well as ξ and ζ, such that:

$$z = \xi\big(\phi(x), \psi(t)\big), \qquad u(x,t) = U(z), \qquad v_t = \zeta\big(u(\cdot,t)\big), \tag{6.4}$$

with ϕ and ψ allowing to disentangle the prediction problem. In the formalism of the functional separation of variables, this amounts to decomposing the full solution u, thereby learning a spatial PDE on ϕ, a temporal ODE on ψ, and a PDE on U, as well as their respective solutions.

6.4.2 Fundamental Limits and Relaxation

Directly learning u is, however, a restrictive choice. Indeed, when formulating PDEs such as in eq. (6.1), spatial coordinates (x, y, etc.) and time t appear as variables of the solution. Yet, unlike in fully observable phenomena studied by (Sirignano and Spiliopoulos 2018) and (Maziar Raissi 2018), directly accessing theses variables in practice can be costly or infeasible in our partially observed setting. In other words, the nature and number of these variables are unknown. For example, the dynamic of the observed sea surface temperature is highly dependent on numerous unobserved variables such as temperature at deeper levels or wind intensity. Explicitly taking into account these unobserved variables

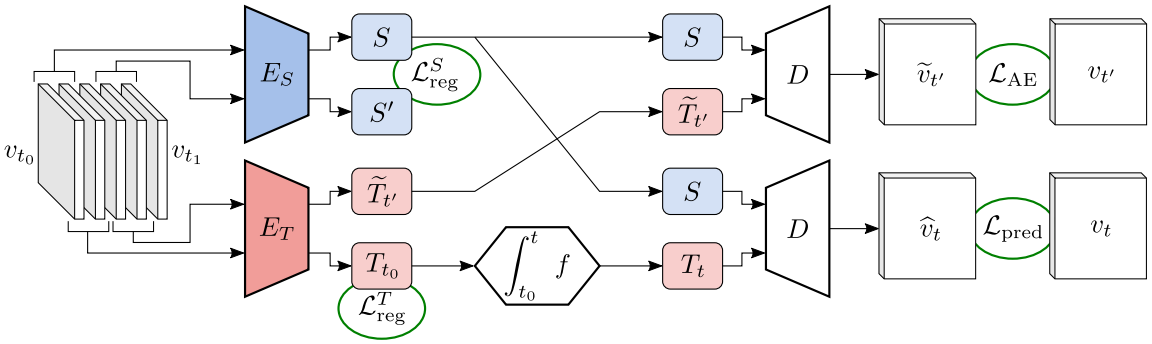


Figure 6.1. – Computational graph of the proposed model. E_S and E_T take contiguous observations as input; time invariance is enforced on S; the evolution of T_t is modeled with an ODE and is constrained to coincide with E_T; T_{t_0} is regularized; forecasting amounts to decoding from S and T_t.

can only be done with prior domain knowledge. To maintain the generality of the proposed approach, we choose not to make any data-specific assumption on these unknown variables.

We overcome these issues by eliminating the explicit modeling of spatial coordinates by learning dynamic and time-invariant representations accounting respectively for the time-dependent and space-dependent parts of the solution. Indeed, eq. (6.4) induces that these spatial coordinates, hence the explicit resolution of PDEs on u or U, can be ignored, as it amounts to learning ϕ, ψ and D such that:

$$v_t = (\zeta \circ U \circ \xi)\big(\phi(\cdot), \psi(t)\big) = D\big(\phi, \psi(t)\big). \tag{6.5}$$

In order to manipulate functionals ϕ and ψ in practice, we respectively introduce learnable time-invariant and time-dependent representations of ϕ and ψ, denoted by S and T, such that:

$$\phi \equiv S \in \mathcal{S} \subseteq \mathbb{R}^d, \qquad \psi \equiv T : t \mapsto T_t \in \mathcal{T} \subseteq \mathbb{R}^p, \tag{6.6}$$

where the dependence of $\psi \equiv T$ on time t will be modeled using a temporal ODE following the separation of variables, and the function ϕ, and consequently its spatial PDE, are encoded into a vectorial representation S. Besides their separation of variables basis, the purpose of S and T is to capture spatial and motion information of the data. For instance, S could encode static information such as objects appearance, while T typically contains motion variables.

S and T_{t_0}, because of their dependence on v in eqs. (6.5) and (6.6), are inferred from an observation history, or conditioning frames, $V_\tau(t_0)$, where $V_\tau(t) = (v_t, v_{t+\Delta t}, \ldots, v_{t+\tau\Delta t})$, using respectively encoder networks E_S and E_T. We parameterize D of eq. (6.5) as a neural network that acts on both S and T_t, and outputs

the estimated observation $\widehat{v}_t = D(S, T_t)$. Unless specified otherwise, S and T_t are fed concatenated into D, which then learns the parameterization ξ of their combination.

6.4.3 Temporal ODE

The separation of variables allows us to partly reduce the complex task of learning and integrating PDEs to learning and integrating an ODE on ψ, which has been extensively studied in the literature, as explained in section 6.2. We therefore model the evolution of T_t, thereby the dynamics of our system, with a first-order ODE:

$$\frac{\partial T_t}{\partial t} = f(T_t) \qquad \Leftrightarrow \qquad T_t = T_{t_0} + \int_{t_0}^{t} f(T_{t'})\,\mathrm{d}t' \tag{6.7}$$

Note that the first-order ODE assumption can be taken without loss of generality since any ODE is equivalent to a higher-dimensional first-order ODE. Following (R. T. Q. Chen et al. 2018), f is implemented by a neural network and eq. (6.7) is solved with an ODE resolution scheme. Suppose initial ODE conditions S and T_{t_0} have been computed with E_S and E_T. This leads to the following simple forecasting scheme, enforced by the corresponding regression loss:

$$\widehat{v}_t = D\left(S, T_{t_0} + \int_{t_0}^{t} f(T_{t'})\,\mathrm{d}t'\right), \quad \mathcal{L}_{\text{pred}} = \frac{1}{\nu+1}\sum_{i=0}^{\nu}\frac{1}{m}\|\widehat{v}_{t_0+i\Delta t} - v_{t_0+i\Delta t}\|_2^2, \tag{6.8}$$

where $\nu+1$ is the number of observations, and m is the dimension of the observed variables v.

eq. (6.8) ensures that the evolution of T is coherent with the observations; we should enforce its consistency with E_T. Indeed, the dynamics of T_t is modeled by eq. (6.7), while only its initial condition T_{t_0} is computed with E_T. However, there is no guaranty that T_t, computed via integration, matches $E_T\big(V_\tau(t)\big)$ at any other time t, while they should in principle coincide. We introduce the following autoencoding constraint mitigating their divergence, thereby stabilizing the evolution of T:

$$\mathcal{L}_{\text{AE}} = \frac{1}{m}\left\| D\Big(S, E_T\big(V_\tau(t_0 + i\Delta t)\big)\Big) - v_{t_0+i\Delta t}\right\|_2^2, \quad \text{with } i \sim \mathcal{U}\big([\![0, \nu-\tau]\!]\big). \tag{6.9}$$

6.4.4 Spatiotemporal Disentanglement

As indicated hereinabove, the spatial PDE on ϕ is assumed to be encoded into S. Nonetheless, since S is inferred from an observation history, we need to explicitly enforce its time independence. In the PDE formalism, this is equivalent to:

$$\frac{\partial E_S\big(V_\tau(t)\big)}{\partial t} = 0 \qquad \Leftrightarrow \qquad \int_{t_0}^{t_1-\tau\Delta t} \left\| \frac{\partial E_S\big(V_\tau(t)\big)}{\partial t} \right\|_2^2 \mathrm{d}t = 0. \tag{6.10}$$

However, enforcing eq. (6.10) raises two crucial issues. Firstly, in our partially observed setting, there can be variations of observable content, for instance when an object conceals another one. Therefore, strictly enforcing a null time derivative is not desirable as it prevents E_S to extract accessible information that could be obfuscated in the sequence. Secondly, estimating this derivative in practice in our setting is unfeasible and costly because of the coarse temporal discretization of the data and the computational cost of E_S; see appendix B.2 for more details. We instead introduce a discretized penalty in our minimization objective, discouraging variations of content between two distant time steps, with d being the dimension of S:

$$\mathcal{L}^S_{\text{reg}} = \frac{1}{d} \left\| E_S\big(V_\tau(t_0)\big) - E_S\big(V_\tau(t_1 - \tau\Delta t)\big) \right\|_2^2. \tag{6.11}$$

It allows us to overcome the previously stated issues by not enforcing a strict invariance of S and removing the need to estimate any time derivative. Note that this formulation actually originates from eq. (6.10) using the Cauchy-Schwarz inequality (see appendix B.2 for a more general derivation).

Abstracting the spatial ODE on ϕ from eq. (6.4) into a generic representation S leads, without additional constraints, to an underconstrained problem where spatiotemporal disentanglement cannot be guaranteed. Indeed, E_S can be set to zero to satisfy eq. (6.11) without breaking any prior constraint, because static information is not prevented to be encoded into T. Accordingly, information in S and T needs to be segmented.

Thanks to the design of our model, it suffices to ensure that S and T are disentangled at initial time t_0 for them be to be disentangled at all t. Indeed, the mutual information between two variables is preserved by invertible transformations. eq. (6.7) is an ODE and f, as a neural network, is Lipschitz-continuous, so the ODE flow $T_t \mapsto T_{t'}$ is invertible. Therefore, disentanglement between S and T_t, characterized by a low mutual information between both variables, is preserved through time; see appendix B.3 for a detailed discussion. We thus only

constrain the information quantity in T_{t_0} by using a Gaussian prior to encourage it to exclusively contain necessary dynamic information:

$$\mathcal{L}^T_{\text{reg}} = \frac{1}{p}\|T_{t_0}\|_2^2 = \frac{1}{p}\left\|E_T\big(V_\tau(t_0)\big)\right\|_2^2. \tag{6.12}$$

6.4.5 Loss Function

The minimized loss is a linear combination of eqs. (6.8), (6.9), (6.11) and (6.12):

$$\mathcal{L}(v) = \mathop{\mathbb{E}}_{v\sim\mathcal{P}}\Big[\lambda_{\text{pred}}\mathcal{L}_{\text{pred}} + \lambda_{\text{AE}}\cdot\mathcal{L}_{\text{AE}} + \lambda^S_{\text{reg}}\cdot\mathcal{L}^S_{\text{reg}} + \lambda^T_{\text{reg}}\cdot\mathcal{L}^T_{\text{reg}}\Big], \tag{6.13}$$

as illustrated in fig. 6.1. In the following, we conventionally set $\Delta t = 1$. Note that the presented approach could be generalized to irregularly sampled observation times thanks to the dedicated literature (Rubanova et al. 2019), but this is out of the scope of this paper.

6.5 Experiments

We study in this section the experimental results of our model on various spatiotemporal phenomena with physical, synthetic video and real-world datasets, which are briefly presented in this section and in more details in appendix B.4. We demonstrate the relevance of our model with ablation studies and its performance by comparing it with more complex state-of-the-art models. Performances are assessed thanks to standard metrics (Denton and Fergus 2018; Le Guen and Thome 2020) Mean Squared Error (MSE, lower is better) or its alternative Peak Signal-to-Noise Ratio (PSNR, higher is better), and Structured Similarity (SSIM, higher is better). We refer to appendix B.6 for more experiments and prediction examples, to appendix B.5 for training information and to the supplementary material for the corresponding code [1] and datasets.

6.5.1 Physical Datasets: Wave Equation and Sea Surface Temperature

We first investigate two synthetic dynamical systems and a real-world dataset in order to show the advantage of PDE-driven spatiotemporal disentanglement for

1. Our source code is also publicly released at the following URL: https://github.com/JeremDona/spatiotemporal_variable_separation.

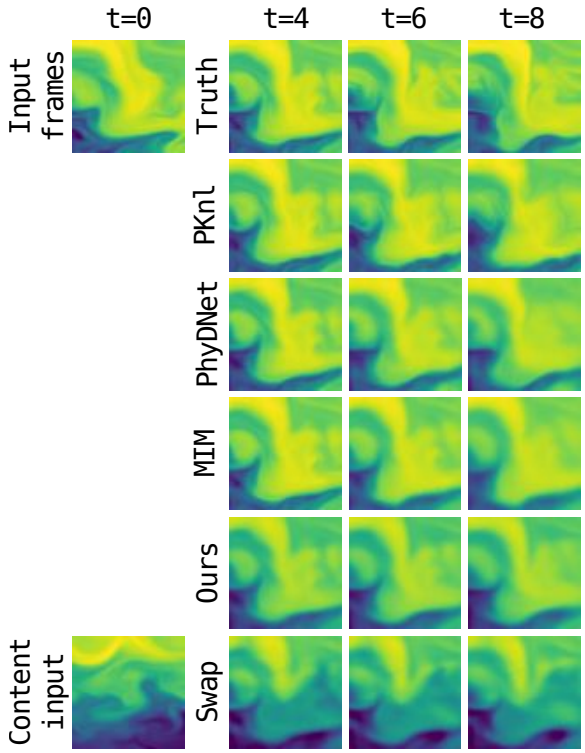


Figure 6.2. – Example of predictions of compared models on SST. Content swap preserves the location of extreme temperature regions which determine the movement while modifying the magnitude of all regions, especially in temperate areas.

forecasting physical phenomena. To analyze our model, we first lean on the wave equation, occurring for example in acoustic or electromagnetism, with source term like (Saha et al. 2020a), to produce the WaveEq dataset consisting in 64×64 normalized images of the phenomenon. We additionally build the WaveEq-100 dataset by extracting 100 pixels, chosen uniformly at random and shared among sequences, from WaveEq frames; this experimental setting can be thought of as measurements from sensors partially observing the phenomenon. We also test and compare our model on the real-world dataset SST, derived from the data assimilation engine NEMO (Madec 2008) and introduced by (Bézenac et al. 2018a), consisting in 64×64 frames showing the evolution of the sea surface temperature. Modeling its evolution is particularly challenging as its dynamic is highly non-linear, chaotic, and involves several unobserved quantities (e.g., forcing terms).

We compare our model on these three datasets to its alternative version with S removed and integrated into T, thus also removing $\mathcal{L}^S_{\text{reg}}$ and $\mathcal{L}^T_{\text{reg}}$. We also include the state-of-the-art PhyDNet (Le Guen and Thome 2020), MIM (Y. Wang et al. 2019b), SVG (Denton and Fergus 2018) and SST-specific PKnl (Bézenac et al. 2018a) in the comparison on SST; only PhyDNet and PKnl were originally tested on this dataset by their authors. Results are compiled in table 6.1 and an example of prediction is depicted in fig. 9.2.

Table 6.1. – Forecasting performance on WaveEq-100, WaveEq and SST of compared models with respect to indicated prediction horizons. Bold scores indicate the best performing method.

Models	WaveEq-100	WaveEq	SST			
	MSE		MSE		SSIM	
	$t+40$	$t+40$	$t+6$	$t+10$	$t+6$	$t+10$
PKnl	—	—	1.28	2.03	0.6686	0.5844
PhyDNet	—	—	1.27	1.91	0.5782	0.4645
SVG	—	—	1.51	2.06	0.6259	0.5595
MIM	—	—	0.91	1.45	0.7406	0.6525
Ours	$\mathbf{4.33 \times 10^{-5}}$	$\mathbf{1.44 \times 10^{-4}}$	**0.86**	**1.43**	**0.7466**	**0.6577**
Ours (without S)	1.33×10^{-4}	5.09×10^{-4}	0.95	1.50	0.7204	0.6446

On these three datasets, our model produces more accurate long-term predictions with S than without it. This indicates that learning an invariant component facilitates training and improves generalization. The influence of S can be observed by replacing the S of a sequence by another one extracted from another sequence, changing the aspect of the prediction, as shown in fig. 9.2 (swap row). We provide in appendix B.6 further samples showing the influence of S in the prediction. Even though there is no evidence of intrinsic separability in SST, our trained algorithm takes advantage of its time-invariant component. Indeed, our model outperforms PKnl despite the data-specific structure of the latter, the stochastic SVG and the high-capacities PhyDNet and MIM model, whereas removing its static component suppresses our advantage.

We highlight that MIM is a computationally-heavy model that manipulates in an autoregressive way 64 times larger latent states than ours, hence its better reconstruction ability at the first time step. However, its sharpness and movement gradually vanish, explaining its lower performance than ours. We refer to appendix B.6.3 for additional discussion on the application of our method and its performance on SST.

6.5.2 A Synthetic Video Dataset: Moving MNIST

We also assess the prediction and disentanglement performance of our model on the Moving MNIST dataset (Srivastava et al. 2015) involving MNIST digits (LeCun et al. 1998) bouncing over frame borders. This dataset is particularly challenging in the literature for long-term prediction tasks. We compare our model to

Table 6.2. – Prediction and content swap scores of all compared models on Moving MNIST. Bold scores indicate the best performing method.

Models	Pred. $(t+10)$		Pred. $(t+95)$		Swap $(t+10)$		Swap $(t+95)$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
SVG	18.18	0.8329	12.85	0.6185	—	—	—	—
MIM	**24.16**	0.9113	16.50	0.6529	—	—	—	—
DrNet	14.94	0.6596	12.91	0.5379	14.12	0.6206	12.80	0.5306
DDPAE	21.17	0.8814	13.56	0.6446	**18.44**	0.8256	13.25	0.6378
PhyDNet	23.12	**0.9128**	16.46	0.3878	12.04	0.5572	13.49	0.2839
Ours	21.70	0.9088	**17.50**	**0.7990**	18.42	**0.8368**	**16.50**	**0.7713**

competitive baselines: the non-disentangled SVG (Denton and Fergus 2018) and MIM (Y. Wang et al. 2019b), as well as forecasting models with spatiotemporal disentanglement ablities DrNet (Denton and Birodkar 2017), DDPAE (Hsieh et al. 2018) and PhyDNet. We highlight that all these models leverage powerful machine learning tools such as adversarial losses, VAEs and high-capacity temporal architectures, whereas ours is solely trained using regression penalties and small-size latent representations. We perform as well a full ablation study of our model to confirm the relevance of the introduced method.

Results reported in table 6.2 and illustrated in fig. 6.3 correspond to two tasks: prediction and disentanglement, at both short and long-term horizons. Disentanglement is evaluated via content swapping, which consists in replacing the content representation of a sequence by the one of another sequence, which should result for a perfectly disentangled model in swapping digits of both sequences. This is done by taking advantage of the synthetic nature of this dataset that allows us to implement the ground truth content swap and compare it to the generated swaps of the model.

Reported results show the advantage of our model against all baselines. Long-term prediction challenges them as their performance and predictions collapse in the long run. This shows that the baselines, including high-capacity models MIM and PhyDNet that leverage powerful ConvLTSMs (Shi et al. 2015), have difficulties separating content and motion. Indeed, a model separating correctly content and motion should maintain digits appearance even when it miscalculates their trajectories, like DDPAE which alters only marginally the digits in fig. 6.3. In contrast, ours manages to produce consistent samples even at $t+95$, making it reach state-of-the-art performance. Moreover, we significantly outperform all baselines in the content swap experiment, showing the clear advantage of the proposed PDE-inspired simple model for spatiotemporally disentangled prediction.

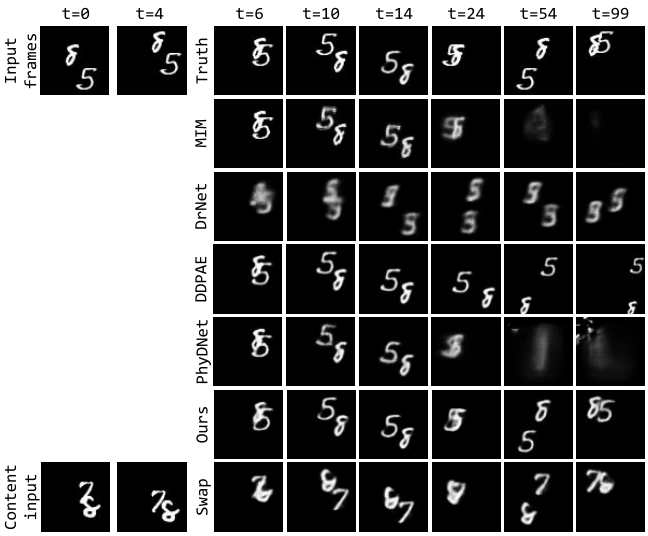


Figure 6.3. – Predictions of compared models on Moving MNIST, and content swap experiment for our model.

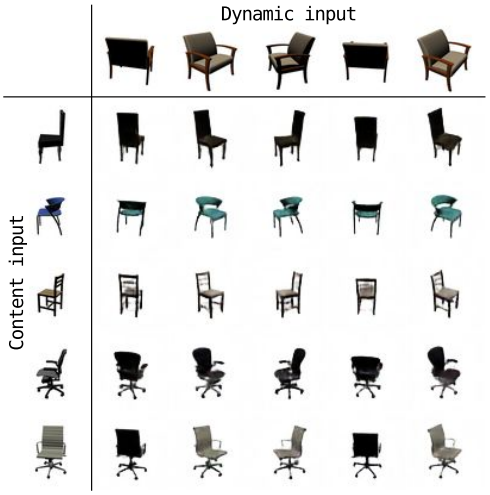


Figure 6.4. – Fusion of content (first column) and dynamic (first row) variables in our model on 3D Warehouse Chairs.

Ablation studies developed in table B.1 confirm that this advantage is due to the constraints motivated by the separation of variables. Indeed, the model without S fails at long-term forecasting, and removing any non-prediction penalty of the training loss substantially harms performances. In particular, the invariance loss on the static component and the regularization of initial condition T_{t_0} are essential, as their absence hinders both prediction and disentanglement. The auto-encoding constraint makes predictions more stable, allowing accurate long-term forecasting and disentanglement. This ablation study also confirms the necessity to constrain the ℓ_2 norm of the dynamic variable (see eq. (6.12)) for the model to disentangle. Comparisons of table 6.2 actually show that enforcing this loss on the first time step only is sufficient to ensure state-of-the-art disentanglement, as advocated in section 6.4.4.

Finally, we assess whether the temporal ODE of eq. (6.7) induced by the separation of variables is advantageous by replacing the dynamic model with a standard GRU RNN (Cho et al. 2014). Results reported in table B.1 show substantially better prediction and disentanglement performance for the original model grounded on the separation of variables, indicating the relevance of our approach.

Table 6.3. – Prediction MSE ($\times 100 \times 32 \times 32 \times 2$) of compared models on TaxiBJ, with best MSE highlighted in bold.

Ours	Ours (without S)	PhyDNet	MIM	E3D	C. LSTM	PredRNN	ConvLSTM
39.5	43.7	41.9	42.9	43.2	44.8	46.4	48.5

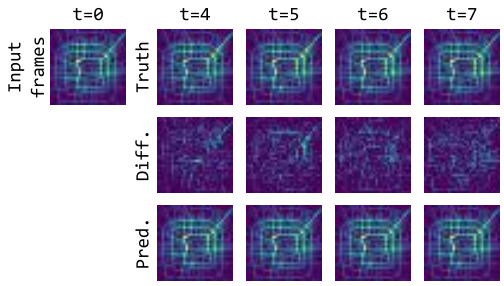


Figure 6.5. – Example of ground truth and prediction of our model on TaxiBJ. The middle row shows the scaled difference between our predictions and the ground truth.

6.5.3 A Multi-View Dataset: 3D Warehouse Chairs

We perform an additional disentanglement experiment on the 3D Warehouse Chairs dataset introduced by (Aubry et al. 2014). This dataset contains 1393 three-dimensional models of chairs seen under various angles. Since all chairs are observed from the same set of angles, this constitutes a multi-view dataset enabling quantitative disentanglement experiments. We create sequences from this dataset for our model by assembling adjacent views of each chair to simulate its rotation from right to left. We then evaluate the disentanglement properties of our model with the same content swap experiments as for Moving MNIST. It is similar to one of (Denton and Birodkar 2017)'s experiments who qualitatively tested their model on a similar, but smaller, multi-view chairs dataset. We achieve 18.70 PSNR and 0.7746 SSIM on this task, outperforming DrNet which only reaches 16.35 PSNR and 0.6992 SSIM. This is corroborated by qualitative experiments in figs. 6.4 and B.7. We highlight that the encoder and decoder architectures of both competitors are identical, validating our PDE-grounded framework for spatiotemporal disentanglement of complex three-dimensional shapes.

6.5.4 A Crowd Flow Dataset: TaxiBJ

We finally study the performance of our spatiotemporal model on the real-world TaxiBJ dataset (Zhang et al. 2017), consisting in taxi traffic flow in Beijing monitored on a 32×32 grid with an observation every thirty minutes. It is highly

structured as the flows are dependent on the infrastructures of the city, and complex since methods have to account for non-local dependencies and model subtle changes in the evolution of the flows. It is a standard benchmark in the spatiotemporal prediction community (Y. Wang et al. 2019b; Le Guen and Thome 2020).

We compare our model in table 6.3 against PhyDNet and MIM, as well as powerful baselines E3D-LSTM (E3D, Y. Wang et al. 2019a), Causal LSTM (C. LSTM, Y. Wang et al. 2018), PredRNN (Y. Wang et al. 2017) and ConvLTSM (Shi et al. 2015), using results reported by (Y. Wang et al. 2019b) and (Le Guen and Thome 2020). An example of prediction is given in fig. 6.5. We observe that we significantly overcome the state of the art on this complex spatiotemporal dataset. This improvement is notably driven by the disentanglement abilities of our model, as we observe in table 6.3 that the alternative version of our model without S achieves results comparable to E3D and worse than PhyDNet and MIM.

6.6 Conclusion

We introduce a novel method for spatiotemporal prediction inspired by the separation of variables PDE resolution technique that induces time invariance and regression penalties only. These constraints ensure the separation of spatial and temporal information. We experimentally demonstrate the benefits of the proposed model, which outperforms prior state-of-the-art methods on physical and synthetic video datasets. We believe that this work, by providing a dynamical interpretation of spatiotemporal disentanglement, could serve as the basis of more complex models further leveraging the PDE formalism. Another direction for future work could be extending the model with more involved tools such as VAEs to improve its performance, or adapt it to the prediction of natural stochastic videos (Denton and Fergus 2018).

Part IV

HYBRID MODELS: COMPLETING PHYSICAL DYNAMICS WITH DEEP LEARNING

Abstract

In this part, we propose two learning frameworks in order to learn hybrid models. These models are made of two components. The first one is a fixed physical-prior with unknown parameters. The second component is learned solely from data. The components are summed and integrated in order to model trajectories matching the data. Two tasks are at stake for this modeling: the correct estimation of the parameters of the prior physical model, and the learning of the data-driven model. The direct learning of the two components is ill-posed. The main pitfall in this task is to learn a DL component that overtakes the physical model. In our first proposition, we constrain the norm of the data-driven component, recovering theoretically the existence and the uniqueness of the decomposition. In our second work, we generalize the previous approach and illustrate how several constraints can be imposed on both the physical model and the data-driven component in order to recover a physically sound decomposition.

The work in this part has led to the publication of two conference papers:

- Yuan Yin, Vincent Le Guen, Jérémie Donà, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari (2021b). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *The Ninth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=kmG8vRXTFv.

- Jérémie Donà, Marie Déchelle, Patrick Gallinari, and Marina Lévy (2022). "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction". In: *The Tenth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=gbe1zHyA73

CHAPTER 7

INTRODUCTION

7.1 Motivation

Combining physical models (denoted MB for model-based) and ML is an emerging trend to develop the interplay between the two paradigms. For example, (Brunton et al. 2016; Long et al. 2018) learn the explicit form of PDE directly from data, (Maziar Raissi et al. 2019b; Sirignano and Spiliopoulos 2018) use NNs as implicit methods for solving PDE, (Seo et al. 2020) learn spatial differences with a graph network, (Ummenhofer et al. 2020) introduce continuous convolutions for fluid simulations, (Bézenac et al. 2018b) learn the velocity field of an advection-diffusion system, (Greydanus et al. 2019; Z. Chen et al. 2020b) enforce conservation laws in the network architecture or in the loss function.

The large majority of aforementioned MB/ML hybrid approaches assume that the physical model adequately describes the observed dynamics. This assumption is, however, commonly violated in practice. This may be due to various factors, e.g. idealized assumptions and difficulty to explain processes from first principles, computational constraints prescribing a fine grain modeling of the system, unknown external factors, forces and sources which are present. Free-form ML models have become a complementary approach to traditional physics based models (MB) (Reichstein et al. 2019; Dueben and Bauer 2018). Both offer advantages: whereas MB approaches generalize and extrapolate better, ML high expressivity approaches benefit from the ongoing explosive growth of available data such as satellite observations, with reduced costs compared to data assimilation.

In chapters 8 and 9, we propose two approaches to leverage prior dynamical ODE/PDE knowledge in the situation where the prior physical model is incomplete, i.e. unable to represent the whole complexity of observed data.

Our first proposition described in chapter 8 illustrated in fig. 8.1 on the pendulum problem, relies on a control of the norm of the data-driven augmentation. In our second work (chapter 9), we aim at providing a general framework for the learning of MB/ML decompositions.

7.2 Elementary Notations

We consider a dynamical system with state at time t denoted $Z_t = Z(t)$. Z_t might be fully or only partially observed: we write $Z_t = (X_t, Y_t)$, where X_t is the observed component and Y_t the unobserved one. The evolution of Z is governed by a differential equation with dynamics :

$$\frac{dZ_t}{dt} = \frac{d}{dt}\begin{pmatrix} X_t \\ Y_t \end{pmatrix} = \begin{pmatrix} f_X(Z_t) \\ f_Y(Z_t) \end{pmatrix} \tag{7.1}$$

The objective is to predict trajectories of X, i.e. to model the evolution of the observable part following $\frac{dX_t}{dt} = f_X(Z_t)$. For simplicity, we omit the index X in f_X and write $f(.) \triangleq f_X(.)$.

Dynamical Hypothesis We assume that the dynamics of the observable part X_t is partially known and writes as:

$$\frac{dX_t}{dt} = f(Z_t) = f_k(Z_t) + f_u(Z_t) \tag{7.2}$$

where $f_k \in \mathcal{H}_k$, accounting for the prior knowledge, is a known operator with unknown parameters θ^*, and f_u is the unknown residual dynamics. $\mathcal{H}_k$ denote a space of functions, for example as illustrated in section 2.4.2. Note that the additive hypothesis in eq. (7.2) is not restrictive and is discussed in appendix D.2.

A critical aspect of learning (f_k, f_u) so that $f = f_k + f_u$, is that the decomposition (f_k, f_u) is in general not unique. More specifically, all the dynamics could be captured by the f_u component. Therefore, learning the decomposition (f_k, f_u) is ill-defined preventing clear interpretability in the learned functions and decreasing generalization performances.

CHAPTER 8

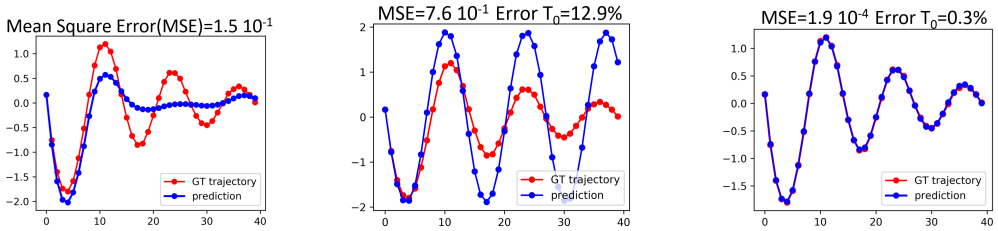
AUGMENTING PHYSICAL MODELS WITH DEEP NETWORKS

Designing a general method for combining MB and ML approaches is still a widely open problem, and a clear problem formulation for the latter is lacking (Reichstein et al. 2019). Our contributions in this section towards these goals are the following:

- We introduce a simple yet principled framework for combining both approaches. We decompose the dynamics into a physical and a data-driven term such that the data-driven component only models information that cannot be captured by the physical model. We provide existence and uniqueness guarantees (section 8.1) for the decomposition given mild conditions, and show that this formulation ensures interpretability and benefits generalization.

- We propose a trajectory-based training formulation along with an adaptive optimization scheme (algorithm 8.1 and section 8.3) enabling end-to-end learning for both the physical and the deep learning components. This allows APHYNITY to *automatically* adjust the complexity of the neural network to different approximation levels of the physical model, paving the way to flexible learned hybrid models.

- We demonstrate the generality of the approach on three use cases (reaction-diffusion, wave equations and the pendulum) representative of different PDE families (parabolic, hyperbolic), having a wide spectrum of application domains, e.g. acoustics, electromagnetism, chemistry, biology, physics (section 8.4). We show that APHYNITY is able to achieve performances close to complete physical models by augmenting incomplete ones, both in terms of forecasting accuracy and physical parameter identification.



(a) Data-driven Neural ODE (b) Simple physical model (c) Our APHYNITY framework

Figure 8.1. – Predicted dynamics for the damped pendulum vs. ground truth (GT) trajectories $\frac{\partial^2\theta}{\partial t^2} + \omega_0^2 \sin\theta + \alpha\frac{\partial\theta}{\partial t} = 0$. We show that in (a) the data-driven approach (R. T. Q. Chen et al. 2018) fails to properly learn the dynamics due to the lack of training data, while in (b) an ideal pendulum cannot take friction into account. The proposed APHYNITY framework shown in (c) augments the simplified physical model in (b) with a data-driven component. APHYNITY improves both forecasting (MSE) and parameter identification (Error T_0) compared to (b).

8.1 Decomposing Dynamics into Physical and Augmented Terms

We set ourselves in a fully observable setting, i.e $X = Z$. As introduced in Section 7.2, we consider the common situation where incomplete information is available on the dynamics, under the form of a family of ODEs or PDEs characterized by their temporal evolution $f_k \in \mathcal{H}_k \subset \mathcal{F}$. The APHYNITY framework leverages the knowledge of $\mathcal{H}_k$ while mitigating the approximations induced by this simplified model through the combination of physical and data-driven components. $\mathcal{F}$ being a vector space, we recall Equation (7.2):

$$f = f_k + f_u$$

The learning of (f_k, f_u) is difficult, indeed, all the dynamics could be captured by the statistical component f_u. This decomposition is thus ill-defined and hampers the interpretability and the extrapolation abilities of the model. In other words, one wants the estimated parameters of f_k to be as close as possible to the true parameter values of the physical model and f_u to play only a complementary role w.r.t f_k, so *as to model only the information that cannot be captured by the physical prior*. For example, when $f \in \mathcal{H}_k$, the data can be fully described by the physical model, and in this case it is sensible to desire f_u to be nullified; this is of central importance in a setting where one wishes to identify physical quantities, and for the model to generalize and extrapolate to new conditions. In a more general

setting where the physical model is incomplete, the action of f_u on the dynamics, as measured through its norm, should be as small as possible.

This general idea is embedded in the following optimization problem:

$$\min_{f_k \in \mathcal{H}_k, f_u \in \mathcal{F}} \|f_u\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{dX_t}{dt} = (f_k + f_u)(X_t) \tag{8.1}$$

The originality of APHYNITY is to leverage model-based prior knowledge by augmenting it with a neurally parametrized dynamics. It does so while ensuring optimal cooperation between the prior model and the augmentation.

A first key question is whether the minimum in eq. (8.1) is indeed well-defined, in other words whether there exists indeed a decomposition with a minimal norm f_u. The answer actually depends on the geometry of f_k, and is formulated in the following proposition proven in appendix C.2:

Proposition 8.1 (Existence of a minimizing pair)**.** *If $\mathcal{H}_k$ is a proximinal set*[1]*, there exists a decomposition minimizing eq.* (8.1).

Proximinality is a mild condition which, as shown through the proof of the proposition, cannot be weakened. It is a property verified by any boundedly compact set. In particular, it is true for closed subsets of finite dimensional spaces. However, if only existence is guaranteed, while forecasts would be expected to be accurate, non-uniqueness of the decomposition would hamper the interpretability of f_k and this would mean that the identified physical parameters are not uniquely determined.

It is then natural to ask under which conditions solving problem eq. (8.1) leads to a unique decomposition into a physical and a data-driven component. The following result provides guarantees on the existence and uniqueness of the decomposition under mild conditions. The proof is given in appendix C.2:

Proposition 8.2 (Uniqueness of the minimizing pair)**.** *If $\mathcal{H}_k$ is a Chebyshev set*[1]*, eq.* (8.1) *admits a unique minimizer. The f_k in this minimizer pair is the metric projection of the unknown f onto $\mathcal{H}_k$.*

The Chebyshev assumption condition is strictly stronger than proximinality but is still quite mild and necessary. Indeed, in practice, many sets of interest are Chebyshev, including all closed convex spaces in strict normed spaces and, if $\mathcal{F} = L^2$, $\mathcal{H}_k$ can be any closed convex set, including all finite dimensional subspaces. In particular, all examples considered in the experiments are Chebyshev sets.

Propositions 8.1 and 8.2 provide, under mild conditions, the theoretical guarantees for the APHYNITY formulation to infer a correct MB/ML decomposition,

1. A proximinal set is one from which every point of the space has at least one nearest point. A Chebyshev set is one from which every point of the space has a unique nearest point. More details in appendix C.1.

thus enabling both recovering the proper physical parameters and accurate forecasting.

8.2 Solving APHYNITY with Deep Neural Networks

In the following, both terms of the decomposition are parametrized and are denoted as $f_k^{\theta_k}$ and $f_u^{\theta_u}$. Solving APHYNITY then consists in estimating the parameters θ_k and θ_u. θ_k are the physical parameters and are typically low-dimensional, e.g. 2 or 3 in our experiments for the considered physical models. For f_u, we need sufficiently expressive models able to optimize over all $\mathcal{F}$: we thus use deep neural networks, which have shown promising performances for the approximation of differential equations (Maziar Raissi et al. 2019b; Ayed et al. 2019).

When learning the parameters of $f_k^{\theta_k}$ and $f_u^{\theta_u}$, we have access to a finite dataset of trajectories discretized with a given temporal resolution Δt: $\mathcal{D}_{\text{train}} = \{(X^{(i)}_{k\Delta t})_{0\leq k\leq \lfloor T/\Delta t \rfloor}\}_{1\leq i\leq N}$. Solving eq. (8.1) requires estimating the state derivative dX_t/dt appearing in the constraint term. One solution is to approximate this derivative using e.g. finite differences as in (Brunton et al. 2016; Greydanus et al. 2019; Cranmer et al. 2020). This numerical scheme requires high space and time resolutions in the observation space in order to get reliable gradient estimates. Furthermore it is often unstable, leading to explosive numerical errors as discussed in appendix C.4. We propose instead to solve eq. (8.1) using an integral trajectory-based approach: we compute $\widetilde{X}^i_{k\Delta t, X_0}$ from an initial state $X_0^{(i)}$ using the current $f_k^{\theta_k} + f_u^{\theta_u}$ dynamics, then enforce the constraint $\widetilde{X}^i_{k\Delta t, X_0} = X^i_{k\Delta t}$. This leads to our final objective function on (θ_k, θ_u):

$$\min_{\theta_k, \theta_u} \quad \left\| f_u^{\theta_u} \right\| \quad \text{subject to} \quad \forall i, \forall k, \widetilde{X}^{(i)}_{k\Delta t} = X^{(i)}_{k\Delta t} \tag{8.2}$$

where $\widetilde{X}^{(i)}_{k\Delta t}$ is the approximate solution of the integral $\int_{X_0^{(i)}}^{X_0^{(i)}+k\Delta t} (f_k^{\theta_k} + f_u^{\theta_u})(X_s) dX_s$ obtained by a differentiable ODE solver.

In our setting, where we consider situations for which $f_k^{\theta_k}$ only partially describes the physical phenomenon, this coupled MB + ML formulation leads to different parameter estimates than using the MB formulation alone, as analyzed more thoroughly in appendix C.3. Interestingly, our experiments show that using this formulation also leads to a better identification of the physical parameters θ_k than when fitting the simplified physical model $f_k^{\theta_k}$ alone (section 8.4). With only an incomplete knowledge on the physics, θ_k estimator will be biased by the additional dynamics which needs to be fitted in the data. Appendix C.6 also confirms

that the integral formulation gives better forecasting results and a more stable behavior than supervising over finite difference approximations of the derivatives.

8.3 Adaptively Constrained Optimization

The formulation in eq. (8.2) involves constraints which are difficult to enforce exactly in practice. We considered a variant of the method of multipliers (Bertsekas 1996) which uses a sequence of Lagrangian relaxations $\mathcal{L}_{\lambda_j}(\theta_k, \theta_u)$:

$$\mathcal{L}_{\lambda_j}(\theta_k, \theta_u) = \|f_u^{\theta_u}\| + \lambda_j \cdot \mathcal{L}_{traj}(\theta_k, \theta_u) \tag{8.3}$$

where $\mathcal{L}_{traj}(\theta_k, \theta_u) = \sum_{i=1}^{N} \sum_{h=1}^{T/\Delta t} \|X_{h\Delta t}^{(i)} - \widetilde{X}_{h\Delta t}^{(i)}\|$. This method needs an increasing sequence $(\lambda_j)_j$ such that the successive minima of $\mathcal{L}_{\lambda_j}$ converge to a solution (at least a local one) of the constrained problem (8.2). We select $(\lambda_j)_j$ by using an iterative strategy: starting from a value λ_0, we iterate, minimizing $\mathcal{L}_{\lambda_j}$ by gradient descent[2], then update λ_j with: $\lambda_{j+1} = \lambda_j + \tau_2 \mathcal{L}_{traj}(\theta_{j+1})$, where τ_2 is a chosen hyper-parameter and $\theta = (\theta_k, \theta_u)$. This procedure is summarized in algorithm 8.1. This adaptive iterative procedure allows us to obtain stable and robust results, in a reproducible fashion, as shown in the experiments.

Algorithm 8.1 APHYNITY

Initialization: $\lambda_0 \geq 0, \tau_1 > 0, \tau_2 > 0$ **for** *epoch* = $1 : N_{epochs}$ **do**
  for *iter in* $1 : N_{iter}$ **do**
    for *batch in* $1 : B$ **do**
      $\theta_{j+1} = \theta_j - \tau_1 \nabla \left[\lambda_j \mathcal{L}_{traj}(\theta_j) + \|f_u\|\right]$
    end
  end
  $\lambda_{j+1} = \lambda_j + \tau_2 \mathcal{L}_{traj}(\theta_{j+1})$
end

8.4 Experimental Validation

We validate our approach on 3 classes of challenging physical dynamics: reaction-diffusion, wave propagation, and the damped pendulum, representative of various application domains such as chemistry, biology or ecology (for reaction-diffusion) and earth physic, acoustic, electromagnetism or even neurobiology (for waves equations). The two first dynamics are described by PDEs and

2. Convergence to a local minimum isn't necessary, a few steps are often sufficient for a successful optimization.

thus in practice should be learned from very high-dimensional vectors, discretized from the original compact domain. This makes the learning much more difficult than from the one-dimensional pendulum case. For each problem, we investigate the cooperation between physical models of increasing complexity encoding incomplete knowledge of the dynamics (denoted *Incomplete physics* in the following) and data-driven models. We show the relevance of APHYNITY (denoted *APHYNITY models*) both in terms of forecasting accuracy and physical parameter identification.

8.4.1 Experimental Setting

We describe the three families of equations studied in the experiments. In all experiments, $\mathcal{F} = \mathcal{L}^2(\mathcal{A})$ where $\mathcal{A}$ is the set of all admissible states for each problem, and the $\mathcal{L}^2$ norm is computed on $\mathcal{D}_{train}$ by: $\|f\|^2 \approx \sum_{i,k} \|f(X^{(i)}_{k\Delta t})\|^2$. All considered sets of physical functionals $\mathcal{H}_k$ are closed and convex in $\mathcal{F}$ and thus are Chebyshev. In order to enable the evaluation on both prediction and parameter identification, all our experiments are conducted on simulated datasets with known model parameters. Each dataset has been simulated using an appropriate high-precision integration scheme for the corresponding equation. All solver-based models take the first state X_0 as input and predict the remaining time-steps by integrating f through the same differentiable generic and common ODE solver (4th order Runge-Kutta) [3]. Implementation details and architectures are given in appendix C.5.

Reaction-diffusion Equations

We consider a 2D FitzHugh-Nagumo type model (Klaasen and Troy 1984). The system is driven by the PDE

$$\frac{\partial u}{\partial t} = a\Delta u + R_u(u, v; k)$$
$$\frac{\partial v}{\partial t} = b\Delta v + R_v(u, v)$$

where a and b are respectively the diffusion coefficients of u and v, Δ is the Laplace operator. The local reaction terms are $R_u(u, v; k) = u - u^3 - k - v$, $R_v(u, v) = u - v$. The state is $X = (u, v)$ and is defined over a compact rectangular domain Ω with periodic boundary conditions. The considered physical models are:

3. This integration scheme is then different from the one used for data generation, the rationale for this choice being that when training a model one does not know how exactly the data has been generated.

- *Param PDE (a, b)*, with unknown (a, b) diffusion terms and without reaction terms:

$$\mathcal{H}_k = \{f_k^{a,b} : (u, v) \mapsto (a\Delta u, b\Delta v) \mid a \geq a_{\min} > 0, b \geq b_{\min} > 0\}$$

- *Param PDE (a, b, k)*, the full PDE with unknown parameters:

$$\mathcal{H}_k = \{f_k^{a,b,k} :(u, v) \mapsto (a\Delta u + R_u(u, v; k), b\Delta v + R_v(u, v) \mid a \geq a_{\min} > 0, b \geq b_{\min} > 0, k \geq k_{\min} > 0\}$$

Damped-wave Equations

We investigate the damped-wave PDE:

$$\frac{\partial^2 w}{\partial t^2} - c^2\Delta w + k\frac{\partial w}{\partial t} = 0$$

where k is the damping coefficient. The state is $X = (w, \frac{\partial w}{\partial t})$ and we consider a compact spatial domain Ω with Neumann homogeneous boundary conditions. Note that this damping differs from the pendulum, as its effect is global. Our physical models are:

- *Param PDE (c)*, without damping term:

$$\mathcal{H}_k = \{f_k^c : (u, v) \mapsto (v, c^2\Delta u) \mid c \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$$

- *Param PDE (c, k)*:

$$\mathcal{H}_k = \{f_k^{c,k} : (u, v) \mapsto (v, c^2\Delta u - kv) \mid c, k \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$$

Damped Pendulum

The motion of a pendulum damped by viscous frictions follows the ODE

$$d^2\theta/dt^2 + \omega_0^2 \sin\theta + \alpha d\theta/dt = 0,$$

where $\theta(t)$ is the angle, ω_0, the proper pulsation (T_0 the period) and α the damping coefficient. With state $X = (\theta, d\theta/dt)$, the ODE is $f_k^{\omega_0,\alpha} : X \mapsto (d\theta/dt, -\omega_0^2 \sin\theta - \alpha d\theta/dt)$. Our physical models are:

- *Hamiltonian* (Greydanus et al. 2019), a conservative approximation, with

$$\mathcal{H}_k = \{f_p^{\mathcal{H}} : (u, v) \mapsto (\partial_y\mathcal{H}(u, v), -\partial_x\mathcal{H}(u, v)) \mid \mathcal{H} \in H^1(\mathbb{R}^2)\},$$

$H^1(\mathbb{R}^2)$ is the first order Sobolev space.

- *Param ODE (ω_0)*, the frictionless pendulum:

$$\mathcal{H}_k = \{f_p^{\omega_0,\alpha=0} |\ \omega_0 \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$$

- *Param ODE (ω_0, α)*, the full pendulum equation:

$$\mathcal{H}_k = \{f_p^{\omega_0,\alpha} |\ \omega_0, \alpha \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$$

Baselines

As purely data-driven baselines, we use Neural ODE (R. T. Q. Chen et al. 2018) for the three problems and PredRNN++ ((Y. Wang et al. 2018), for reaction-diffusion only) which are competitive models for datasets generated by differential equations and for spatio-temporal data. As MB/ML methods, in the ablations studies (see appendix C.6), we compare for all problems, to the vanilla MB/ML cooperation scheme found in (Y. Wang et al. 2019b; Mehta et al. 2020). We also show results for *True PDE/ODE*, which corresponds to the equation for data simulation (which do not lead to zero error due to the difference between simulation and training integration schemes). For the pendulum, we compare to Hamiltonian neural networks (Greydanus et al. 2019; Toth et al. 2020) and to the the deep Galerkin method (DGM, (Sirignano and Spiliopoulos 2018)). See additional details in appendix C.5.

8.4.2 Results

We analyze and discuss below the results obtained for the three kind of dynamics. We successively examine different evaluation or quality criteria. The conclusions are consistent for the three problems, which allows us to highlight clear trends for all of them.

Forecasting Accuracy The data-driven models do not perform well compared to *True PDE/ODE* (all values are test errors expressed as $\log$ MSE): -4.6 for PredRNN++ vs. -9.17 for reaction-diffusion, -2.51 vs. -5.24 for wave equation, and -2.84 vs. -8.44 for the pendulum in table 8.1. The Deep Galerkin method for the pendulum in complete physics *DGM (ω_0, α)*, being constrained by the equation, outperforms Neural ODE but is far inferior to APHYNITY models. In the incomplete physics case, *DGM (ω_0)* fails to compensate for the missing information. The *incomplete physical models*, *Param PDE (a, b)* for the reaction-diffusion, *Param PDE (c)* for the wave equation, and *Param ODE (ω_0)* and *Hamiltonian models* for the

Table 8.1. – Forecasting and identification results on the (a) reaction-diffusion, (b) wave equation, and (c) damped pendulum datasets. We set for (a) $a = 1 \times 10^{-3}, b = 5 \times 10^{-3}, k = 5 \times 10^{-3}$, for (b) $c = 330$, $k = 50$ and for (c) $T_0 = 6$, $\alpha = 0.2$ as true parameters. $\log$ MSEs are computed respectively over 25, 25, and 40 predicted time-steps. %Err param. averages the results when several physical parameters are present. For each level of incorporated physical knowledge, equivalent best results according to a Student t-test are shown in bold. n/a corresponds to non-applicable cases.

Dataset		Method	$\log$ MSE	%Err param.	$\|f_u\|^2$
(a) Reaction-diffusion	Data-driven	Neural ODE	-3.76±0.02	n/a	n/a
		PredRNN++	-4.60±0.01	n/a	n/a
	Incomplete physics	Param PDE (a, b)	-1.26±0.02	67.6	n/a
		APHYNITY Param PDE (a, b)	**-5.10±0.21**	**2.3**	67
	Complete physics	Param PDE (a, b, k)	**-9.34±0.20**	0.17	n/a
		APHYNITY Param PDE (a, b, k)	**-9.35±0.02**	**0.096**	1.5e-6
		True PDE	-8.81±0.05	n/a	n/a
		APHYNITY True PDE	**-9.17±0.02**	n/a	1.4e-7
(b) Wave equation	Data-driven	Neural ODE	-2.51±0.29	n/a	n/a
	Incomplete physics	Param PDE (c)	0.51±0.07	10.4	n/a
		APHYNITY Param PDE (c)	**-4.64±0.25**	**0.31**	71.
	Complete physics	Param PDE (c, k)	-4.68±0.55	1.38	n/a
		APHYNITY Param PDE (c, k)	**-6.09±0.28**	**0.70**	4.54
		True PDE	-4.66±0.30	n/a	n/a
		APHYNITY True PDE	**-5.24±0.45**	n/a	0.14
(c) Damped pendulum	Data-driven	Neural ODE	-2.84±0.70	n/a	n/a
	Incomplete physics	Hamiltonian	-0.35±0.10	n/a	n/a
		APHYNITY Hamiltonian	**-3.97±1.20**	n/a	623
		Param ODE (ω_0)	-0.14±0.10	13.2	n/a
		Deep Galerkin Method (ω_0)	-3.10±0.40	22.1	n/a
		APHYNITY Param ODE (ω_0)	**-7.86±0.60**	**4.0**	132
	Complete physics	Param ODE (ω_0, α)	**-8.28±0.40**	**0.45**	n/a
		Deep Galerkin Method (ω_0, α)	-3.14±0.40	7.1	n/a
		APHYNITY Param ODE (ω_0, α)	**-8.31±0.30**	**0.39**	8.5
		True ODE	**-8.58±0.20**	n/a	n/a
		APHYNITY True ODE	**-8.44±0.20**	n/a	2.3

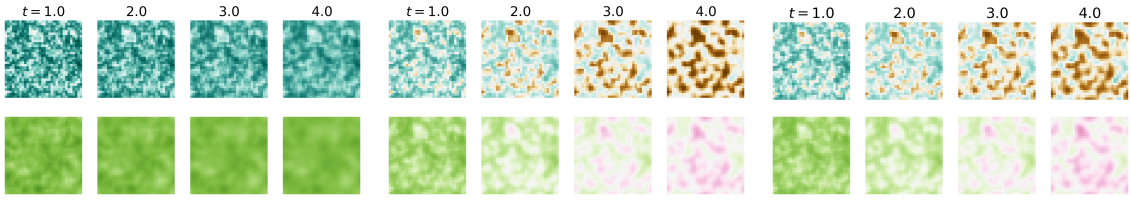
damped pendulum, have even poorer performances than purely data-driven ones, as can be expected since they ignore important dynamical components, e.g. friction in the pendulum case. Using APHYNITY with these imperfect physical models greatly improves forecasting accuracy in all cases, significantly outperforming purely data-driven models, and reaching results often close to the accuracy of the true ODE, when APHYNITY and the true ODE models are integrated with the same numerical scheme (which is different from the one used for data genera-

tion, hence the non-null errors even for the true equations), e.g. -5.92 vs. -5.24 for wave equation in table 8.1. This clearly highlights the capacity of our approach to augment incomplete physical models with a learned data-driven component.

Physical Parameter Estimation Confirming the phenomenon mentioned in the introduction and detailed in appendix C.3, incomplete physical models can lead to bad estimates for the relevant physical parameters: an error respectively up to 67.6% and 10.4% for parameters in the reaction-diffusion and wave equations, and an error of more than 13% for parameters for the pendulum in table 8.1. APHYNITY is able to significantly improve physical parameters identification: 2.3% error for the reaction-diffusion, 0.3% for the wave equation, and 4% for the pendulum. This validates the fact that augmenting a simple physical model to compensate its approximations is not only beneficial for prediction, but also helps to limit errors for parameter identification when dynamical models do not fit data well. This is crucial for interpretability and explainability of the estimates.

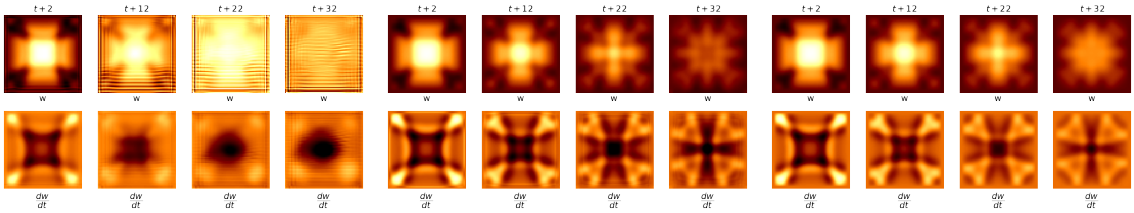
Ablation Study We conduct ablation studies to validate the importance of the APHYNITY augmentation compared to a naive strategy consisting in learning $f = f_k + f_u$ without taking care on the quality of the decomposition, as done in (Y. Wang et al. 2019b; Mehta et al. 2020). Results shown in table 8.1 of appendix C.6 show a consistent gain of APHYNITY for the three use cases and for all physical models: for instance for *Param ODE* (a, b) in reaction-diffusion, both forecasting performances ($\log$ MSE =-5.10 vs. -4.56) and identification parameter (Error= 2.33% vs. 6.39%) improve. Other ablation results are provided in appendix C.6 showing the relevance of the trajectory-based approach described in section 8.2 (vs supervising over finite difference approximations of the derivative f).

Flexibility When applied to complete physical models, APHYNITY does not degrade accuracy, contrary to a vanilla cooperation scheme (see ablations in appendix C.6). This is due to the least action principle of our approach: when the physical knowledge is sufficient for properly predicting the observed dynamics, the model learns to ignore the data-driven augmentation. This is shown by the norm of the trained neural net component f_u, which is reported in table 8.1 last column: as expected, $\|f_u\|^2$ diminishes as the complexity of the corresponding physical model increases, and, relative to incomplete models, the norm becomes very small for complete physical models (for example in the pendulum experiments, we have $\|f_u\| = 8.5$ for the APHYNITY model to be compared with 132 and 623 for the incomplete models). Thus, we see that the norm of f_u is a good indication of how imperfect the physical models $\mathcal{H}_k$ are. It highlights the flexibility of APHYNITY to successfully adapt to very different levels of prior knowledge.



(a) Param PDE (a, b), diffusion-only (b) APHYNITY Param PDE (a, b) (c) Ground truth simulation

Figure 8.2. – Comparison of predictions of two components u (top) and v (bottom) of the reaction-diffusion system. Note that $t = 4$ is largely beyond the dataset horizon ($t = 2.5$).



(a) Neural ODE (b) APHYNITY Param PDE (c) (c) Ground truth simulation

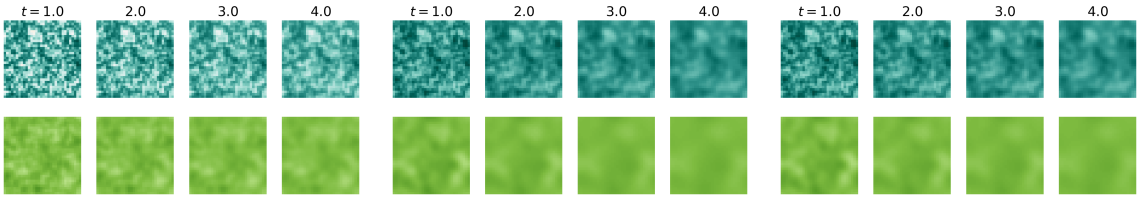
Figure 8.3. – Comparison between the prediction of APHYNITY when c is estimated and Neural ODE for the damped wave equation. Note that $t + 32$, last column for (a, b, c) is already beyond the training time horizon ($t + 25$), showing the consistency of APHYNITY method.

Note also that APHYNITY sometimes slightly improves over the true ODE, as it compensates the error introduced by different numerical integration methods for data simulation and training (see appendix C.5).

Qualitative Visualizations Results in Figure 8.2 for reaction-diffusion show that the incomplete diffusion parametric PDE in Figure 8.2(a) is unable to properly match ground truth simulations: the behavior of the two components in Figure 8.2(a) is reduced to simple independent diffusions due to the lack of interaction terms between u and v. By using APHYNITY in Figure 8.2(b), the correlation between the two components appears together with the formation of Turing patterns, which is very similar to the ground truth. This confirms that f_u can learn the reaction terms and improve prediction quality. In Figure 8.3, we see for the wave equation that the data-driven Neural ODE model fails at approximating dw/dt as the forecast horizon increases: it misses crucial details for the second component dw/dt which makes the forecast diverge from the ground truth. APHYNITY incorporates a Laplacian term as well as the data-driven f_u thus capturing the damping phenomenon and succeeding in maintaining physically sound results for long term forecasts, unlike Neural ODE.

Extension to Non-stationary Dynamics We provide additional results in appendix C.7 to tackle datasets where physical parameters of the equations vary in each sequence. To this end, we design an encoder able to perform parameter estimation for each sequence. Results show that APHYNITY accommodates well to this setting, with similar trends as those reported in this section.

Additional Illustrations We give further visual illustrations to demonstrate how the estimation of parameters in incomplete physical models is improved with APHYNITY. For the reaction-diffusion equation, we show that the incomplete parametric PDE underestimates both diffusion coefficients. The difference is visually recognizable between the poorly estimated diffusion (fig. 8.4(a)) and the true one (fig. 8.4(c)) while APHYNITY gives a fairly good estimation of those diffusion parameters as shown in fig. 8.4(b).



(a) $a = 0.33 \times 10^{-3}, b = 0.94 \times 10^{-3}$, diffusion estimated with Param PDE (a, b)

(b) $a = 0.97 \times 10^{-3}, b = 4.75 \times 10^{-3}$, diffusion estimated with APHYNITY Param PDE (a, b)

(c) $a = 1.0 \times 10^{-3}, b = 5.0 \times 10^{-3}$, true diffusion

Figure 8.4. – Diffusion predictions using coefficient learned with (a) incomplete physical model Param PDE (a, b) and (b) APHYNITY-augmented Param PDE(a, b), compared with the (c) true diffusion

8.5 Conclusion

In this work, we introduce the APHYNITY framework that can efficiently augment approximate physical models with deep data-driven networks, performing similarly to models for which the underlying dynamics are entirely known. We exhibit the superiority of APHYNITY over data-driven, incomplete physics, and state-of-the-art approaches combining ML and MB methods, both in terms of forecasting and parameter identification on three various classes of physical systems. Besides, APHYNITY is flexible enough to adapt to different approximation levels of prior physical knowledge.

CHAPTER 9

CONSTRAINED PHYSICAL-STATISTICS MODELS FOR DYNAMICAL SYSTEM IDENTIFICATION AND PREDICTION

9.1 Introduction

As illustrated in chapter 8, learning a linear MB/ML decomposition with the sole supervision on the system trajectories is ill-posed and admits an infinite number of decompositions. This highlights the need to incorporate physically motivated constraints in the learning of hybrid models, e.g. through regularization penalties. In chapter 8, we have introduced a first principled approach to this problem. Different authors have also introduced ad-hoc schemes to guide the model towards physical solutions (X. Jia et al. 2019; Linial et al. 2021). In this chapter, we introduce an extension of chapter 8 that generalizes several previous attempts in the regularization of hybrid models. This allows us to introduce more diverse and general sets of constraints providing a grounded alternative to previous methods. Our contributions in this chapter are :

- In section 9.3.1, we introduce a novel way to recover well-posedness in the learning of hybrid MB/ML models through the control of an upper bound. We further extend our framework to incorporate auxiliary data when available to handle complex real-world problems.
- In section 9.3.2, we propose an alternate optimization algorithm to learn hybrid models. In section 9.3.3, we provide convergence analysis in a simplified case and experimentally evidence the soundness of our approach on more complex settings including challenging real world problems (section 9.4).

9.2 Background and Problem Setup

Dynamical Hypothesis We assume partial knowledge of the dynamics of the observed X_t:

$$\frac{dX_t}{dt} = f(Z_t) = f_k(Z_t) + f_u(Z_t) \tag{9.1}$$

where $f_k \in \mathcal{H}_k$ is a known operator with unknown parameters θ^*, and $f_u \in \mathcal{H}_u$ is the unknown residual dynamics. $\mathcal{H}_k$ and $\mathcal{H}_u$ denote function spaces.

Learning Problem Our objective is to approximate f with a function h learned from the observed data. According to eq. (9.1), we assume $h = h_k + h_u$. $h_k \in \mathcal{H}_k$ belongs to the same hypothesis space as f_k, i.e. it has the same parametric form. Its parameters are denoted θ_k. Note that $h_k(., \theta^*) = f_k$. $h_u \in \mathcal{H}_u$ is represented by a free form functional with parameters θ_u, e.g. a neural network. The learning problem is then to estimate from data the parameters of h_k so that they match the true physical ones and those of h_u so as to approximate at best the unknown dynamics f. In this regard, an intuitive training objective is to minimize a distance d between $h = h_k + h_u$ and f:

$$\mathrm{d}(h, f) = \mathbb{E}_{Z \sim p_Z} \|h(Z) - f(Z)\|_2, \tag{9.2}$$

where p_Z is the distribution of the state values Z that accounts for varying initial states. Each Z then defines a training sample. Again, minimizing eq. (9.2) with $h = h_k + h_u$ enables to predict accurate trajectories but may have an infinite number of solutions. For instance, h_u may bypass the physical hypothesis h_k. Thus, interpretability is not guaranteed. We develop in the following section a method to overcome this ill-posedness.

9.3 Method

In hybrid modeling, two criteria are essentials: 1. identifiability, i.e. the estimated parameters of h_k should correspond to the true physical ones; 2. prediction power, i.e. the statistical component h_u should complete h_k so that $h = h_k + h_u$ performs accurate prediction over the system states. To control the contribution of each term h_k and h_u, we work upper bounds out of eq. (9.2) (section 9.3.1). We then propose to minimize $\mathrm{d}(h, f)$ while constraining the upper bounds, which provide us with a well-posed learning framework (section 9.3.2). Besides, we show that several previous works that introduced constrained optimization to solve related problems are specific cases of our formulation notably (X. Jia et al. 2019; Linial et al. 2021) and our approach of chapter 8. Finally, we introduce an alternate optimization algorithm which convergence is shown in section 9.3.3 for a linear approximation of f.

9.3.1 Structural Constraints for Dynamical Systems

To ensure identifiability, we derive regularizations on h_k and h_u flowing from the control of an upper bound of $\mathrm{d}(h, f)$. In particular, to minimize $\mathrm{d}(h_k, f_k)$ would enable us to accurately interpret h_k as the true f_k, and thus h_u as the residual dynamics f_u. However, since we do not access the parameters of f_k, computing $\mathrm{d}(h_k, f_k)$ is not tractable. We then consider two possible situations. In the first one, the only available information on the physical system is the parametric form of f_k (or equivalently of h_k), training thus only relies on observed trajectories (eq. (9.3)). In the second one, we consider that auxiliary information about f_k is available and will be used to minimize the distance between h_k and f_k (eq. (9.4)). 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 and better corresponds to real cases as will be shown in the experimental section (see section 9.4.2).

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_k. In this case, to make h_k accountable in our observed phenomena, a solution is to minimize $\mathrm{d}(h_k, f)$. Following the triangle inequality we link up both errors $\mathrm{d}(h, f)$ and $\mathrm{d}(h_k, f)$ (computations available in appendix D.3.1):

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, h_k) + \mathrm{d}(h_k, f) = \mathrm{d}(h_u, 0) + \mathrm{d}(h_k, f) \tag{9.3}$$

We want the physical-statistical model $h = h_k + h_u$ to provide high quality forecasts. Minimizing the sole upper bound does not ensure such aim, as h_u is only penalized through $\mathrm{d}(h_u, 0)$ and is not optimized to contribute to predictions. We thus propose to minimize $\mathrm{d}(h, f)$ while controlling both $\mathrm{d}(h_u, 0)$ and $\mathrm{d}(h_k, f)$. Such a control of the upper bound of eq. (9.3) amounts to balancing the contribution of the ML and the MB components. This will be formally introduced in section 9.3.2.

Link to the Literature The least action principle on the ML component i.e. constraining $\mathrm{d}(h_u, 0)$ is invoked for a geometric argument in chapter 8, and appears as a co-product of the introduction of $\mathrm{d}(h_k, f)$ in eq. (9.3). Optimizing $\mathrm{d}(h_k, f)$ to match the physical model with observations is investigated in (Forssell and Lindskog 1997).

The general approach of eq. (9.3) allows us to perform prediction (via h) and system identification (via h_k) on simple problems (see section 9.4.1). The learning of real-world complex dynamics, via data-driven hybrid models, often fails at yielding a physically sound estimation, as illustrated in section 9.4.2. This suggests that 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_k. Indeed, a common issue in physics is to infer an unobserved variable of interest (in our case f_k parameters $\theta^\star$) from indirect or noisy measurements that we refer to as proxy data. For instance, one can access a physical quantity but only at a coarse resolution, as in (Um et al. 2020; Belbute-Peres et al. 2020) and in the real world example detailed in section 9.4.2. We show in the next subsection how to incorporate such an information in order to approximate $\mathrm{d}(h_k, f_k)$.

Matching the Physical Hypotheses: Introducing Auxiliary Data

We here assume one accesses a proxy of f_k, denoted $f_k^{pr} \in \mathcal{H}_k$. Our goal is to adapt our framework to incorporate such auxiliary information, bringing the regularization induced by f_k^{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. We have:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, h_k) + \mathrm{d}(h_k, f_k^{pr}) + \Gamma = \mathrm{d}(h_u, 0) + \mathrm{d}(h_k, f_k^{pr}) + \Gamma \tag{9.4}$$

where Γ is a constant of the problem that cannot be optimized (see appendix D.3.2). In that context, we can benefit from auxiliary information providing us with coarse estimates of $\theta^\star$, denoted θ^{pr}, such that $f_k^{pr} = h_k(., \theta^{pr}) \approx f_k$. To use the available θ^{pr} to guide our estimation towards the true parameters $\theta^\star$ of f_k, a simple solution is to directly enforce the minimization of $\mathrm{d}(h_k, f_k^{pr})$ in the parameter space by minimizing $\|\theta_k - \theta^{pr}\|_2$, where θ_k are the parameters of h_k. Indeed, because f_k and f_k^{pr} have identical parametric forms (as both belong to the same functional space $\mathcal{H}_k$), minimizing $\|\theta_k - \theta^{pr}\|_2$ will bring h_k closer to f_k^{pr} and thus to f_k. As above, we propose to minimize $\mathrm{d}(h, f)$ while controlling both $\mathrm{d}(h_u, 0)$ and $\mathrm{d}(h_k, f_k^{pr})$, as described in section 9.3.2.

Link to the Literature In (Linial et al. 2021) f_k^{pr} stands for true observations used to constrain a learned latent space, minimizing $\mathrm{d}(h_k, f_k^{pr})$. (X. Jia et al. 2019) use synthetic data as f_k^{pr} to pre-train their model which amounts to the control an upper bound, see appendix D.3.3. Finally, this setting finds an extension, when the model f_k^{pr} is a learned model, for example trained using eq. (9.3), leading to a self-supervision approach described in appendix D.3.4.

9.3.2 Learning Algorithm and Optimization Problem

From the upper bounds, we first recover the well-posedness of the optimization and derive a theoretical learning scheme (section 9.3.2). We then discuss its practical implementation (section 9.3.2).

Well-Posedness and Alternate Optimization Algorithm

Recovering Well-Posedness We reformulate the ill-posed learning of $\min_{h_k,h_u \in \mathcal{H}_k \times \mathcal{H}_u} \mathrm{d}(h,f)$, by instead optimizing $\mathrm{d}(h,f)$ while constraining the upper bounds. Let us define $\mathcal{S}_k$ and $\mathcal{S}_u$ as

$$\mathcal{S}_k = \{\, h_k \in \mathcal{H}_k \mid \ell(h_k) \leq \mu_k \,\} \qquad \mathcal{S}_u = \{\, h_u \in \mathcal{H}_u \mid \mathrm{d}(h_u, 0) \leq \mu_u \,\} \tag{9.5}$$

where μ_k, μ_u are two positive scalars and $\ell(h_k) = \mathrm{d}(h_k, f)$ in the case of section 9.3.1 and $\ell(h_k) = \mathrm{d}(h_k, f_k^{pr})$ in the case of section 9.3.1. Our proposition then amounts to optimizing $\mathrm{d}(h,f)$ over the Minkowski-sum $\mathcal{S}_k + \mathcal{S}_u = \{\, h = h_k + h_u \mid h_k \in \mathcal{S}_k, h_u \in \mathcal{S}_u \,\}$:

$$\min_{h \in \mathcal{S}_k + \mathcal{S}_u} \mathrm{d}(h,f), \tag{9.6}$$

This constrained optimization setting enables us to recover the well-posedness of the optimization problem under the relative compactness of the family of function $\mathcal{H}_k$ (proof in appendix D.4.3).

Proposition 9.1 (Well-posedness)**.** *Under the relative compactness of $\mathcal{S}_k$, eq. (9.6) finds a solution h that writes as $h = h_k + h_u \in \mathcal{S}_k + \mathcal{S}_u$. Moreover, this solution is unique.*

Alternate Optimization Algorithm As the terms in both upper bounds of eqs. (9.3) and (9.4) specifically address either h_k or h_u, we isolate losses relative to h_k and h_u and alternate projections of h_k on $\mathcal{S}_k$ and h_u on $\mathcal{S}_u$, as described in Algorithm 9.1. Said otherwise, we learn h by alternately optimizing h_k (h_u being fixed) and h_u (h_k being fixed). In practice, we rely on a dual formulation (see section 9.3.2 and the SGD version of Algorithm 9.1 in Appendix D.6).

Algorithm 9.1 Alternate estimation: General Setting

Result: Converged h_k and h_u
Set $h_u^0 = 0,\ h_k^0 = \min_{h_k \in \mathcal{H}_k} \mathrm{d}(h_k, f)$, $tol \in \mathbb{R}^+$
while $\mathrm{d}(h,f) > tol$ **do**

$$h_k^{n+1} = \arg\min_{h_k \in \mathcal{S}_k} \mathrm{d}(h_k + h_u^n, f); \quad h_u^{n+1} = \arg\min_{h_u \in \mathcal{S}_u} \mathrm{d}(h_k^{n+1} + h_u, f) \tag{9.7}$$

$n \leftarrow n+1$
end

The convergence of the alternate projections is well studied for the intersection of convex sets or smooth manifolds (Neumann 1950; Lewis and Malick 2008) and has been extended in our setting of Minkowski-sum of convex sets (Lange et al. 2019). Notably, our proposition amounts to the Alternating Direction Method of Multipliers (Boyd et al. 2011). Because d as defined in eq. (9.2) is convex, $\mathcal{S}_u$ and $\mathcal{S}_k$ are convex sets as soon as $\mathcal{H}_k$ and $\mathcal{H}_u$ are convex (Appendix D.1). Thus, if $\mathrm{d}(., f)$ is strongly convex, eq. (9.7) finds one and only one solution (Boyd et al. 2004). However, neither the convexity of $\mathcal{H}_u$ nor of $\mathcal{H}_k$ is practically ensured. Nonetheless, we recover the well-posedness of eq. (9.6) and show the convergence of Algorithm 9.1 in the simplified case where h is an affine function of X_t (see section 9.3.3). For complex PDE where convexity may not hold, we validate our approach experimentally and we evidence in section 9.4 that this formulation enables us to recover both an interpretable decomposition $h = h_k + h_u$ and improved prediction and identification performances.

Practical Optimization

Equation (9.5) involves the choice of μ_k and μ_u. In practice, we implement the projection algorithm by descending gradients on the parameters of h_k and h_u, with respect to the following losses:

$$\mathcal{L}_k(h_k) = \lambda_h \,\mathrm{d}(h, f) + \lambda_{h_k} \ell(h_k) \qquad \mathcal{L}_u(h_u) = \lambda_h \,\mathrm{d}(h, f) + \lambda_{h_u} \,\mathrm{d}(h_u, 0) \tag{9.8}$$

where $\lambda_h, \lambda_{h_k}, \lambda_{h_u}$ are positive real values, dynamically increased/decreased during training. Indeed, $\mathrm{d}(h_u, 0)$ can be interpreted as a *stability loss*, preventing the neural networks to trump the physical component. On the other hand, $\mathrm{d}(h_k, f)$ can be interpreted has an *initialization loss* yield a first estimate of θ_k explaining the dynamics.

Yet, f being unknown: $\mathrm{d}(h, f)$ is not tractable. To estimate $\mathrm{d}(h, f)$, 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, $\mathrm{d}_\phi(\phi_h, \phi_f)$, over all initial conditions X_0:

$$\mathrm{d}_\phi(\phi_h, \phi_f) = \mathbb{E}_{X_0} \int_{t_0}^{t} \left\| \phi_h(\tau, X_0) - \phi_f(\tau, X_0) \right\|_2 d\tau \tag{9.9}$$

We have: $\mathrm{d}_\phi(\phi_h, \phi_f) = 0 \Leftrightarrow \mathrm{d}(h, f) = 0$. Definitions of flows for ODE and in depth consideration on these distances are available in appendix D.1. The gradients of $\mathrm{d}_\phi(\phi_h, \phi_f)$ with respect to the parameters of h_k or h_u 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 the backpropagation, see (Onken and Ruthotto 2020b). To compute eq. (9.9), 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. (9.1) 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 to $\mathrm{d}_\phi(\phi_h, \phi_f)$. Let `ODESolve` be the function integrating any arbitrary initial state x_{t_0} up to time t with dynamics h, so that $x_t = \texttt{ODESolve}(x_{t_0}, h, t)$. The estimate of $\mathrm{d}_\phi(\phi_h, \phi_f)$ then writes as:

$$\mathrm{d}_\phi(\phi_h, \phi_f) \approx \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{N} \left\| \texttt{ODEsolve}(X^i_{t_0}, h, t_j) - X^i_{t_j} \right\|_2$$

Note that the way to compute `ODEsolve` differs across the experiments (see section 9.4).

9.3.3 Theoretical Analysis for a Linear Approximation

We investigate the validity of our proposition when approximating an unknown derivative with an affine function (interpretable first guess approximators). We here consider h_k as a linear function. We do not assume any information on f, thus relieving this section from the need of an accurate prior knowledge f_k. In this context, we show the convergence of the learning scheme introduced in Algorithm 9.1 with $\ell = \mathrm{d}(h_k, f)$, hence 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 9.4. All proofs of this section are conducted using the distance d_ϕ. Let X^s be the unique solution to the initial value problem:

$$\frac{dX_t}{dt} = f(X_t) \quad \text{with} \quad X_{t=0} = X_0 \tag{9.10}$$

With $h_k(X) = AX$ and $h_u(X) = D_A$, the affine approximation of f writes as:

$$\frac{dX_t}{dt} = AX_t + D_A \quad \text{with} \quad X_{t=0} = X_0 \tag{9.11}$$

where $A \in \mathcal{M}_{p,p}(\mathbb{R})$, $D_A \in \mathbb{R}^p$. We write X^D the solution to eq. (9.11) and X^A the solution to eq. (9.11) when $D_A = 0$. The alternate projection algorithm with the distance d_ϕ writes as:

$$\hat{A} = \arg\min_A \int_{t_0}^{t} \left\| X^s(\tau) - X^D(\tau) \right\|_2 d\tau + \lambda_A \int_{t_0}^{t} \left\| X^s(\tau) - X^A(\tau) \right\|_2 d\tau \tag{9.12}$$

$$\hat{D}_A = \arg\min_{D_A} \int_{t_0}^{t} \left\| X^s(\tau) - X^D(\tau) \right\|_2 d\tau + \lambda_D \|D_A\|_2 \tag{9.13}$$

where $\lambda_D, \lambda_A > 0$. As the optimization of eq. (9.12) 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 D.4.2. Such strategy is common practice in system identification. Generic theoretical considerations on existence and uniqueness of solutions to eqs. (9.12) and (9.13) are hard to retrieve. Nonetheless, if A is an invertible matrix, we prove in appendix D.4.4:

Proposition 9.2 (Existence and Uniqueness)**.** *If $\hat{A}$ is invertible, There exists a unique D_A, hence a unique X^D, solving eq.* (9.13)*.*

Finally, formulating Algorithm 9.1 as a least square problem in an affine setting (see appendix D.4.5), we prove the convergence of the alternate projection algorithm (appendix D.4.6) :

Proposition 9.3. *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.* (9.12) *and* (9.13) *converges.*

9.4 Experiments

We validate Algorithm 9.1 on datasets of increasing difficulty (see appendix D.5), where the system state is either fully or partially observed (resp. section 9.4.1 and section 9.4.2). We no longer rely on an affine prior and explicit h_k and h_u for each dataset. Performances are evaluated via standard metrics: MSE (lower is better) and relative Mean Absolute Error (rMAE, lower is better). We assess the relevance of our proposition based on eqs. (9.3) and (9.4), against NeuralODE (R. T. Q. Chen et al. 2018) and state of the art Aphynity (Yin et al. 2021b) and ablation studies. We denote Ours eq. (9.3) (resp. Ours eq. (9.4)) the results when $\ell = \mathrm{d}(h_k, f)$ i.e eq. (9.3), (resp. $\ell = \mathrm{d}(h_k, f_k^{pr})$ i.e. eq. (9.4)). When $\mathrm{d}(h_k, f)$ (resp. $\mathrm{d}(h_u, 0)$) is not considered in the optimization, we refer to the results as $\mathrm{d}(h, f) + \mathrm{d}(h_u, 0)$ (resp. $\mathrm{d}(h, f) + \mathrm{d}(h_k, f)$). When h is trained by only minimizing the discrepancy between actual and predicted trajectories the results are denoted «Only $\mathrm{d}(h, f)$». We report between brackets the standard deviation of the metrics over 5 runs and refer to Appendices D.6 and D.7 for training information and additional results.

9.4.1 Fully Observable Dynamics

To illustrate the learning scheme induced by eq. (9.3), we focus on fully observed low dimensional dynamics: a simple example emerging from Newtonian mechanics and a population dynamics model.

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 et al. 2019; Yin et al. 2021b). Newtonian mechanics provide an ODE describing the evolution of the angle x of the pendulum:

$$\ddot{x} - g/L \sin(x) + k\dot{x} = 0 \tag{9.14}$$

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:

$$\dot{x} = \alpha x - \beta xy, \quad \text{and} \quad \dot{y} = -\gamma y + \delta xy \tag{9.15}$$

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. Further experimental details are available in appendices D.5.1, D.5.2 and D.6.

Identification and Prediction Results Table 9.1 shows that despite accurate trajectory forecasting, the unconstrained setting «Only $\mathrm{d}(h, f)$» fails at estimating the models parameters, showing the need for regularization for identification.

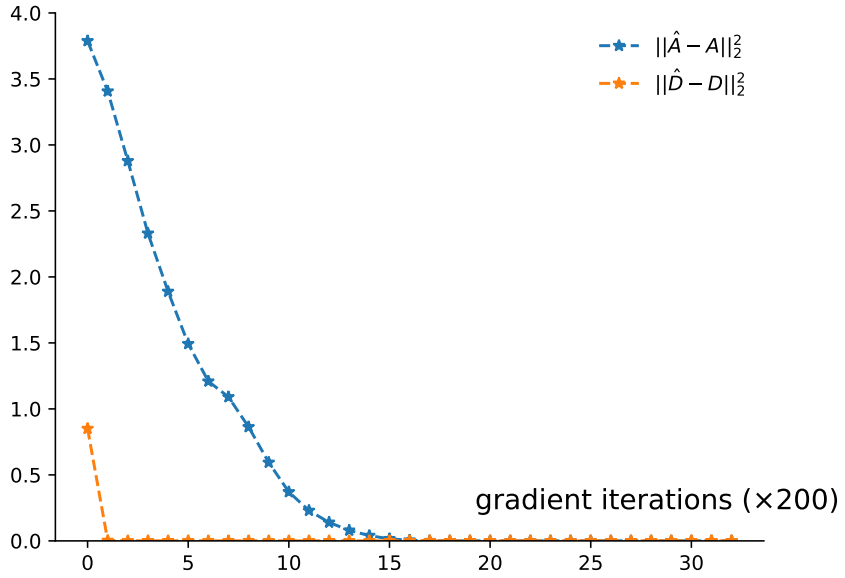


Figure 9.1. – Affine Case : Evolution of the MSE between estimated dynamics $(\hat{A}, \hat{D})$ and the true one (A, D) with the number of gradients steps for linearized DPL.

Constraining the norm of the ML component can be insufficient: for LV data, both Aphynity and $\mathrm{d}(h, f) + \mathrm{d}(h_u, 0)$ do not accurately estimate the model parameters. However, the control of $\mathrm{d}(h_k, f)$, following eq. (9.3), 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. (9.3) 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 $\mathrm{d}(h, f)$» in Table 9.1), h_u learns to corrects the inaccurate h_k. Table 9.1 and figs. D.2 and D.3 (appendix D.7.1) 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. (9.3) increase interpretability and maintain prediction performances.

Throwback to the Affine Case We verify the convergence proved in section 9.3.3 using the damped pendulum (eq. (9.14)) linearized in the small oscillations regime (see appendix D.5.1). Making an affine hypothesis following eq. (9.11), we apply our alternate projection algorithm and optimize A and D_A alternately using SGD. Figure 9.1 shows that we are able to accurately estimate

Table 9.1. – 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. (9.3)	**1.56 (0.009)**	**-13.7 (0.84)**	**7.80 (0.011)**	-9.28 (0.75)
Only $\mathrm{d}(h, f)$	9.35 (0.04)	-13.3 (0.65)	24.5 (0.017)	-9.21 (0.91)
$\mathrm{d}(h, f) + \mathrm{d}(h_k, f)$	1.82 (0.01)	-13.4 (0.56)	7.91 (0.02)	-9.01 (0.99)
$\mathrm{d}(h, f) + \mathrm{d}(h_u, 0)$	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)**

A and D using our proposition, recovering both the oscillation pulsation and the damping coefficient.

9.4.2 High Dimensional Dynamics

We now address the learning of transport equations, describing a wide range of physical phenomena such as chemical concentration, fluid dynamics or material properties. We evaluate the learning setting induced by eq. (9.3) and (9.4) on two physical datasets depicting the evolution of the temperature T advected by a time-dependent velocity field U and subject to forcing S, following:

$$\frac{\partial T}{\partial t} + \nabla.(TU) = S(U) \tag{9.16}$$

The system state $Z = (T, U, S)$ is partially observed, we only access 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 S_t are all of size 64×64.

Experimental Setting We consider as physical prior the advection i.e $h_k(T, \theta_k) = -\nabla.(T\theta_k)$. Thus, θ_k is time-dependent, as we learn it to approximate $\theta^\star = U$. We identify the velocity field θ_k from observations of T, learning a mapping between T and U parameterized by a neural network G_ψ, so that $\theta_k = G_\psi(T_{t-l}, ..., T_t) \approx U_t$, which is common practice in oceanography (Bérériat and Herlin 2015). G_ψ is optimized following eq. (9.8). S remains to be learned by h_u. h_k implements a differentiable semi-Lagrangian scheme (Jaderberg et al. 2015) (see appendix D.5.3) and h_u is a ResNet. G_ψ is a UNet. Training details and a schema of our model are to be found in appendix D.6.

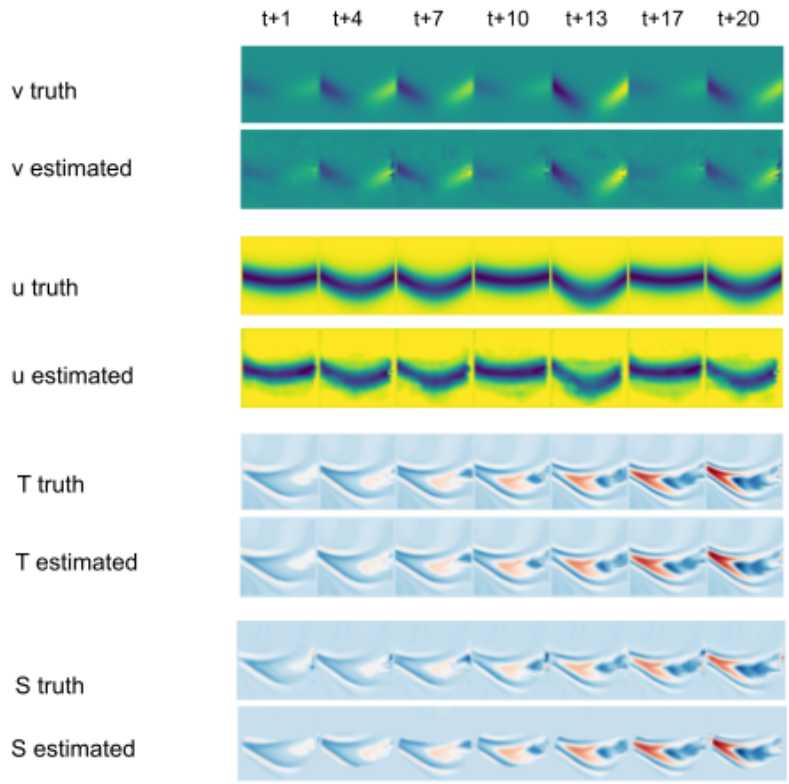


Figure 9.2. – *Best viewed in color*. Estimations of S, T and $U = (u, v)$ on Adv+S. Prediction ranges from 1 to 20 half-days.

Synthetic Advection and Source (Adv+S) To test the applicability of the learning setting induced by eq. (9.3) on partially observed settings, we first study a synthetic setting (denoted Adv+S) of eq. (9.16) by generating velocity fields U, simulated following (Boffetta et al. 2001) and adding a source term S inspired by (Frankignoul 1985). The simulation details are given in appendix D.5.3.

Real Ocean Dynamics (Natl) We consider a complex dataset emulating real world observations of the North ATLantic ocean (denoted Natl) (Ajayi et al. 2019). Modeling the evolution of T in Natl is particularly challenging as its dynamics is chaotic and highly non-linear. This simulation is representative of the complexity encountered in real world data. The principled approach corresponding to eq. (9.3) is insufficient here and one must resort to additional physical information. To illustrate the extension developed in section 9.4.2, we make use of available auxiliary data: measurements from satellite observations provide a coarse estimate of the surface ocean currents velocity fields (see appendix D.5). Then, the learning goal is to refine the approximated velocity fields to fit the ocean dynamics. Therefore we proceed as described in eq. (9.4) and enforce $\mathrm{d}(h_k, f_k^{pr})$ by supervising G_ψ with the proxy data (see appendix D.5.3).

Table 9.2. – Results for Adv+S and Natl data. We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the source term S over 6 time steps on test set.

Models	Adv+S			Natl		
	T	U	S	T	U	S
Ours eq. (9.3)	**0.74 (0.05)**	**1.99 (0.13)**	**0.17 (0.01)**	8.27 (0.06)	11.72 (0.07)	6.01 (0.08)
Ours eq. (9.4)	–	–	–	**6.86 (0.12)**	**6.81 (0.07)**	**4.35 (0.11)**
Aphynity	0.85 (0.35)	3.07 (0.74)	0.18 (0.05)	8.18 (0.16)	11.75 (0.49)	6.02 (0.02)
NODE	1.35 (0.02)	–	–	8.83 (0.98)	–	–

Identification and Prediction Results Table 9.2 indicates that for Adv+S dataset, we estimate accurately the unobserved velocity fields. Qualitatively, Figure 9.2 shows that controlling our proposed upper bound eq. (9.3) facilitates the recovery of truthful velocity fields U along with an accurate prediction of T. For the highly complex Natl, Table 9.2 shows that the introduction of auxiliary data following the formulation in eq. (9.4) significantly helps identification, as the dynamics is too complex to be able to recover physically interpretable velocity fields using the bound of eq. (9.3).
Regarding prediction performances on the Adv+S data, Table 9.2 shows that thanks to our truthful estimates of U, our model provides more precise prediction than NODE and Aphynity. For real world data, thanks to the proxy data our model recovers better velocity fields terms while providing a better estimate for T. Besides, adding prior knowledge in the prediction systems improves prediction performances: appendix D.7 shows that NODE minimizes $\mathrm{d}(h, f)$ by predicting average and blurred frames. This shows the need for regularization when learning on structured physical data.

Ablation Study We present in Table 9.3 an ablation study on the Adv+S dataset evidencing the influence of our learning choices on the resolution of both identification and prediction tasks (see appendix D.7 for detailed results). "Joint" rows of Table 9.3 indicate that the learning of h_u and h_k is done simultaneously. As shown in Table 9.3, the sole optimization of $\mathrm{d}(h, f)$ fails at estimating physically sounded U. This evidences the ill-posedness in such unconstrained optimization. Table 9.3 indicates that all introduced regularizations improve the recovery of U w.r.t. the «Only $\mathrm{d}(h, f)$» baseline, while adding $\mathrm{d}(h_u, 0)$ 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_k and h_u. Notably, our proposition to optimize h_k and h_u alternately beats all baselines on both T prediction and U identification (Table 9.3, Joint rows). Finally,

jointly trained models fail at estimating U in Table 9.3, forcing h_u to capture the whole dynamics.

Table 9.3. – Ablation Study on Adv+S. We report the MSE ($\times$ 100) on the predicted observations T, the velocity fields U and the source term S over 6 time steps. "Joint" rows refer to the simultaneous optim. of h_k and h_u.

Training	Models	T	U	S
	Ours (U known)	0.52	n/a	0.19
Alternate	Ours eq. (9.3)	**0.74 (0.05)**	**1.99 (0.13)**	**0.17 (0.01)**
	Only $\mathrm{d}(h, f)$	1.02 (0.16)	4.08 (0.23)	0.19 (0.06)
	$\mathrm{d}(h, f) + \mathrm{d}(h_k, f)$	1.02 (0.09)	3.66 (0.15)	0.19 (0.03)
	$\mathrm{d}(h, f) + \mathrm{d}(h_u, 0)$	0.77 (0.06)	2.38 (0.17)	0.19 (0.01)
Joint	Ours eq. (9.3)	1.44 (0.08)	3.30 (0.18)	0.30 (0.03)
	Only $\mathrm{d}(h, f)$	1.38 (0.19)	6.96 (0.21)	0.39 (0.08)

9.5 Discussion

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. Besides theoretical considerations, we empirically evidence the soundness of our approach thanks to ablation studies and comparison with recent baselines on several low and high dimensional datasets. This work can see several extensions, e.g. considering non uniform 3-D grid for climate models, further considerations on the investigated upper bounds, or less restrictive decomposition hypothesis.

Part V

DEEP LEARNING FOR DYNAMICAL SYSTEM THAT GENERALIZES

CHAPTER 10

GENERALIZING TO NEW PHYSICAL SYSTEMS VIA CONTEXT-INFORMED DYNAMICS MODEL

Chapter abstract

Data-driven approaches to modeling physical systems fail to generalize to unseen systems that share the same general dynamics with the learning domain, but correspond to different physical contexts. We propose a new framework for this key problem, context-informed dynamics adaptation (CoDA), which takes into account the distributional shift across systems for fast and efficient adaptation to new dynamics. CoDA leverages multiple environments, each associated to a different dynamic, and learns to condition the dynamics model on contextual parameters, specific to each environment. The conditioning is performed via a hypernetwork, learned jointly with a context vector from observed data. The proposed formulation constrains the search hypothesis space to foster fast adaptation and better generalization across environments. It extends the expressivity of existing methods. We theoretically motivate our approach and show state-of-the-art generalization results on a set of nonlinear dynamics, representative of a variety of application domains. We also show, on these systems, that new system parameters can be inferred from context vectors with minimal supervision. This work is under review and available at:

Matthieu Kirchmeyer, Yuan Yin, Jérémie Donà, Nicolas Baskiotis, Alain Rakotomamonjy, and Patrick Gallinari (Jan. 2022). "Generalizing to New Physical Systems via Context-Informed Dynamics Model". working paper or preprint. URL: https://hal.archives-ouvertes.fr/hal-03547546.

10.1 Introduction

Neural Network (NN) approaches to modeling dynamical systems have recently raised the interest of several communities leading to an increasing number of contributions. This topic was explored in several domains, ranging from simple dynamics e.g. Hamiltonian systems (Greydanus et al. 2019; Z. Chen et al. 2020c) to more complex settings e.g. fluid dynamics (Kochkov et al. 2021; Zongyi Li

et al. 2021; Wandel et al. 2021b), earth system science and climate (Reichstein et al. 2019), or health (Fresca et al. 2020). NN emulators are attractive as they may for example provide fast and low cost approximations to complex numerical simulations (Duraisamy et al. 2019; Kochkov et al. 2021), complement existing simulation models when the physical law is partially known (Yin et al. 2021b) or even offer solutions when classical solvers fail e.g. with very high number of variables (Sirignano and Spiliopoulos 2018).

A model of a real-world dynamical system should account for a wide range of contexts resulting from different external forces, spatio-temporal conditions, boundary conditions, sensors characteristics or system parameters. These contexts characterize the dynamics phenomenon. For instance, in cardiac electrophysiology (Neic et al. 2017; Fresca et al. 2020), each patient has its own specificities and represents a particular context. In the study of epidemics' diffusion (Shaier et al. 2021), computational models should handle a variety of spatial, temporal or even sociological contexts. The same holds for most physical problems, e.g. forecasting of spatial-location-dependent dynamics in climate (de Bézenac et al. 2018c), fluid dynamics prediction under distinct external forces (Zongyi Li et al. 2021), *etc*.

The physics approach for modeling dynamical systems relies on a strong prior knowledge about the underlying phenomenon. This provides a causal mechanism which is embedded in a physical dynamics model, usually a system of differential equations, and allows the physical model to handle a whole set of contexts. Moreover, it is often possible to adapt the model to new or evolving situations, e.g. via data assimilation (Kalman 1960; Courtier et al. 1994).

On the other hand, Expected Risk Minimization (ERM) based machine learning (ML) fails to generalize to unseen dynamics. Indeed, it requires i.i.d. data for training and inference while dynamical observations are non-i.i.d. as the distributions change with initial conditions or physical contexts. Thus any ML framework that handles this question should consider other assumptions. A common one used e.g. in domain generalization (J. Wang et al. 2021), states that data come from several environments a.k.a. domains, each with a different distribution. Training is performed on a sample of the environments and test corresponds to new ones. Domain generalization methods attempt to capture problem invariants via a unique model, assuming that there exists a representation space suitable for all the environments. This might be appropriate for classification, but not for dynamical systems where the underlying dynamics differs for each environment. For this problem, we need to learn a function that adapts to each environment, based on a few observations, instead of learning a single domain-invariant function. This is the objective of meta-learning (Thrun and Pratt 1998), a general framework for fast adaptation to unknown contexts. The standard gradient-based methods (e.g. (Finn et al. 2017)) are unsuitable for complex dynamics due to their bi-level

optimization process and are known to overfit when little data is available for adaptation, as in the few-shot learning setting explored in this paper (Mishra et al. 2018). Like invariant methods, meta-learning usually handles basic tasks like classification or regression on static data or simple sequences. Generalization for modeling real-world dynamical systems is a recent topic. Simple simulated dynamics were considered in Reinforcement Learning (Kimin Lee et al. 2020; Clavera et al. 2019) while physical dynamics were modeled in recent works (Yin et al. 2021a; R. Wang et al. 2021). These approaches consider either simplified settings or additional hypotheses e.g. prior knowledge and do not offer general solutions to our adaptation problem (details in Section 10.6).

We propose a new ML framework for generalization in dynamical systems, called **Co**ntext-Informed **D**ynamics **A**daptation (CoDA). Like in domain generalization, we assume availability of several environments, each with its own specificity, yet sharing some physical properties. Training is performed on a sample of the environments. At test time, we assume access to example data from a new environment, here a trajectory. Our goal is to adapt to the new environment distribution with this trajectory. More precisely, CoDA assumes that the underlying system is described by a parametrized differential equation, either an ODE or a PDE. The environments share the parametrized form of the equation but differ by the values of the parameters or initial conditions. CoDA conditions the dynamics model on learned environment characteristics a.k.a. contexts and generalizes to new environments and trajectories with few data. Our main contributions are the following:

- We introduce a multi-environment formulation of the generalization problem for dynamical systems.
- We propose a novel context-informed framework, CoDA, to this problem. It conditions the dynamics model on context vectors via a hypernetwork. CoDA introduces a locality and a low-rank constraint, which enable fast and efficient adaptation with few data.
- We analyze theoretically the validity of our low-rank adaptation setting for modeling dynamical systems.
- We evaluate two variations of CoDA on several ODEs/PDEs representative of a variety of application domains, e.g. chemistry, biology, physics. CoDA achieves SOTA generalization results on in-domain and one-shot adaptation scenarios. We also illustrate how, with minimal supervision, CoDA infers accurately new system parameters from learned contexts.

The paper is organized as follows. In Section 10.2, we present our multi-environment problem. In Section 10.3, we introduce the CoDA framework. In

Section 10.4, we detail how to implement our framework. In Section 10.5, we present our experimental results. In Section 10.6, we present related work.

10.2 Generalization for Dynamical Systems

We present our generalization problem for dynamical systems, then introduce our multi-environment formalization.

10.2.1 Problem setting

We consider dynamical systems that are driven by unknown temporal differential equations of the form:

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = f(X(t)), \tag{10.1}$$

where $t \in \mathbb{R}$ is a time index, $X(t)$ is a time-dependent state in a space $\mathcal{X}$ and $f : \mathcal{X} \to T\mathcal{X}$ a function that maps $X(t) \in \mathcal{X}$ to its temporal derivatives in the tangent space $T\mathcal{X}$. f belongs to a class of vector fields $\mathcal{F}$. $\mathcal{X} \subseteq \mathbb{R}^d$ ($d \in \mathbb{N}^\star$) for ODEs or $\mathcal{X}$ is a space of functions defined over a spatial domain (e.g. 2D or 3D Euclidean space) for PDEs.

Functions $f \in \mathcal{F}$ define a space $\mathcal{D}^f(\mathcal{X})$ of state trajectories $X : I \to \mathcal{X}$, mapping t in an interval I including 0, to the state $X(t) \in \mathcal{X}$. Trajectories are defined by the initial condition $X(0) \triangleq X_0 \sim p(X_0)$ and take the form:

$$\forall t \in I, X(t) = X_0 + \int_0^t f(X(\tau))\mathrm{d}\tau \in \mathcal{X} \tag{10.2}$$

In the following, we assume that $f \in \mathcal{F}$ is parametrized by some unknown attributes e.g. physical parameters, external forcing terms which affect the trajectories.

10.2.2 Multi-environment Learning Problem

We propose to learn the class of functions $\mathcal{F}$ with a data-driven *dynamics model* g_θ parametrized by $\theta \in \mathbb{R}^{d_\theta}$. Given $f \in \mathcal{F}$, we observe N trajectories in $\mathcal{D}^f(\mathcal{X})$ with the form in Equation (10.2). The standard ERM objective considers that all trajectories are i.i.d. Here, we propose a multi-environment learning formulation which considers that observed trajectories of f form an environment $e \in \mathcal{E}$. We denote f^e and $\mathcal{D}^e \triangleq \{\mathcal{D}_i^e\}_{i=1}^N$ the corresponding function and set of trajectories.

We assume that we observe a set of known functions in training environments $\mathcal{E}_{\text{tr}}, \{f^e\}_{e\in\mathcal{E}_{\text{tr}}}$. The goal is to learn g_θ that adapts easily and efficiently to new environments $\mathcal{E}_{\text{ad}}$, corresponding to unseen functions $\{f^e\}_{e\in\mathcal{E}_{\text{ad}}}$ ("ad" stands for adaptation). We define $\forall e \in \mathcal{E}$ the corresponding Mean Squared Error (MSE) loss, over $\mathcal{D}^e$ as

$$\mathcal{L}(\theta, \mathcal{D}^e) \triangleq \sum_{i=1}^{N} \int_{t\in I} \|f^e(X^{e,i}(t)) - g_\theta(X^{e,i}(t))\|_2^2 \mathrm{d}t \tag{10.3}$$

In practice, f^e is unavailable and we can only approximate it from discretized trajectories. We detail later in Equation (10.10) our approximation method based on an integral formulation. It fits observed trajectories directly in state space.

10.3 The CoDA Learning Framework

We introduce CoDA, a new context-informed framework for learning dynamics on multiple environments. It relies on a general adaptation rule (Section 10.3.1) and introduces two key properties: locality, enforced in the objective (Section 10.3.2) and low-rank adaptation, enforced in the proposed model via hypernetwork-decoding (Section 10.3.3). The validity of this framework for dynamical systems is analyzed in Section 10.3.4 and its benefits are discussed in Section 10.3.5.

10.3.1 Adaptation Rule

The dynamics model g_θ should adapt to new environments. Hence, we propose to condition g_θ on observed trajectories $\mathcal{D}^e, \forall e \in \mathcal{E}$. Conditioning is performed via an *adaptation network* A_π, parametrized by π, which adapts the weights of g_θ to an environment $e \in \mathcal{E}$ according to

$$\theta^e \triangleq A_\pi(\mathcal{D}^e) \triangleq \theta^c + \delta\theta^e, \quad \pi \triangleq \{\theta^c, \{\delta\theta^e\}_{e\in\mathcal{E}}\} \tag{10.4}$$

$\theta^c \in \mathbb{R}^{d_\theta}$ are shared parameters, used as an initial value for fast adaptation to new environments. $\delta\theta^e \in \mathbb{R}^{d_\theta}$ are environment-specific parameters conditioned on $\mathcal{D}^e$.

10.3.2 Constrained Optimization Problem

Given the adaptation rule in Equation (10.4), we introduce a constrained optimization problem which learns parameters π such that $\forall e \in \mathcal{E}, \delta\theta^e$ is small and g fits observed trajectories. It introduces a locality constraint with a norm $\|\cdot\|$:

$$\min_{\pi} \sum_{e \in \mathcal{E}} \|\delta\theta^e\|^2 \text{ s.t. } \forall X^e(t) \in \mathcal{D}^e, \frac{\mathrm{d}X^e(t)}{\mathrm{d}t} = g_{\theta^c + \delta\theta^e}(X^e(t))$$

We consider an approximation of this problem which relaxes the equality constraint with the MSE loss $\mathcal{L}$ in Equation (10.3).

$$\min_{\pi} \sum_{e \in \mathcal{E}} \left(\mathcal{L}(\theta^c + \delta\theta^e, \mathcal{D}^e) + \lambda \|\delta\theta^e\|^2 \right) \tag{10.5}$$

λ is a hyperparameter. For training, we minimize Equation (10.5) w.r.t. π over training environments $\mathcal{E}_{\text{tr}}$. After training, θ^c is freezed. For adaptation, we minimize Equation (10.5) over new environments $\mathcal{E}_{\text{ad}}$ w.r.t. $\{\delta\theta^e\}_{e \in \mathcal{E}_{\text{ad}}}$.

The locality constraint in the training objective Equation (10.5) enforces $\delta\theta^e$ to remain close to the shared θ^c solutions. It plays several roles. First, it fosters fast adaptation by acting as a constraint over $\theta^c \in \mathbb{R}^{d_\theta}$ during training s.t. minimas $\{\theta^{e\star}\}_{e \in \mathcal{E}}$ are in a neighborhood of θ^c i.e. can be reached from θ^c with few update steps. Second, it constrains the hypothesis space at fixed θ^c. Under some assumptions, it can simplify the resolution of the optimization problem w.r.t. $\delta\theta^e$ by turning optimization to a quadratic convex problem with an unique solution. We show this property for our solution in Theorem 10.1. The positive effects of this constraint will be illustrated on an ODE system in Section 10.3.3.

10.3.3 Context-Informed Hypernetwork

Solving Equation (10.5) involves learning $\delta\theta^e$ for each environment. For adaptation, $\delta\theta^e$ should be inferred from few observations of the new environment. Learning such high-dimensional parameters is prone to over-fitting, especially under scarce data. We propose a hypernetwork-based solution to solve efficiently this problem by operating on a low-dimensional space. It yields fixed-cost adaptation and enables efficient sharing of information across environments.

Formulation We estimate $\delta\theta^e$ through a linear mapping of conditioning information, called context, learned from $\mathcal{D}^e$ and denoted $\xi^e \in \mathbb{R}^{d_\xi}$. $W = (W_1, \cdots, W_{d_\xi}) \in \mathbb{R}^{d_\theta \times d_\xi}$ is the weight matrix of the linear decoder s.t.

$$A_\pi(\mathcal{D}^e) \triangleq \theta^c + W\xi^e, \quad \pi \triangleq \{W, \theta^c, \{\xi^e\}_{e \in \mathcal{E}}\} \tag{10.6}$$

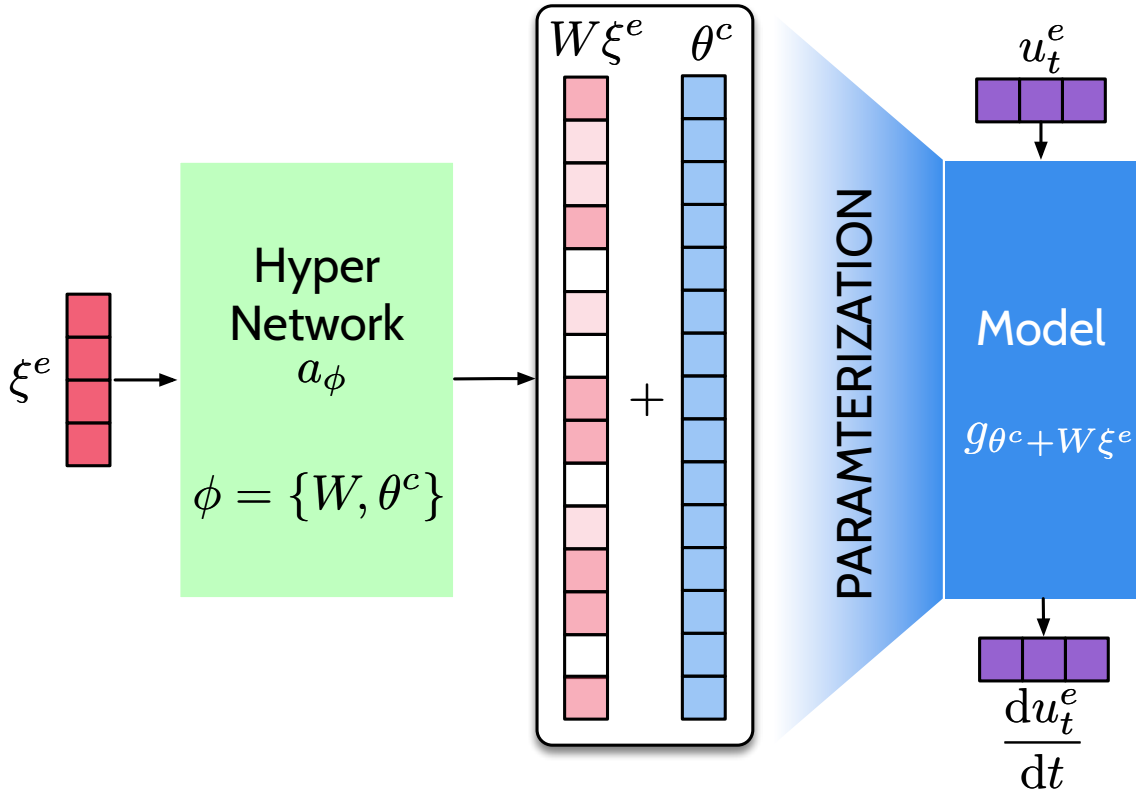


Figure 10.1. – CoDA generates parameters θ^e of a prediction model g on environment e as $\theta^e = \theta^c + W\xi^e$. This model satisfies a low-rank property i.e. $\theta^e - \theta^c$ lies on a subspace of small dimension.

W is shared across environments and defines a low-dimensional subspace $\mathcal{W} \triangleq \mathrm{Span}(W_1, ..., W_{d_\xi})$, of dimension at most d_ξ, to which the search space of $\delta\theta^e$ is restricted. ξ^e is specific to each environment and can be interpreted as learning rates along the rows of W. In our experiments, $d_\xi \ll d_\theta$ is small, at most 2. Thus, *adaptation to new environments only requires to learn very few parameters, which define a completely new dynamics model* g.

A_π corresponds to an affine mapping of ξ^e parametrized by $\{W, \theta^c\}$, a.k.a. a linear hypernetwork. Note that hypernetworks (Ha et al. 2017b) have been designed to handle single-environment problems and learn a separate context per layer. Our formalism involves multiple environments and defines a context per environment for all layers of g. Linearity of the hypernetwork is not restrictive as contexts are directly learned through an inverse problem detailed in eqs. (10.7) and (10.8), s.t. expressivity is similar to a nonlinear hypernetwork with a final linear activation.

Objectives We derive the training and adaptation objectives by inserting Equation (10.6) into Equation (10.5). For training, both contexts and hypernetwork are learned with Equation (10.7):

$$\min_{\theta^c, W, \{\xi^e\}_{e\in\mathcal{E}_{\text{tr}}}} \sum_{e\in\mathcal{E}_{\text{tr}}} \Big(\mathcal{L}(\theta^c + W\xi^e, \mathcal{D}^e) + \lambda \|W\xi^e\|^2 \Big) \tag{10.7}$$

After training, θ^c is kept fixed and for adaptation to a new environment, only the context vector ξ^e is learned with:

$$\min_{\{\xi^e\}_{e\in\mathcal{E}_{\text{ad}}}} \sum_{e\in\mathcal{E}_{\text{ad}}} \Big(\mathcal{L}(\theta^c + W\xi^e, \mathcal{D}^e) + \lambda \|W\xi^e\|^2 \Big) \tag{10.8}$$

Implementation of eqs. (10.7) and (10.8) is detailed in Section 10.4. We apply gradient descent. In Theorem 10.1, we show for $\|\cdot\| = \ell_2$, that Equation (10.8) admits an unique solution, recovered from initialization at $\mathbf{0}$ with a single preconditioned gradient step, projected onto subspace $\mathcal{W}$ defined by W.

Proposition 10.1 (Proof in Appendix E.2)**.** *Given $\{\theta^c, W\}$ fixed, if $\|\cdot\| = \ell_2$, then Equation (10.8) is quadratic. If $\lambda' W^\top W$ or $\bar{H}^e(\theta^c) = W^\top \nabla^2_\theta \mathcal{L}(\theta^c, \mathcal{D}^e) W$ are invertible then $\bar{H}^e(\theta^c) + \lambda' W^\top W$ is invertible except for a finite number of λ' values. The problem in Equation (10.8) is then also convex and admits an unique solution, $\{\xi^{e\star}\}_{e\in\mathcal{E}_{ad}}$. With $\lambda' \triangleq 2\lambda$,*

$$\xi^{e*} = -\Big(\bar{H}^e(\theta^c) + \lambda' W^\top W\Big)^{-1} W^\top \nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e) \tag{10.9}$$

Interpretation We now interpret CoDA by visualizing its loss landscape around θ^c in Figure 10.2, following H. Li et al. (2018). We consider the *Lotka-Volterra* system, described in Section 10.5.1. Loss values are projected onto subspace $\mathcal{W}$, where $d_\xi = 2$. We make three observations. First, across environments, the loss is smooth and has a single minimum around θ^c. Second, the local optimum of the loss is close to θ^c across environments. Finally, the minimal loss value on $\mathcal{W}$ around θ^c is low across environments. The two first properties were discussed in Section 10.3.2 and are a direct consequence of the locality constraint on subspace $\mathcal{W}$. When $\|\cdot\| = \ell_2$, it makes the optimization problem in Equation (10.7) quadratic w.r.t. ξ^e and convex under invertibility of $\bar{H}^e(\theta^c) + \lambda' W^\top W$ as detailed in Theorem 10.1. We provided in Equation (10.9) the closed form expression of the solution. It also imposes small $\|\xi^e\|$ s.t. when minimizing the loss in Equation (10.7), θ^c is forced to be close to local optimas of all training environments. The final observation illustrates that CoDA is able to find a subspace $\mathcal{W}$ on which there are environment-specific parameters with low loss values i.e. that low-rank adaptation performs well. We provide in Appendix E.8, some further comparison with the loss landscape of ERM, projected onto the Span of the two principal gradient directions. We show that ERM does not find low loss values, as it aims

at finding θ^c with good performance across environments, thus cannot model several dynamics.

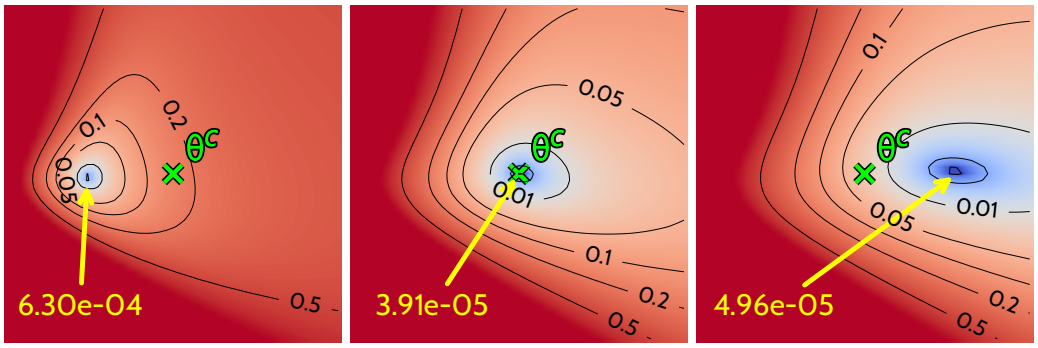


Figure 10.2. – CoDA's loss landscape centered in θ^c, marked with ×, for 3 environments on the *Lotka-Volterra* ODE. Loss values are projected onto subspace $\mathcal{W}$, with $d_\xi = 2$. $\forall e, \rightarrow$ points to the local optimum $\theta^{e\star}$ with loss value reported in yellow.

10.3.4 Validity for Dynamical Systems

We further motivate low-rank decoding in our context-informed hypernetwork approach by providing some evidence that gradients at θ^c across environments define a low-dimensional subspace. We consider the loss $\mathcal{L}$ in Equation (10.3) and define the gradient subspace in Theorem 10.2.

Definition 10.2 (Gradient directions)**.** With $\mathcal{L}$ in Equation (10.3), $\forall \theta^c \in \mathbb{R}^{d_\theta}$ parameterizing a dynamics model g_{θ^c}, the subspace generated by gradient directions at θ^c across environments $\mathcal{E}$ is denoted $\mathcal{G}_{\theta^c} \triangleq \text{Span}(\{\nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e)\}_{e\in\mathcal{E}})$.

We show, in Theorem 10.3, low-dimensionality of $\mathcal{G}_{\theta^c}$ for linearly parametrized systems.

Proposition 10.3 (Low-rank under linearity. Proof in Appendix E.2)**.** *Given a class of linearly parametrized dynamics $\mathcal{F}$ with d_p varying parameters, $\forall \theta^c \in \mathbb{R}^{d_\theta}$, subspace $\mathcal{G}_{\theta^c}$ in Theorem 10.2 is low-dimensional and* $\dim(\mathcal{G}_{\theta^c}) \leq d_p \ll d_\theta$.

The linearity assumption is not restrictive as it is present in a wide variety of real-world systems e.g. Burger or Korteweg–De Vries PDE (M. Raissi et al. 2019), convection-diffusion (Long et al. 2018), wave and reaction diffusion equations (Yin et al. 2021b) *etc*. Under nonlinearity, we do not have the same theoretical guarantee, yet, we show empirically in Appendix E.4 that low-dimensionality of parameters of the dynamics model g_θ still holds for several systems. This property is comforted by recent studies that highlighted that gradients are low-rank throughout optimization in single-domain settings, meaning that the solution space is low-dimensional (Gur-Ari et al. 2019; C. Li et al. 2018; H. Li et al. 2018).

In the same spirit as CoDA, this property was leveraged to design efficient solutions to the learning problems (Frankle and Carbin 2019; Vogels et al. 2019).

10.3.5 Benefits of CoDA

We highlight the benefits of CoDA w.r.t. related methods, with further details in Appendix E.1.1. The adaptation rule in Equation (10.4) is similar to the one used in gradient-based meta-learning. Yet, our first order joint optimization problem in Equation (10.5) considerably simplifies the complex bi-level optimization problem (Antoniou et al. 2019). Moreover, CoDA introduces the two key properties of locality constraint and low-rank adaptation which guarantee efficient adaptation to new environments as discussed in Section 10.3.3. It generalizes contextual meta-learning methods (Garnelo et al. 2018; Zintgraf et al. 2019), which also perform low-rank adaptation, via the hypernetwork decoder (details in Appendix E.1.2). Our decoder learns complex environment-conditional dynamics models while controlling their complexity. CoDA learns context vectors through an inverse problem as Zintgraf et al. (2019). This decoder-only strategy is particularly efficient and flexible in our setting. An alternative is to infer them via a learned encoder of $\mathcal{D}^e$ as Garnelo et al. (2018). Yet, the latter was observed to underfit (Kim et al. 2019), requiring extensive tuning of the encoder and decoder architecture. CoDA is easy to implement and maintains expressivity with a linear decoder.

10.4 Framework Implementation

We detail how to perform trajectory-based learning with our framework and describe two instantiations of the locality constraint. We detail the corresponding pseudo-code.

Trajectory-Based Formulation As derivatives in Equation (10.3) are not directly observed, we use in practice for training a trajectory-based formulation of Equation (10.3), defined over N trajectories $\{\mathcal{D}_i^e\}_{i=1}^N$, discretized over a uniform temporal and spatial grid. $\Delta t, \Delta s$ are the temporal and spatial resolutions and T, S the temporal horizon and spatial grid size. Each trajectory includes $\frac{T}{\Delta t}\left(\frac{S}{\Delta s}\right)^{d_s}$ states, where for PDEs d_s is the spatial dimension and for ODEs $d_s = 0$. With $t_k = k\Delta t$, and s_j the j^{th} spatial coordinate, we denote $x^{e,i}(t_k, s_j)$ the state value

Table 10.1. – Test MSE (↓) in training environments $\mathcal{E}_{\text{tr}}$ (*In-Domain*) and new adaptation environments $\mathcal{E}_{\text{ad}}$ (*Adaptation*).

	LV ($\times 10^{-5}$)		GO ($\times 10^{-4}$)		GS ($\times 10^{-3}$)		NS ($\times 10^{-4}$)	
	In-domain	Adaptation	In-domain	Adaptation	In-domain	Adaptation	In-domain	Adaptation
MAML	60.3±1.3	3150±940	57.3±2.1	1081±62	3.67±0.53	2.25±0.39	68.0±8.0	51.1±4.0
ANIL	381±76	4570±2390	74.5±11.5	1688±226	5.01±0.80	3.95±0.11	61.7±4.3	48.6±3.2
Meta-SGD	32.7±12.6	7220±4580	42.3±6.9	1573±413	2.85±0.54	2.68±0.20	53.9±28.1	44.3±27.1
LEADS	3.70±0.27	47.61±12.47	31.4±3.3	113.8±41.5	2.90±0.76	1.36±0.43	14.0±1.55	28.6±7.23
CAVIA-FiLM	4.38±1.15	8.41±3.20	4.44±1.46	3.87±1.28	2.81±1.15	1.43±1.07	23.2±12.1	22.6±9.88
CAVIA-Concat	2.43±0.66	6.26±0.77	5.09±0.35	2.37±0.23	2.67±0.48	1.62±0.85	25.5±6.31	26.0±8.24
CoDA-ℓ_2	1.52±0.08	1.82±0.24	2.45±0.38	1.98±0.06	1.01±0.15	0.77±0.10	9.40±1.13	10.3±1.48
CoDA-ℓ_1	**1.35±0.22**	**1.24±0.20**	**2.20±0.26**	**1.86±0.29**	**0.90±0.057**	**0.74±0.10**	**8.35±1.71**	**9.65±1.37**

in trajectory i from environment e at spatial coordinate s_j and time t_k. Our loss writes as:

$$\mathcal{L}(\theta, \mathcal{D}^e) = \sum_{i=1}^{N} \sum_{j=1}^{(S/\Delta s)^{d_S}} \sum_{k=1}^{T/\Delta t} \left\| X^{e,i}(t_k, s_j) - \tilde{X}^{e,i}(t_k, s_j) \right\|_2^2$$
$$\tilde{X}^{e,i}(t_k) = X^{e,i}(t_{k-1}) + \int_{t_{k-1}}^{t_k} g_\theta \left(\tilde{X}^{e,i}(\tau) \right) \mathrm{d}\tau \tag{10.10}$$

where $X(t) = [X(t, s_1), \cdots, X(t, s_{(S/\Delta s)^{d_S}})]^\top$ is the state vector over the spatial domain at t. We apply for integration a numerical solver (E. Hairer et al. 2000) as detailed later.

Locality Constraint Instead of penalizing $\lambda \|W\xi^e\|^2$ in Equation (10.7), we found it more efficient to penalize separately W and ξ^e. We thus introduce the following regularization:

$$R(W, \xi^e) \triangleq \lambda_\xi \|\xi^e\|_2^2 + \lambda_\Omega \Omega(W) \tag{10.11}$$

It involves hyperparameters $\lambda_\xi, \lambda_\Omega$ and a norm $\Omega(W)$ which depends on the choice of $\|\cdot\|$ in Equation (10.5). We consider two variations of $\|\cdot\|$. CoDA-ℓ_2 sets $\|\cdot\| \triangleq \ell_2(\cdot)$ and $\Omega \triangleq \ell_2^2$, constraining $W\xi^e$ to a sphere. CoDA-ℓ_1 sets $\|\cdot\| \triangleq \ell_1(\cdot)$ and $\Omega = \ell_{1,2}$ over rows i.e. $\Omega(W) \triangleq \sum_{i=1}^{d_\theta} \|W_{i,:}\|_2$ to induce sparsity and find most important parameters for adaptation. $\ell_{1,2}$ constrains $\mathcal{W}$ to be axis-aligned; then the number of solutions is finite as $\dim(\mathcal{W})$ is finite. Minimizing $R(W, \xi^e)$ can be interpreted as minimizing an upper-bound to $\|\cdot\|$, derived in appendix E.5 for each variation.

Pseudo-code We solve Equation (10.7) for training and Equation (10.8) for adaptation using eqs. (10.10) and (10.11) and Algorithm 10.1. We back-propagate through the solver's internals with torchdiffeq (R. T. Q. Chen 2021) and apply exponential Scheduled Sampling (Goyal et al. 2016) to stabilize training.

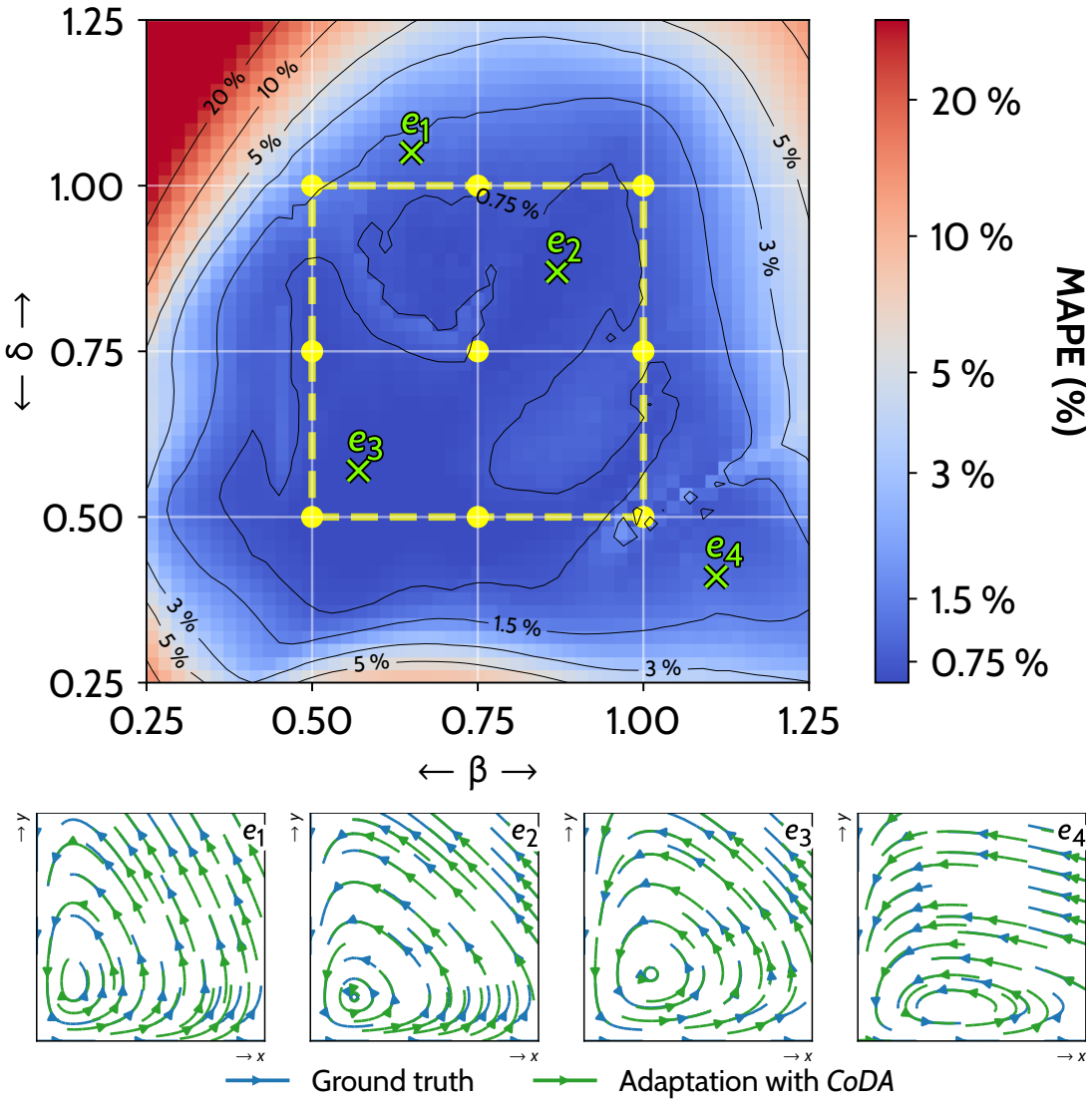


Figure 10.3. – *Adaptation* results with CoDA-ℓ_1 on LV. Parameters (β, δ) are sampled in $[0.25, 1.25]^2$ on a 51×51 uniform grid, leading to 2601 adaptation environments $\mathcal{E}_{\text{ad}}$. • are training environments $\mathcal{E}_{\text{tr}}$. We report MAPE (↓) across $\mathcal{E}_{\text{ad}}$ (Top). On the bottom, we choose four of them (×, e_1–e_4), to show the ground-truth (blue) and predicted (green) phase space portraits. x, y are respectively the quantity of prey and predator in the system in Equation (E.4).

Algorithm 10.1 CoDA Pseudo-code: Training and Adaptation

Training

Data: $\mathcal{E}_{\text{tr}} \subset \mathcal{E}$, $\{\mathcal{D}^e\}_{e\in\mathcal{E}_{\text{tr}}}$ with $\forall e \in \mathcal{E}_{\text{tr}}, \#\mathcal{D}^e = N$;

Input: $\pi = \{W, \theta^c, \{\xi^e\}_{e\in\mathcal{E}_{\text{tr}}}\}$ where $W \in \mathbb{R}^{d_\theta \times d_\xi}, \theta^c \in \mathbb{R}^{d_\theta}$ are randomly initialized and $\forall e \in \mathcal{E}_{\text{tr}}, \xi^e = \mathbf{0} \in \mathbb{R}^{d_\xi}$;

while *Not Converged* **do**

$$\pi \leftarrow \pi - \eta\nabla_\pi\Big(\sum_{e\in\mathcal{E}_{\text{tr}}}\sum_{i=1}^{N}\mathcal{L}(\theta^c + W\xi^e, \mathcal{D}_i^e) + R(W, \xi^e)\Big) \tag{10.12}$$

Adaptation

Data: $e \in \mathcal{E}_{\text{ad}}$; $\mathcal{D}^e$ where $\#\mathcal{D}^e = N$;

Input: Trained $W \in \mathbb{R}^{d_\theta \times d_\xi}, \theta^c \in \mathbb{R}^{d_\theta}$ and $\xi^e = \mathbf{0} \in \mathbb{R}^{d_\xi}$

while *Not Converged* **do**

$$\xi^e \leftarrow \xi^e - \eta\nabla_{\xi^e}\Big(\sum_{i=1}^{N}\mathcal{L}(\theta^c + W\xi^e, \mathcal{D}_i^e) + R(W, \xi^e)\Big) \tag{10.13}$$

10.5 Experiments

We validate our approach on four classes of challenging nonlinear temporal and spatiotemporal physical dynamics, representative of various fields e.g. chemistry, biology and fluid dynamics. We evaluate in-domain and adaptation prediction performance and compare them to related baselines. We also investigate how learned context vectors can be used for system parameter estimation. We consider a few-shot adaptation setting where only a single trajectory is available at adaptation time on new environments.

10.5.1 Dynamical Systems

We consider four ODEs and PDEs described in Appendix E.6.1. ODEs include *Lotka-Volterra* (`LV`, (Lotka 1925)) and *Glycolitic-Oscillator* (`GO`, (Daniels and Nemenman 2015)), modelling respectively predator-prey interactions and the dynamics of yeast glycolysis. PDEs are defined over a 2D spatial domain and include *Gray-Scott* (`GS`, (Pearson 1993)), a reaction-diffusion system with complex spatiotemporal patterns and the challenging *Navier-Stokes* system (`NS`, (Stokes 1851)) for incompressible flows. All systems are nonlinear w.r.t. system states and all but `GO` are linearly parametrized. The analysis in Section 10.3.4 covers all systems but `GO`. Experiments on the latter show that CoDA also extends to nonlinearly parameterized systems.

10.5.2 Baselines

We consider three families of baselines, compared in Appendix Figure E.1 and detailed in Section 10.6. First, Gradient-Based Meta-Learning (GBML) methods MAML (Finn et al. 2017), ANIL (Rusu et al. 2019) and Meta-SGD (Zhenguo Li et al. 2017). Second, the Multi-Task Learning method LEADS (Yin et al. 2021a). Finally, the contextual meta-learning method CAVIA (Zintgraf et al. 2019), with conditioning via concatenation (Concat) or linear modulation of final hidden features (FiLM, (Perez et al. 2018)). Baselines consider also the loss in Equation (10.10).

10.5.3 Architecture, Optimizer and Hyperparameters

We use MLPs for ODEs, ConvNets for GS and Fourier Neural Operators Zongyi Li et al. 2021 for NS (details in Appendix E.6.2). Adam optimizer Diederik P. Kingma and Ba 2015 is used for all datasets. We tuned d_ξ and observed that $d_\xi = d_p$, the number of system parameters that vary across environments, performed best. This is reported in an ablation study in Section 10.5.5. Solvers, optimization and regularization hyperparameters are detailed in Appendix E.6.2.

10.5.4 Experimental Setting

Each environment $e \in \mathcal{E}$ is defined by system parameters and we denote $p^e \in \mathbb{R}^{d_p}$ those that vary across $\mathcal{E}$. d_p represents the degrees of variations in $\mathcal{F}$ and is set to $d_p = 2$ for LV, GO, GS and $d_p = 1$ for NS. We define in Appendix E.6.1 for each system the number of training and adaptation environments ($\#\mathcal{E}_{\text{tr}}$ and $\#\mathcal{E}_{\text{ad}}$) and the corresponding parameters. We also define in Appendix E.6.1 the number of training trajectories N per environment and the distribution $p(X_0)$ from which are sampled initial conditions.

We perform two types of evaluation: in-domain generalization on $\mathcal{E}_{\text{tr}}$ *(In-domain)* and out-of-domain adaptation to new environments $\mathcal{E}_{\text{ad}}$ (*Adaptation*). Evaluation is performed on 32 new test trajectories per environment with initial conditions sampled from $p(X_0)$. We report, in our tables, mean and standard deviation of Mean Squared Error (MSE) across test trajectories (Equation (10.10)) over four different seeds. We report, in our figures, Mean Absolute Percentage Error (MAPE) in $\%$ over trajectories, as it allows to better compare performance across environments and systems. We define $\text{MAPE}(z, y)$ between a d-dimensional input z and target y as $\frac{1}{d}\sum_{j=1\ldots d: y_j \neq 0} \frac{|z_j - y_j|}{|y_j|}$. Over a trajectory, it extends into $\int_{t\in I} \text{MAPE}(\tilde{X}(t), X(t))\mathrm{d}t$, with $\tilde{X}$ in Equation (10.10).

10.5.5 Generalization Results

In Table 10.1, we observe that CoDA improves significantly test MSE w.r.t. our baselines for both *In-Domain* and *Adaptation* settings. We visualize trajectories for PDE systems in Appendix E.7 and also notice improvements. Across datasets, all baselines are subject to a drop in performance between *In-Domain* and *Adaptation* while CoDA maintains remarkably the same level of performance in both cases. In more details, GBML methods (MAML, ANIL, Meta-SGD) overfit on training *In-Domain* data especially when data is scarce. This is the case for ODEs which include less system states for training than PDEs. LEADS performs better than GBML but overfits for *Adaptation* as it does not adapt efficiently. CAVIA-Concat/FiLM perform better than GBML and LEADS, as they leverage a context, but are less expressive than CoDA. Both variations of CoDA perform best as they combine the benefits of low-rank adaptation and locality constraint. CoDA-ℓ_1 is better than CoDA-ℓ_2 as it induces sparsity, further constraining the hypothesis space.

We evaluate in Figure 10.3 CoDA-ℓ_1 on LV for *Adaptation* over a wider range of adaptation environments ($\#\mathcal{E}_{\text{ad}} = 51 \times 51 = 2601$). We report mean MAPE over $\mathcal{E}_{\text{ad}}$ (Top). We observe three regimes: inside the convex hull of training environments $\mathcal{E}_{\text{tr}}$, MAPE is very low; outside the convex-hull, MAPE remains low in a neighborhood of $\mathcal{E}_{\text{tr}}$; beyond this neighborhood, MAPE increases. CoDA thus generalizes efficiently in the neighborhood of training environments and degrades outside this neighborhood. We plot reconstructed phase space portraits (Bottom) on four selected environments and observe that the learned solution closely follows the target trajectories.

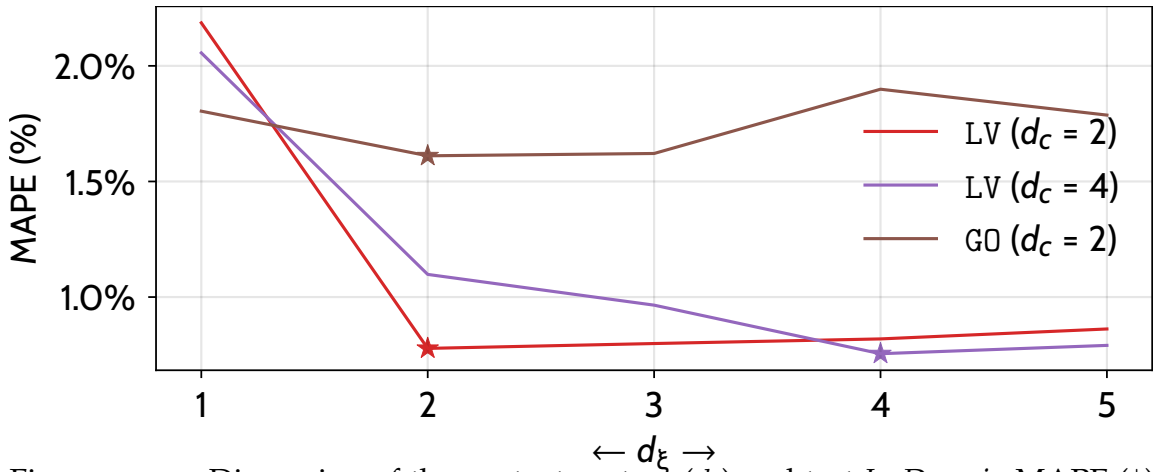


Figure 10.4. – Dimension of the context vectors (d_ξ) and test *In-Domain* MAPE (↓) with CoDA-ℓ_1. "$\star$" is the minimum of MAPE.

Table 10.2. – ℓ_2 locality constraint and *In-Domain* test MSE (↓).

	LV ($\times 10^{-5}$)		GO ($\times 10^{-4}$)	
CoDA	W/o ℓ_2	With ℓ_2	W/o ℓ_2	With ℓ_2
Full	2.28±0.29	1.52±0.08	2.98±0.71	2.45±0.38
FirstLayer	2.25±0.29	2.41±0.23	2.38±0.71	**2.12±0.55**
LastLayer	1.86±0.24	**1.27±0.03**	28.4±0.60	28.4±0.64

10.5.6 Ablation Studies

We perform two studies on LV and GO. In a first study in Table 10.2, we evaluate the gains due to using ℓ_2 locality constraint on *In-Domain* evaluation. On line 1 (Full), we observe that CoDA-ℓ_2 performs better than CoDA without locality constraint. Prior work perform adaptation only on the final layer with some performance improvements on classification or Hamiltonian system modelling (Raghu et al. 2020; Y. Chen et al. 2020). In order to evaluate this strategy, we manually restrict hypernetwork-decoding to only one layer in the dynamics model g_θ, either the first layer (line 2) or the last layer (line 3). We observe that the importance of the layer depends on the parameterization of the system: for LV, linearly parametrized, the last layer is better while for GO, nonlinearly parametrized, the first layer is better. CoDA-ℓ_1 generalizes this idea by automatically selecting the useful adaptation subspace via $\ell_{1,2}$ regularization, offering a more flexible approach to induce sparsity.

In a second study in Section 10.5.5, we analyze the impact on MAPE of the dimension of context vectors d_ξ for CoDA-ℓ_1. We recall that d_ξ upper-bounds the dimension of the adaptation subspace $\mathcal{W}$ and was cross-validated in Table 10.1. In the following, d_p is the number of parameters that vary across environments. We illustrate the effect of the cross-validation on MAPE for $d_p = 2$ on LV and GO as in Section 10.5.5 and additionally for $d_p = 4$ on LV. We observe in Section 10.5.5 that the minimum of MAPE is reached for $d_\xi = d_p$ with two regimes: when $d_\xi < d_p$, performance decreases as some system dimensions cannot be learned; when $d_\xi > d_p$, performance degrades slightly as unnecessary directions of variations are added, increasing the hypothesis search space. This study shows the validity of the low-rank assumption and illustrates how the unknown d_p can be recovered through cross-validation.

10.5.7 Parameter Estimation

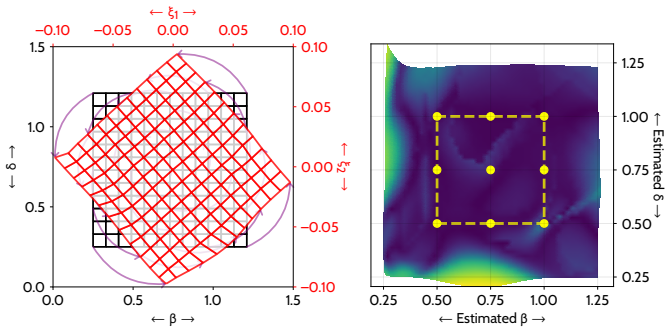
In Figure 10.5a (Left), we visualize on LV the learned context vectors ξ^e (Red) and the system parameters p^e (Black), $\forall e \in \mathcal{E}_{\text{tr}} \cup \mathcal{E}_{\text{ad}}$. We observe empirically a linear bijection between these two sets of vectors. Such a correspondence being learned on the training environments, we can use the correspondence to verify if it still applies to the new adaptation environments. Said otherwise, we can check if our model is able to infer the true parameters for new environments. We evaluate in Table 10.3 the parameter estimation MAPE over LV, GS and NS. Figure 10.5 displays estimated parameters along estimation MAPE for LV and NS. This visualization is provided for GS in Figure E.6 (Appendix E.9). Experimentally, we observe low MAPE inside the convex-hull of training environments and even outside it. This shows that CoDA identifies accurately the unknown system parameters p^e with little supervision.

We justify these empirical observations theoretically in Theorem 10.5 under the conditions in Theorem 10.4:

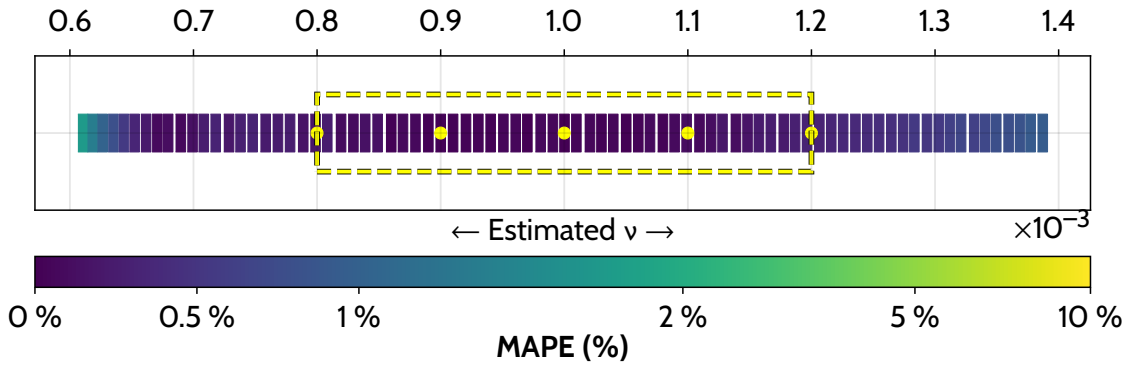
Assumption 10.4. (a) the dynamics in $\mathcal{F}$ are linear w.r.t. inputs and system parameters, (b) the dynamics model g and hypernetwork A are linear, (c) $\forall e \in \mathcal{E}$, the system parameters of f^e, $p^e \in \mathbb{R}^{d_p}$, are unique, (d) the dimension of context vectors d_ξ is fixed to d_p, (e) the system parameters p_i of dynamic $f_i \in \mathcal{B}$, where $\mathcal{B}$ is a basis of $\mathcal{F}$, are known.

Proposition 10.5 (Identification under linearity. Proof in Appendix E.3)**.** *Under Theorem 10.4, system parameters are perfectly identified on new environments if the dynamics model g and hypernetwork A satisfy $\forall f_i \in \mathcal{B}, g_{A(p_i)} = f_i$.*

Intuitively, Theorem 10.5 says that given some observations representative of the degrees of variation of the data (a basis of $\mathcal{F}$) and given the system parameters for these observations (condition (d) in Theorem 10.4), we are guaranteed to recover the parameters of new environments for a family systems. This strong guarantee requires strong conditions described in Theorem 10.4. (a), (b) state that the systems should be linear w.r.t. inputs and that the dynamics model should be linear too. Linearity of the hypernetwork is not an issue as detailed in Section 10.3.3. (c) applies to several real-world systems used in our experiments (*cf.* Appendix E.3 lemmas E.1 and E.2). (d) is not restrictive as we showed that d_p is recovered through cross-validation (Section 10.5.5). We propose an extension of Theorem 10.5 to nonlinear systems w.r.t. inputs and nonlinear dynamics model g in Appendix E.3 Theorem E.4. This alleviates the linearity assumption in (a), (b) and fits our experimental setting.



(a) LV, true parameters $(\beta, \delta) \in [0.25, 1.25]^2$



(b) NS, true parameter $\nu \in [0.6 \cdot 10^{-3}, 1.4 \cdot 10^{-3}]$

Figure 10.5. – Parameter estimation with CoDA-ℓ_1 in new adaptation environments on LV (a) and NS (b). On LV, we visualize, on the left, context vectors ξ (red) and true parameters (β, δ) (black). On other figures, we visualize estimated parameters with corresponding estimation MAPE (↓). • are training environments $\mathcal{E}_{\text{tr}}$ with known parameters. - - delimits the convex hull of $\mathcal{E}_{\text{tr}}$.

10.6 Related Work

We review OoD, Multi-Task Learning (MTL) and meta-learning methods and extensions to dynamical systems.

Learning in Multiple Environments OoD methods extend the ERM objective to learn domain invariants e.g. via robust optimization (Sagawa et al. 2020) or Invariant Risk Minimization (IRM) Arjovsky et al. (2019) and Krueger et al. (2021). However, they are not adapted to our problem as an unique model is learned. CoDA is closer to meta-learning and MTL. A standard meta-learning approach is gradient-based meta-learning (GBML), which learns a model initialisation through bi-level optimization. GBML can then adapt to a new task with few gradient steps. The standard GBML method is MAML (Finn et al. 2017), extended in various work. ANIL (Raghu et al. 2020) restricts meta-learning to the last layer of a classifier while other work improve adaptation by preconditioning the gradient (Y. Lee

Table 10.3. – Parameter estimation MAPE (↓) for CoDA-ℓ_1 on LV ($\#\mathcal{E}_{\text{tr}} = 9$), GS ($\#\mathcal{E}_{\text{tr}} = 4$) and NS ($\#\mathcal{E}_{\text{tr}} = 5$).

	In-convex-hull		Out-of-convex-hull		Overall
	MAPE (%)	$\#\mathcal{E}_{\text{ad}}$	MAPE (%)	$\#\mathcal{E}_{\text{ad}}$	MAPE (%)
LV	0.15±0.11	625	0.73±1.33	1976	0.59±1.33
GS	0.37±0.25	625	0.74±0.67	1976	0.65±0.62
NS	0.10±0.08	40	0.51±0.35	41	0.30±0.33

and Choi 2018; Flennerhag et al. 2020; Park et al. 2019) e.g. Meta-SGD (Zhenguo Li et al. 2017) learns dimension-wise inner-loop learning rates. Contextual meta-learning approaches in Zintgraf et al. (2019) and Garnelo et al. (2018) partition parameters into context parameters, adapted on each task, and meta-trained parameters, shared across tasks. CoDA follows the same objective of learning a low-dimensional representation of each task but generalizes these approaches with hypernetworks. For MTL, a standard approach is hard-parameter sharing which shares earlier layers of the network (Caruana 1997). Several extensions were proposed to learn more efficiently from a set of related tasks (S.-A. Rebuffi et al. 2017; S. Rebuffi et al. 2018). Yet, MTL does not address adaptation to new tasks, which is the focus of CoDA. Some extensions have also considered this problem, mainly for classification (H. Wang et al. 2021; Requeima et al. 2019).

Generalization for Dynamical Systems Only few work have considered generalization for dynamical systems. LEADS (Yin et al. 2021a) is a MTL approach that performs adaptation in functional space. CoDA operates in parameter space, making adaptation more expressive and efficient, and scales better with the number of environments as it does not require training a full new network per environment as LEADS does. A second work is DyAd (R. Wang et al. 2021), a context-aware meta-learning method. DyAd adapts the dynamics model by decoding a time-invariant context, obtained by encoding observed states. However, unlike CoDA, DyAd uses weak supervision obtained from physics quantities to supervise the encoder, which may not always be possible. Moreover, it performs AdaIN modulation (instance normalization + FiLM), a particular case of hypernetwork decoding, which performed worse than CoDA in our experiments.

10.7 Conclusion

We introduced CoDA, a new framework to learn context-informed data-driven dynamics models on multiple environments. CoDA generalizes with little retrain-

ing and few data to new related physical systems and outperforms prior methods on several real-world nonlinear dynamics. Many promising applications of CoDA are possible, notably for spatiotemporal problems, e.g. partially observed systems, reinforcement learning, or NN-based simulation.

Part VI

OTHER WORKS: FEATURE SELECTION VIA REPARAMETERIZATION

CHAPTER 11

FEATURE SELECTION VIA REPARAMETERIZATION

Chapter abstract

We consider the task of feature selection for reconstruction which consists in choosing a small subset of features from which whole data instances can be reconstructed. This is of particular importance in several contexts involving for example costly physical measurements, sensor placement or information compression. To break the intrinsic combinatorial nature of this problem, we formulate the task as optimizing a binary mask distribution enabling an accurate reconstruction. We then face two main challenges. One concerns differentiability issues due to the binary distribution. The second one corresponds to the elimination of redundant information by selecting variables in a correlated fashion which requires modeling the covariance of the binary distribution. We address both issues by introducing a relaxation of the problem via a novel reparameterization of the logitNormal distribution. We demonstrate that the proposed method provides an effective exploration scheme and leads to efficient feature selection for reconstruction through evaluation on several high dimensional image benchmarks. We show that the method leverages the intrinsic geometry of the data, facilitating reconstruction.

The work in this chapter has led to the publication of a conference paper:

Jérémie Dona and Patrick Gallinari (2021). "Differentiable Feature Selection, a Reparameterization Approach". In: *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. European Conference on Machine Learning, Principles, and Practice of Knowledge Discovery in Databases. Vienne, Austria

11.1 Introduction

Learning sparse representations of data finds essential real-world applications as in budget learning where the problem is limited by the number of features available or in embedded systems where the hardware imposes computational limitations. Feature selection serves similar objectives giving insights about variable dependencies and reducing over-fitting (Guyon and Elisseeff 2003). Combined with a reconstruction objective, feature selection is a sensible problem when collecting data is expensive which is often the case with physical processes. For example, consider optimal sensor placement. This task consists in optimizing the location of sensors measuring a scalar field over an area of interest (e.g pressure, temperature) to enable truthful reconstruction of the signal on the whole area. It finds applications in climate science (Haeberli et al. 2007; McPhaden et al. 2009), where key locations are monitored to evaluate the impact of climate change on snow melt and Monsoon. These examples illustrate how feature selection for reconstruction may be critically enabling for large scale problems where measurements are costly.

Common practices for feature selection involves a ℓ_1-regularization over the parameters of a linear model to promote sparsity (Tibshirani 1996). Initiated by (Tibshirani 1996), several refinements have been developed for feature selection. For example, (Yang et al. 2019) employs a $\ell_{2,1}$-norm in a linear auto-encoder. (Han et al. 2017) impose a ℓ_1-penalty on the first layer of a deep auto-encoder to select features from the original signal. Finally, Group-Lasso methods extended lasso by applying the sparse ℓ_1-penalty over pre-computed chunks of variables to take prior knowledge into account while selecting features. Theses approaches suffer from two main limitations: the design of the groups for Group-Lasso methods and the loss of the intrinsic structure of the data as both (Yang et al. 2019; Han et al. 2017) treat the input signal as a vector. Moreover, non-linear ℓ_1 based methods for feature selection and reconstruction are intrinsically ill posed. Like Group-Lasso methods, our proposition aims at selecting variables in a correlated fashion, to eliminate redundant information, while leveraging the structure of the data. We illustrate its efficiency on images but it can be adapted to exploit patterns in other types of structured data as graphs.

We propose a novel sparse embedding method that can tackle feature selection through an end-to-end-approach. To do so, we investigate the learning of binary masks sampled from a distribution over binary matrices of the size of the image, with 1 indicating a selected pixel. We alleviate differentiability issues of learning categorical variables by relying on a continuous relaxation of the problem. The learned latent binary distribution is optimized via a stochastic exploration scheme. We consider the dependency between the selected pixels and we propose to

sample the pixels in the mask in a correlated fashion to perform feature selection efficiently. Accordingly, we learn a correlated logitNormal distribution via the reparameterization trick allowing for an efficient exploration of the masks space while preserving structural information for reconstruction. Finally, sparsity in the embedding is enforced via a relaxation of the ℓ_0-norm. To summarize, we aim at learning a binary mask for selecting pixels from a distribution of input signals x, with $x \in \mathbb{R}^{n \times n}$ for images, enabling an accurate reconstruction. We formulate our problem as learning jointly a parametric sampling operator S which takes as input a random variable $z \in \mathcal{Z} \subseteq \mathbb{R}^d$ and outputs binary masks, i.e. $S : \mathcal{Z} \to \{0, 1\}^{n \times n}$. We introduce two ways to learn the sampling operator S. For reconstruction, an additional operator denoted G learns to reconstruct the data x from the sparse measurements $s \odot x$. Our proposed approach is fully differentiable and can be optimized directly via back-propagation. Our main contributions are:

- We introduce a correlated logitNormal law to learn sparse binary masks, optimized thanks to the reparameterization trick. This reparameterization is motivated statistically. Sparsity is enforced via a relaxed ℓ_0-norm.
- We formulate the feature selection task for 2-D data as the joint learning of a binary mask and a reconstruction operator and propose a novel approach to learn the parameters of the considered logitNormal law.
- We evidence the efficiency of our approach on several datasets: Mnist, CelebA and a complex geophysical dataset.

11.2 Related Work

Our objective of learning binary mask lies in between a few major domains: density modeling, feature selection and compressed sensing.

Density Modeling via Reparameterization

Sampling being not differentiable, different solutions have been developed in order to estimate the gradients of the parameters of a sampling operator. Gradient estimates through score functions (Williams 1992; Yoshua Bengio et al. 2013b) usually suffer from high variance or bias. Reparameterization (Diederik P. Kingma and Welling 2013) provides an elegant way to solve the problem. It consists in sampling from a fixed distribution serving as input to a parametric transformation in order to obtain both the desired distribution and the gradient with respect to the parameters of interest. However, the learning of categorical variables remains tricky as optimizing on a discrete set lacks differentiability. Continuous relaxation

of discrete variables enables parameters optimization through the reparameterization trick. Exploiting this idea, (Maddison et al. 2016; Jang et al. 2017) developed the concrete law as a reparameterization of the Gumbel max variable for sampling categorical variables (Luce 1959). Alternative distributions, defining relaxations of categorical variables can be learned by reparameterization such as the Dirichlet or logitNormal distribution (Figurnov et al. 2018; Kočiský et al. 2016). Nonetheless, most previous approaches learn factorized distribution, thus selecting variables independently when applied to a feature selection task. In contrast, we rely on the logitNormal distribution to propose a reparameterization scheme enabling us to sample the mask pixels jointly, taking into account dependencies between them and exploiting the patterns present in 2-D data.

Feature Selection

Wrapper methods, (Guyon and Elisseeff 2003; Xing et al. 2001; Maldonado and Weber 2009) select features for a downstream task whereas filter methods (X. He et al. 2006; Lei Yu and H. Liu 2003; Koller and Sahami 1996) rank the features according to tailored statistics. Our work belongs to the category of *embedded* methods, that address selection as part of the modeling process. ℓ_1-penalization over parameters, as for instance in Lasso and in Group Lasso variants (Yuan and Y. Lin 2006; Simon et al. 2013; Y. Zhou et al. 2010), is a prototypical embedded method. ℓ_1-penalty was used for feature selection for example in (P. Zhu et al. 2015; Yang et al. 2019) learning a linear encoding with a $\ell_{2,1}$-constraint for a reconstruction objective. Auto-encoders (G. Hinton and R. Salakhutdinov 2006) robustness to noise and sparsity is also exploited for feature selection (Vincent et al. 2008; Makhzani and Frey 2013). For example, AEFS (Han et al. 2017) extends Lasso with non linear auto-encoders, generalizing (P. Zhu et al. 2015). Another line of work learns embedding preserving local properties of the data and then find the best variables in the original space to explain the learned embedding, using either ℓ_1 or $\ell_{2,1}$ constraints (C. Deng et al. 2010; Hou et al. 2014). Closer to our work, (Abid et al. 2019) learn a matrix of weights m, where each row follow a concrete distribution (Maddison et al. 2016). That way each row of matrix m samples one feature in x. The obtained linear projection $m.x$ is decoded by a neural network, and m is trained to minimize the ℓ_2-loss between reconstructions and targets. Because x is treated as a vector, here too, the structure of the data is ignored and lost in the encoding process. Compared to these works, we leverage the dependencies between variables in the 2-D pixel distribution, by sampling binary masks via an adaptation of the logitNormal distribution.

Compressed Sensing

Our work is also related to compressed sensing (CS) where the objective is to reconstruct a signal from limited (linear) measurements (Donoho 2006). Deep learning based compressed sensing algorithms have been developed recently: (Bora et al. 2017) use a pre-trained generative model and optimize the latent code to match generated measurements to the true ones; The measurement process can be optimized along with the reconstruction network as in (Wu et al. 2019). Finally, (Manohar et al. 2018) use a CS inspired method based on the pivots of a QR decomposition over the principal components matrix to optimize the placement of sensors for reconstruction, but scales poorly for large datasets. Our approach differs from CS. Indeed, for CS, measurements are linear combinations of the signal sources, whereas we consider pixels from the original image. Thus, when CS aims at reconstructing from linear measurements, our goal is to preserve the data structural information to select a minimum number of variables for reconstruction.

11.3 Method

We now detail our framework to learn correlated sparse binary masks for feature selection and reconstruction through an end-to-end approach. The choice of the logitNormal distribution, instead of the concrete distribution (Maddison et al. 2016), is motivated by the simplicity to obtain correlated variables thanks to the stability of independent Gaussian law by addition as detailed below. We experimentally show in section 11.4 that taking into account such correlations helps the feature selection task. This section is organized as follows : we first introduce in section 11.3.1 some properties of the logitNormal distribution and sampling method for this distribution. We detail in section 11.3.2 our parameterization for the learning of the masks distribution. Finally, in section 11.3.3 we show how to enforce sparsity in our learned distribution before detailing our reconstruction objective in section 11.3.4.

11.3.1 Preliminaries: logitNormal Law on $[0, 1]$

Our goal is to sample a categorical variable in a differentiable way. We propose to parameterize the sampling on the simplex by the logitNormal law, introduced in (Aitchison and Shen 1980). We detail this reparameterization scheme for the uni-

dimensional case since we aim at learning binary encodings. It can be generalized to learn k-dimensional one-hot vector. Let $z \sim \mathcal{N}(\mu, \sigma)$, and Y defined as:

$$Y = \text{sigmoid}(z) \tag{11.1}$$

Then Y is said to follow a logitNormal law. This distribution defines a probability over $[0, 1]$, admits a density and its cumulative distribution function has an analytical expression used to enforce sparsity in section 11.3.3.

This distribution can take various forms as shown in fig. 11.1 and be flat as well as bi-modal. By introducing a temperature in the sigmoid so that we have, $\text{sigmoid}_\lambda(z) = \frac{1}{1+\exp^{-z/\lambda}}$, we can polarize the logitNormal distribution. In Theorem 11.1 we evidence the link between the o-temperature logitNormal distribution and Bernoulli distribution:

Proposition 11.1 (Limit Distribution). *Let $W \in \mathbb{R}^n$ be a vector and $b \in \mathbb{R}$ a scalar. Let $Y = sigmoid_\lambda(W.z^T + b)$, where $z \sim \mathcal{N}(0, I_n)$, when λ decrease towards 0, Y converges in law towards a Bernoulli distribution and we have:*

$$\lim_{\lambda \to 0} \mathbb{P}(Y = 1) = 1 - \Phi(\frac{-b}{\sqrt{\sum_i w_i^2}}) \tag{11.2}$$

$$\lim_{\lambda \to 0} \mathbb{P}(Y = 0) = \Phi(\frac{-b}{\sqrt{\sum_i w_i^2}}) \tag{11.3}$$

Where Φ is the cumulative distribution function of the Normal law $\mathcal{N}(0, 1)$,

Theorem 11.1 characterizes the limit distribution as the temperature goes down to 0, and Y defines a differentiable relaxation of a Bernoulli variable. This proposition is used to remove randomness in the learned distribution, see section 11.4.

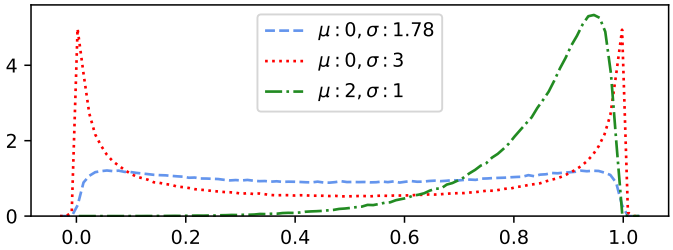


Figure 11.1. – Density of the logitNormal law for various couple (μ, σ): $(\mu = 0, \sigma = 1.78)$ (dashed line), $(\mu = 0, \sigma = 3)$ (dotted line) $(\mu = 2, \sigma = 1)$ (dotted and dashed).

We relax the objective of learning of a binary mask in $\{0, 1\}$ by learning in $[0, 1]$ using the logitNormal law. Let $m \in \mathbb{N}$, be the dimension of the desired logitNor-

mal variable Y. A simple solution for learning the logitNormal distribution of the masks is via independent sampling.

Independent Sampling

A common assumption is that the logitNormal samples originate from a factorized Normal distribution (Kočiský et al. 2016). Thus, the learned parameters of the distribution are: the average $\mu \in \mathbb{R}^m$ and the diagonal coefficients of the covariance matrix $\sigma \in \mathbb{R}^m$, according to:

$$Y = \text{sigmoid}_\lambda(\mu + z \odot \sigma) \tag{11.4}$$

where $\odot$ is the element-wise product and $Y \in \mathbb{R}^m$. Note that, for feature selection on images, one aims at learning a binary mask and thus the latent space has the same dimension as the images, i.e. $m = n \times n$, then $z \in \mathbb{R}^{n \times n}$.

This sampling method has two main drawbacks. First, the coordinates of z are independent and so are the coordinates of Y, therefore such sampling scheme does not take correlations into account. Also, the dimension of the sampling space $\mathcal{Z}$ is the same as Y which might be prohibitive for large images.

We address both limitations in the following section, by considering the relations between the pixel values. In that perspective, Group-Lasso selects variables among previously designed group of variables (Yuan and Y. Lin 2006), reflecting different aspects of the data. Similarly, we want to select variables evidencing different facets of the signal to be observed. Indeed, finding the best subset of variables for the reconstruction implies to eliminate the redundancy in the signal and to explore the space of possible masks. We propose to do so by selecting the variables in a correlated fashion, avoiding the selection of redundant information.

Correlated Sampling:

To palliate the limitations of independent sampling, we model the covariance between latent variables by learning linear combinations between the variables in the prior space $\mathcal{Z}$. Besides, considering dependencies between latent variables, this mechanism reduces the dimension of the sampling space $\mathcal{Z}$, allowing for a better exploration of the latent space. In order to generate correlated variables from a lower dimensional space, we investigate the following transformation: let $z \sim \mathcal{N}_d(0, I_d) \in \mathcal{Z} = \mathbb{R}^d$ with $d << m$, $W \in \mathcal{M}_{m,d}(\mathbb{R})$ a weight matrix of size $m \times d$ and $b \in \mathbb{R}^m$ a real vector, then

$$Y = \text{sigmoid}_\lambda(Wz + b) \tag{11.5}$$

represents m-one dimension logitNormal laws due to the stability of independent Gaussian laws by addition. However, the Normal law induced by $Wz+b$ has now a full covariance matrix and not only diagonal coefficient as in eq. (11.4). This reparameterization provides a simple way to sample correlated (quasi)-binary variables, even for high dimension latent space, i.e with m large. Compared to (Abid et al. 2019), our proposition offers a significant advantage for feature selection in images. Indeed, let G be the neural network aiming to reconstruct data x from the selected variable. With our proposition G can access a sparse version of the original signal $Y \odot x$ and can thus leverage both the pixel values and their position in the image for reconstruction. In (Abid et al. 2019) only the selected feature values without structural information are available for the reconstruction.

11.3.2 Parameterizing logitNormal Variables for Feature Selection

Now we have established how to compute correlated logitNormal variables following eq. (11.5), we detail our parameterization for learning. Let $S : \mathcal{Z} \to [0,1]^{n \times n}$ be our sampling operator that generates a binary mask from a random sample z. We consider two approaches to parameterize S so that it follows a logitNormal law. Our first proposition denoted vanilla parameterization directly optimizes W and b from eq. (11.5), while our second approach proposes to explore and optimize the spaces of linear combinations W and biases b.

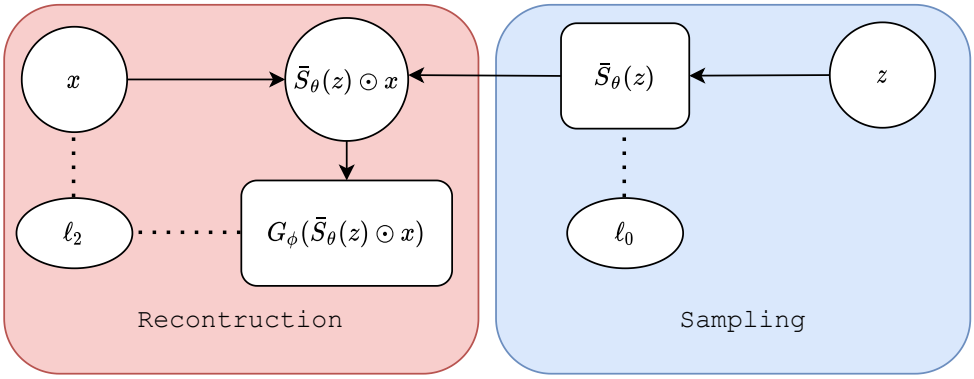


Figure 11.2. – Algorithmic flow of our framework for feature selection for reconstruction. $S_\theta(z)$ has a correlated logitNormal distribution. We sample $z \sim \mathcal{N}(0,1)$. $\bar{S}_\theta(z)$ defines the binary masks and G_ϕ estimate x from $x^{obs} = \bar{S}_\theta(z) \odot x$.

Vanilla Parameterization:

A simple approach is to parameterize S as S_θ according to eq. (11.5). Then, the optimized parameters are: $\theta = (W, b)$ with $W \in \mathcal{M}_{n\times n,d}(\mathbb{R})$ and $b \in \mathbb{R}^{n\times n}$. This sampling process can be summarized by eq. (11.6):

$$\begin{cases} \text{Initialize } W \in \mathcal{M}_{n\times n,d}(\mathbb{R}), b \in \mathbb{R}^{n\times n} \\ z \sim \mathcal{N}(0, I_d) & \text{(11.6a)} \\ S_\theta(z) = \text{sigmoid}(W.z + b) & \text{(11.6b)} \end{cases}$$

In that case each variable in $S_\theta(z)$ follows a logitNormal law. The selected variables are indicated for $S_\theta = 1$. The optimization process allows two degrees of freedom (b and W) for the control of the variance, of the covariance and of the average of the variables of the masks. Note that, this parameterization corresponds to a linear layer followed by a sigmoid activation so that besides tractability for the distribution of Y, it presents the advantage of a simple implementation. Unlike (Abid et al. 2019), our proposition preserves the structure of the data.

HyperNetworks Parameterization:

Aiming to learn a matrix W and a bias vector b that fully characterizes our logitNormal law as eq. (11.5), we leveraged in eq. (11.6) the stability of independent Gaussian law by addition. However, the space of the linear combinations to be learned is high dimensional and structured, hence hard to learn. Also, the optimization of the parameterization as eq. (11.6) is highly dependent on the initialization, as we optimize W and b from a (randomly) chosen start point. Therefore, we want to be able to reach a wider space of parameters W, b. To do so, we build on (Karras et al. 2019) that successfully leverages latent code preprocessing with neural network in the context of adversarial learning for image generation, and (Ha et al. 2017a) where a neural network generates the weights of another neural network to facilitate learning. Therefore, instead of learning directly W, b as in the vanilla approach we propose to learn to sample on the space of linear combination W and biases b. The core idea is to leverage neural networks expressivity to enrich the space of reachable matrices W and vectors b compared to the vanilla approach. To do so we use the random sample z to extract a representation vector $r \in \mathbb{R}^k$. This representation r serves as input to neural net-

works F_b, F_W providing estimates of W and b. To sum up, in the HyperNetwork approach we learn a logitNormal law according to:

$$\begin{cases} z \sim \mathcal{N}(0, I_d),\ r = F_{rep}(z) \in \mathbb{R}^k, & \text{(11.7a)} \\ W = F_W(r) \in \mathcal{M}_{n\times n,d}(\mathbb{R}), & \text{(11.7b)} \\ b = F_b(r) \in \mathbb{R}^{n\times n},\ \text{and finally:} & \text{(11.7c)} \\ S_\theta(z) = \text{sigmoid}(F_b(r) + F_W(r).z), & \text{(11.7d)} \end{cases}$$

Note that as desired $S_\theta(z)$ follows a logitNormal law $\mathcal{LN}(F_b(r), F_W(r)^T.F_W(r))$. This proposition presents several advantages. First, in eq. (11.6) W is a randomly initialized weight matrix, then we only explore one trajectory of optimization from this (randomly chosen) starting point. Also, instead of learning a distribution of masks, this parameterization learns a distribution of transport matrices and biases. Therefore, both F_W and F_b stochastically explore a direction for each sample of z, providing more feedback with respect to the objective of feature selection for reconstruction. This parameterization of W and b offers a way to explore efficiently the space of biases and linear combinations. Also, because it rely on matrix multiplication, this procedure is computationally barely less efficient than the naive one when F_W and F_b are small neural networks. We show experimentally the superiority of this approach in section 11.4.

11.3.3 Sparsity Constraint: ℓ_0-Relaxation

We detail our approach promoting sparsity. Frequently, sparsity in regression settings is enforced thanks to a ℓ_1 penalty on the parameters. However, ℓ_1 approaches may suffer from a shrinking effect due to ill-posedness. Consequently, we introduce an alternative approximation of the ℓ_0-formulation better suited to our feature selection application: we minimize the expected ℓ_0-norm, i.e the probability of each variable in our binary mask to be greater than 0. Thus, we need a non zero probability of sampling 0 which is not the case with the current scheme. Accordingly, we introduce a stretching scheme to obtain a non-zero mass at points 0 and 1 while maintaining differentiability.

Stretched Distribution

To create a mass at 0, we proceed as in (Louizos et al. 2017). Let $Y \in [0,1]^m$ be a logitNormal variable, $\gamma < 0$ and $\eta > 1$ and HT be the hard-threshold function defined by $HT(Y) = \min(\max(Y, 0), 1)$, the stretching is defined as:

$$\bar{Y} = HT\{(\eta - \gamma)Y + \gamma\} \tag{11.8}$$

Thanks to this stretching of our distribution, we have a non zero probability to be zero, i.e $\mathbb{P}(\bar{Y} = 0) > 0$ and also $\mathbb{P}(\bar{Y} = 1) > 0$. We can now derive a relaxed version of the ℓ_0-norm penalizing the probability of the coordinates of $\bar{Y}$.

Sparsity Constraint:

Let $L_0(\bar{Y})$ the expected ℓ_0-norm of our stretched output $\bar{Y}$. Using the notation in eq. (11.8), we have:

$$L_0(\bar{Y}) = \mathbb{E}[\ell_0(\bar{Y})] = \sum_{i=1}^{m} \mathbb{P}(\bar{Y}_i > 0) = \sum_{i=1}^{m} 1 - F_Y(\frac{-\gamma}{\eta - \gamma}), \tag{11.9}$$

where F_Y denotes the cumulative distribution function (CDF) of Y. This loss constrains the random variable Y to provide sparse outputs as long as we can estimate F_Y in a differentiable way. In the case of the logitNormal law, we maintain tractability as Y satisfies eq. (11.5) or eq. (11.4). Thus, for our m-dimensional logitNormal law defined as in eq. (11.5), we have:

$$L_0(\bar{Y}) = \sum_{i} 1 - \Phi\Big(\frac{\log(\frac{-\gamma}{\eta}) - b}{\sqrt{\sum_j W_{j,i}^2}}\Big), \tag{11.10}$$

where Φ is the CDF of the unitary Normal law. Minimizing eq. (11.10) promotes sparsity in the law of Y by minimizing the expected true ℓ_0-norm of the realisation of the random variable Y. We have developed a constraint that promotes sparsity in a differentiable way. Now we focus on how to learn efficiently the parameters of our correlated logitNormal law.

11.3.4 Reconstruction for Feature Selection

We have designed a sparsity cost function and detailed our parameterization to learn our sampling operator, we focus on the downstream task. Consider data $(x_i, y_i)_{i\in[1..N]}$, consisting in paired input x and output y. Feature selection consists in selecting variables in x with a mask s, so that the considered variables: $s \odot x$ explain at best y. Let G be a prediction function and $\mathcal{L}$ a generic cost functional, feature selection writes as:

$$\min_{s,f} \mathbb{E}_{x,y}\ \mathcal{L}(G(s \odot x), y) \quad \text{s.t } ||s||_0 < \lambda, \tag{11.11}$$

In this work we focus on a reconstruction as final task, i.e $y = x$. Besides the immediate application of such formulation to optimal sensors placement and data compression, reconstruction as downstream task requires no other source of data to perform feature selection. Naturally, this framework is adaptable to classification tasks. As a sparse auto-encoding technique, feature selection with a reconstruction objective aims at minimizing the reconstruction error while controlling the sparsity. In this case $G_\phi : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ is our reconstruction network (of parameter ϕ) taking as inputs the sparse image. The feature selection task with an ℓ_2-auto-encoding objective writes as:

$$\min_{\theta,\phi} \mathbb{E}_x ||G_\phi(\bar{S}_\theta(z) \odot x) - x||_2 + \lambda_{sparse} L_0(\bar{S}_\theta(z)) \tag{11.12}$$

A schematic view of our proposition, illustrating the sampling and the reconstruction component is available in Figure 11.2.

11.4 Experiments

We provide experimental results on 3 datasets: MNIST, CelebA and a geophysical dataset resulting from complex climate simulations (IPSL 2018; Sepulchre et al. 2019). We use the traditional train-test split for MNIST and a 80-20 train-test split for the other datasets. The geophysical dataset is composed of surface temperatures anomalies (deviations between average temperature at each pixel for a reference period and observations) and contains 21000 samples (17000 for train). The data have both high (Gulf stream, circum-polar current ...) and low frequencies (higher temperature in the equatorial zone, difference between northern and southern hemispheres ...) that need to be treated accurately due to their influence on the Earth climate. Accuracy in the values of reconstructed pixel is then essential for the physical interpretation. These dense images represent complex dynamics and allow us to explore our method on data with crucial applications and characteristics very different from the digits and faces.

11.4.1 Experimental and Implementation Details

Baselines

Besides our models Vanilla logitNormal, denoted *VLN*, and its hyper-networks couterpart denoted *HNet-LN*, we consider as competing methods the following approaches:

1. Concrete-Autoencoder (Abid et al. 2019) denoted *CAE*.

2. To assess the relevance of our correlated proposition, we investigate a binary mask approach based on the independent logitNormal mask that corresponds to equation eq. (11.4) denoted *ILN*,
3. Another independent binary mask method based on the concrete law (Maddison et al. 2016), denoted *SCT*.

Implementation Details

For all binary mask based methods, we use a ResNet for G_ϕ, (K. He et al. 2015) following the implementation of (Isola et al. 2016). F_{rep}, F_W and F_b are two layers MLP with leaky ReLU activation. For CAE, because the structure of the data is lost in the encoding process, we train G_ϕ as a MLP for MNIST and a DcGAN for geophysical data and CelebA. The code is available at: `https://github.com/JeremDona/feature_selection_public`

Removing Randomness:

All masked based algorithms learn distributions of masks. To evaluate the feature selection capabilities, we evaluate the different algorithms using fixed masks. We rely on theorem 11.1 to remove the randomness during test time. Let S_θ^0 be the o-temperature distribution of the estimated S_θ. We first estimate the expected ℓ_0-norm of the o-temperature distribution: $L_0(S_\theta^0)$. We then estimate two masks selecting respectively the $10 \times \lfloor \frac{L_0(S_\theta^0)}{10} \rfloor$ and $10 \times \lceil \frac{L_0(S_\theta^0)}{10} \rceil$ most likely features (rounding $L_0(S_\theta^0)$ up and down to the nearest ten). This method has the advantage of implicitly fixing a threshold in the learned mask distribution to select or reject features.

We now illustrate the advantage of selecting features in a correlated fashion.

11.4.2 Independent vs Correlated Sampling Scheme:

Is a Covariance Matrix Learned ?

Because we model the local dependencies in the sampling by learning linear mixing of latent variables z, we first verify the structure of the covariance matrix. Figure 11.3 reports the learned covariance matrix of the sampling for MNIST dataset using eq. (11.6) method. Besides the diagonal, extra-diagonal structures emerge, revealing that local correlations are taken into account to learn the sampling distribution.

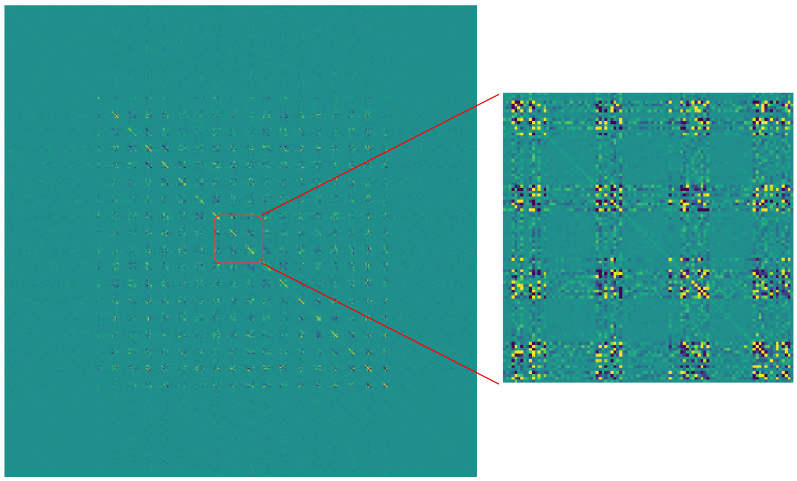


Figure 11.3. – Covariance matrix learned with eq. (11.6), with ≈ 30 pixels selected. Yellow values indicates high positive covariance, blue ones low negative covariance

Independent Sampling Does not Choose

We show in fig. 11.4 the empirical average of the sampled masks for each masked base competing algorithm where all algorithms were trained so that at $L_0(S_\theta^0) \approx 30$. Figure 11.4 clearly shows that concrete base algorithm (SCT) and in a lesser sense (ILN) do not select features, but rather put a uniformly low probability to sample pixels in the center of the image. This means that both algorithms struggle at discriminating important features from less relevant ones. On the other hand, our correlated propositions, Vanilla logitNormal (V-LN, eq. (11.6)) and particularly the hyper-network approach (HNetL, eq. (11.7)) manage to sparsify the distribution prioritizing the selection of important pixels for reconstruction.

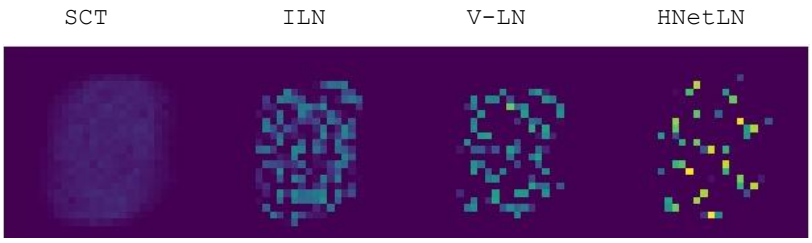


Figure 11.4. – Masks empirical distribution for competing binary masks algorithms on the MNIST datasets for about 30 features in the sampled mask

Table 11.1. – Average Reconstruction Error (MSE) on MNIST, Climate and CelebA datasets for all considered baselines

	MNIST			Climat			CelebA		
# Features	20	30	50	100	200	300	100	200	300
CAE	3.60	3.05	2.40	2.07	1.98	1.96	7.65	6.42	5.7
ILN	3.67	2.41	1.41	1.44	1.05	0.83	7.1	2.56	1.87
SCT	3.72	3.61	2.60	2.20	1.89	1.51	7.99	3.31	2.44
VLN (Ours)	3.22	2.19	1.33	**1.11**	**0.93**	0.79	3.11	1.96	1.50
HNet-Ln (Ours)	**2.15**	**1.53**	**1.06**	1.78	0.96	**0.60**	**2.81**	**1.7**	**1.46**

11.4.3 Feature Selection and Reconstruction

We now quantitatively estimate the impact of our choices on the reconstruction error on the various datasets. First, the mean squared error reconstruction results from table 11.1 tells us that considering the spatial structure of the data enhances reconstruction performance. Indeed, mask based methods consistently over-perform CAE where the data structure is linearized in the encoding process. Furthermore for mask based method, correlated sampling (row V-LN and HNet-LN) also consistently improves over independent sampling based method (row ILN and SCT). Finally, our hyper-network HNet-Ln proposition also improves over the vanilla approach validating our proposition.

11.4.4 Quality of the Selected Features: MNIST Classification

We now assess the relevance of the selected features of our learned masks on another task. To do so, for each learned distribution we train a convolutional neural network, with a DcGAN architecture on MNIST classification task. Here also, the randomness in test set is removed. For each mask we run 5 experiments to account for the variability in the training. Classification results reported in table 11.2 indicate that both our correlated logitNormal propositions consistently beat all considered baselines, validating our choices to learn a sampling scheme in a correlated fashion. Indeed, our propositions systematically reach the lowest minimum and average classification error.

11.4.5 Extension: cGAN

We detailed in the previous experiments feature selection results obtained thanks to an ℓ_2-auto-encoding approach. This choice was motivated because in

Table 11.2. – Classification error in percent for MNIST on test set for all considered baselines. Minimum and average are taken over 5 runs.

# Features	20		30		50	
Metric	Min	Mean	Min	Mean	Min	Mean
CAE	24.4	31.64	8.89	19.60	5.45	6.65
ILN	21.58	28.26	7.96	16.63	4.17	5.33
SCT	20.88	32.79	9.49	18.22	4.11	6.77
VLN (Ours)	**12.15**	**24.74**	**6.38**	**15.07**	3.32	**4.67**
HNet-LN (Ours)	19.23	25.07	7.24	17.80	**2.84**	6.45

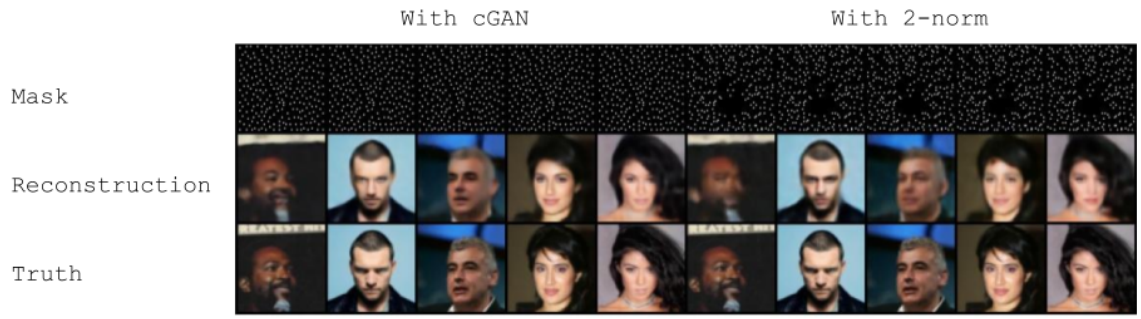


Figure 11.5. – Examples of masks (first row), reconstructions (second row) and true data (last row) for CelebA dataset using either a cGAN (4 first columns) or simple auto-encoding (4 last columns) for 200 selected features. Best viewed in color.

physical measurement all points are equals: we don't want to favor the reconstruction of some part of the image while neglecting another. However, for images such as CelebA this assumption does not hold. Indeed, a realistically reconstructed face can be preferred to a truthful background. Moreover, ℓ_2-auto-encoding suffers from blur in the reconstruction. In that perspective, we can leverage conditional generative adversarial networks (cGAN) approaches (Mirza and Osindero 2014; Isola et al. 2016) that solves the blurriness occurring in ℓ_2-decoding. We implement the cGAN approach of (Isola et al. 2016). Figure 11.5 illustrates that despite both method show good reconstruction, the cGAN approach on CelebA enables a stronger focus on faces facilitating realistic reconstruction.

11.5 Conclusion

In this work, we formulate the feature selection task as the learning of a binary mask. Aiming to select features in images for reconstruction, we developed a novel way to sample and learn a correlated discrete variable thanks to a reparam-

eterization of the logitNormal distribution. The proposed learning framework also preserves the spatial structure of the data, enhancing reconstruction performance. We experimentally show that our proposition to explore the space of covariance matrices and average vectors as in eq. (11.7) is efficient providing us with a sampling with lower variance. Finally, we experimentally evidenced the advantage of learning a correlated sampling scheme instead of independent ones.

Part VII

CONCLUSION AND PERSPECTIVES

Chapter 12

CONCLUDING REMARKS

In this thesis, we have investigated the learning of spatiotemporal data through various prisms. The first part of this thesis explores the construction of inductive biases for deep learning algorithms. In particular, for a spatio-temporal prediction task, we asked ourselves how to design an efficient algorithm that enables interpretability with time-extrapolation performances. The second part of this thesis addressed the learning of dynamical systems with an emphasis on the generalization performances. To do so, we first consider efficient cooperation between numerical models and NN. Then, we proposed a novel approach to generalize to unobserved (but close in some sense) dynamics via an hypernetwork approach.

For each part, we begin by summarizing our contributions and develop elements of perspectives.

12.1 Weak Prior for Neural Networks Architectures

Chapter 6 is dedicated to the construction of purely data-driven algorithms for video-prediction, grounded on an analytical method that aims at solving PDE. It shows promising results to enable the separation of content and motion into two separated representations, hence increasing interpretability. This method relies on several penalties flowing from the separation of variables method. In particular: the content variables is constrained to evolve minimally with time and the temporal variables should evolve in order to match future frames. The use of more involved tools, such as variational encoding, or adversarial learning is a promising way to handle more complex data.

We believe this work contributes to a research track that studies the inductive biases in the design of learning algorithms, especially neural networks. Indeed, the design of NN architectures grounded on physics or PDE is an active topic that has seen recent advances, for example the use of PDE-motivated network architectures for physical data (Lienen and Günnemann 2022; Eliasof et al. 2021). Yet, such physically inspired inductive priors require careful experimental examinations since the source of their gain in performances is insightful (and sometimes unexpected) (Gruver et al. 2022). Also, inductive biases have brought significant

successes to the ML tasks, with for instance the use of convolutions instead of MLP for classification and regression, enforcing the equivariance through translation. Therefore, the incorporation of symmetries and equivariance in learning algorithms, which is a very active research line (Villar et al. 2021; Finzi et al. 2021) in particular for DL algorithm, is a promising path to both explain and improve the performances of DL-algorithms for dynamical system modeling.

12.2 Learning Hybrid-Models from Data

The second component of this thesis, chapters 8 and 9, provides insights on how to bridge physical numerical simulations of dynamical systems and statistical methods. Because of the flexibility of DL, we design constraints that aim at a better cooperation between physical and statistical models. Both propositions of chapters 8 and 9 echoe the difficulties of training of deep networks for physical systems.

In that perspective we developed and analyzed a first framework constraining the magnitude of the statistical component. In a second work, we propose to interpret constraints on either the physical or statistical component as a control of an upper bound of the original forecast error, yielding a general framework to learn hybrid ML-Physical models. These propositions enable us to learn an interpretable hybrid ML-Physics decompostion with increased prediction performances for both forward and inverse problems.

Both our works, addressing the learning of forward and inverse models for dynamical systems, deepen the flourishing literature of NN-based PDE solutions (Maziar Raissi et al. 2019b) and gradient estimates of ODE-solvers (R. T. Q. Chen et al. 2018) that suffer from training and optimization difficulties (Krishnapriyan et al. 2021; S. Wang et al. 2021b; Haber and Ruthotto 2017). Therefore, a task of utmost interest for the scientific machine learning community resides in extending the proposed works to domain specific and real-life applications. Also, an essential extension of these works consist in extending physical constraints to a broader range of Physics-ML decomposition.

12.3 Generalization by Learning from Several Environments

Our last work of chapter 10, close to Meta-Learning and Multi-task learning, tackles the learning of dynamical systems which obeying different dynamics. In

this work, the various dynamics share the same paramteric form but the value of the parameters may differ. To leverage the common nature of all observed dynamics, we propose to learn an affine space in the space of parameters of NN along with the (optimal) coordinate for each environment. This simple formulation enables to adapt efficiently to novel environments with only a small amount of data.

Such a proposition reflects a real-life scenario and enables us to learn from several sources of data. In that perspective, several opportunities emerge. First, a natural extension of this work considers variations in the border and initial conditions instead of variations in the dynamics. Then, considering the topology of the dynamics to learn from could yield more adapted inductive biases for the design of the hyper-network. Eventually, both latter extensions would provide us with a finer understanding on how to learn dynamical systems with NN. Finally, our proposition paves the way towards learning jointly from real-life and simulation dynamics in an efficient way, making an essential step towards handling real-world data.

BIBLIOGRAPHY

Abid, Abubakar, Muhammad Fatih Balin, and James Zou (Jan. 2019). "Concrete Autoencoders for Differentiable Feature Selection and Reconstruction". In: *arXiv:1901.09346 [cs, stat]* (cit. on pp. 128, 132, 133, 136, 271).

Achille, Alessandro and Stefano Soatto (2018). "Emergence of Invariance and Disentanglement in Deep Representations". In: *Journal of Machine Learning Research* 19.50, pp. 1–34 (cit. on p. 194).

Aitchison, J. (1982). "The Statistical Analysis of Compositional Data". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 44.2, pp. 139–177 (cit. on p. 265).

Aitchison, J. and S. M. Shen (1980). "Logistic-Normal Distributions: Some Properties and Uses". In: *Biometrika* 67.2, pp. 261–272 (cit. on pp. 129, 265).

Ajayi, Adekunle, Julien Le Sommer, Eric Chassignet, Jean-Marc Molines, Xiaobiao Xu, Aurelie Albert, and Emmanuel Cosme (2019). "Spatial and Temporal Variability of North Atlantic Eddy Field at Scale less than 100km." In: *Earth and Space Science Open Archive*, p. 28 (cit. on pp. 98, 241).

Ames, William F. (1977). *Numerical Methods for Partial Differential Equations (Second Edition)*. Second Edition. Academic Press, p. ii (cit. on p. 31).

Antoniou, Antreas, Harrison Edwards, and Amos J. Storkey (2019). "How to train your MAML". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL: https://openreview.net/forum?id=HJGven05Y7 (cit. on p. 112).

Arjovsky, Martin, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz (2019). "Invariant Risk Minimization". In: *CoRR* abs/1907.02893. arXiv: 1907.02893. URL: http://arxiv.org/abs/1907.02893 (cit. on p. 120).

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). *Wasserstein GAN*. arXiv: 1701.07875 [stat.ML] (cit. on p. 46).

Arnold, Douglas N. (2015). "Stability, Consistency, and Convergence of Numerical Discretizations". In: *Encyclopedia of Applied and Computational Mathematics*. Ed. by Björn Engquist. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1358–1364. URL: https://doi.org/10.1007/978-3-540-70529-1_407 (cit. on p. 30).

Aubry, Mathieu, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic (June 2014). "Seeing 3D Chairs: Exemplar Part-based 2D-3D Alignment using a Large Dataset of CAD Models". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3762–3769 (cit. on pp. 67, 197).

Avila Belbute-Peres, Filipe de, Kevin A. Smith, Kelsey R. Allen, Joshua B. Tenenbaum, and J. Zico Kolter (2018). "End-to-End Differentiable Physics for Learning and Control". In: *Advances in Neural Information Processing Systems 31*. Ed. by Samy Bengio, Hanna Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. Curran Associates, Inc., pp. 7178–7189 (cit. on p. 55).

Ayed, Ibrahim, Emmanuel de Bézenac, Arthur Pajot, Julien Brajard, and Patrick Gallinari (2019). "Learning dynamical systems from partial observations". In: *arXiv preprint arXiv:1902.11136* (cit. on pp. 36, 78).

Ayed, Ibrahim, Emmanuel de Bézenac, Arthur Pajot, and Patrick Gallinari (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 (cit. on pp. 36, 54, 55, 206).

Babaeizadeh, Mohammad, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine (2018). "Stochastic Variational Video Prediction". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rk49Mg-CW (cit. on p. 28).

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio (Jan. 2015). "Neural machine translation by jointly learning to align and translate". English (US). In: 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015 (cit. on pp. 26, 27).

Behrmann, Jens, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen (Sept. 2019a). "Invertible Residual Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 573–582. URL: http://proceedings.mlr.press/v97/behrmann19a.html (cit. on p. 48).

Behrmann, Jens, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen (June 2019b). "Invertible Residual Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 573–582 (cit. on p. 194).

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 (cit. on p. 90).

Benenti, Sergio (1997). "Intrinsic characterization of the variable separation in the Hamilton-Jacobi equation". In: *Journal of Mathematical Physics* 38.12, pp. 6578–6602 (cit. on p. 57).

Bengio, Y., P. Frasconi, and P. Simard (1993). "The problem of learning long-term dependencies in recurrent networks". In: 1183–1188 vol.3 (cit. on p. 23).

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013a). "Representation Learning: A Review and New Perspectives". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8, pp. 1798–1828 (cit. on p. 55).

Bengio, Yoshua, Nicholas Léonard, and Aaron Courville (Aug. 2013b). "Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation". In: *arXiv:1308.3432 [cs]* (cit. on p. 127).

Bengio, Yoshua, Li Yao, Guillaume Alain, and Pascal Vincent (2013c). "Generalized Denoising Auto-Encoders as Generative Models". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., pp. 899–907 (cit. on p. 42).

Béréziat, Dominique and Isabelle Herlin (2015). "Coupling dynamic equations and satellite images for modelling ocean surface circulation". In: *Communications in Computer and Information Science* 550, pp. 191–205. URL: https://hal.inria.fr/hal-01245369 (cit. on pp. 31, 97).

Bertinetto, Luca, Joao F. Henriques, Philip Torr, and Andrea Vedaldi (2019). "Meta-learning with differentiable closed-form solvers". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HyxnZh0ct7 (cit. on p. 251).

Bertsekas, Dimitri P. (1996). *Constrained Optimization and Lagrange Multiplier Methods (Optimization and Neural Computation Series)*. 1st ed. Athena Scientific (cit. on p. 79).

Bézenac, Emmanuel de, Arthur Pajot, and Patrick Gallinari (2018a). "Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge". In: *International Conference on Learning Representations* (cit. on pp. 40, 55, 63, 196, 199, 200).

Bézenac, Emmanuel de, Arthur Pajot, and Patrick Gallinari (2018b). "Deep learning for physical processes: Incorporating prior scientific knowledge". In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 11, 73).

Boffetta, Guido, G Lacorata, G Redaelli, and A Vulpiani (2001). "Detecting barriers to transport: a review of different techniques". In: *Physica D: Nonlinear Phenomena* 159.1-2, pp. 58–70 (cit. on pp. 98, 239).

Bora, Ashish, Ajil Jalal, Eric Price, and Alexandros G. Dimakis (June 2017). "Compressed Sensing using Generative Models". 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. International Convention Centre, Sydney, Australia: PMLR, pp. 537–546 (cit. on p. 129).

Bora, Ashish, Eric Price, and Alexandros G. Dimakis (2018). "AmbientGAN: Generative models from lossy measurements". In: *International Conference on*

Learning Representations. URL: https://openreview.net/forum?id=Hy7fDog0b (cit. on p. 46).

Boyd, Stephen, Stephen P Boyd, and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge university press (cit. on p. 92).

Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein (Jan. 2011). "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Found. Trends Mach. Learn.* 3.1, pp. 1–122. URL: https://doi.org/10.1561/2200000016 (cit. on p. 92).

Brunton, Steven L., Joshua L. Proctor, and J. Nathan Kutz (2016). "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". In: *Proceedings of the National Academy of Sciences* 113.15, pp. 3932–3937 (cit. on pp. 38, 55, 73, 78).

Bungartz, Hans-Joachim and Michael Griebel (2004). "Sparse grids". In: *Acta Numerica* 13, pp. 147–269 (cit. on pp. 56, 58).

Butcher, John Charles and Nicolette Goodwin (2008). *Numerical methods for ordinary differential equations*. Vol. 2. Wiley Online Library (cit. on p. 30).

Cai, Mengmeng, Manisa Pipattanasomporn, and Saifur Rahman (2019). "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques". In: *Applied Energy* 236, pp. 1078–1088. URL: https://www.sciencedirect.com/science/article/pii/S0306261918318609 (cit. on p. 27).

Carrassi, Alberto, Marc Bocquet, Laurent Bertino, and Geir Evensen (2018). "Data assimilation in the geosciences: An overview of methods, issues, and perspectives". In: *WIREs Climate Change* 9.5, e535. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wcc.535. URL: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wcc.535 (cit. on p. 31).

Cartan, H. and J. Kouneiher (1967). *Cours de calcul différentiel*. Collection Méthodes. Editions Hermann (cit. on p. 30).

Caruana, Rich (1997). "Multitask Learning". In: *Machine Learning* 28.1, pp. 41–75 (cit. on pp. 121, 251).

Che, Zhengping, S. Purushotham, Kyunghyun Cho, D. Sontag, and Y. Liu (2018). "Recurrent Neural Networks for Multivariate Time Series with Missing Values". In: *Scientific Reports* 8 (cit. on p. 27).

Chen, Changyou, Chunyuan Li, Liquan Chen, Wenlin Wang, Yunchen Pu, and Lawrence Carin (2018). "Continuous-Time Flows for Efficient Inference and Density Estimation". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 823–832. URL: http://proceedings.mlr.press/v80/chen18d.html (cit. on p. 49).

Chen, Ricky T. Q. (2021). *torchdiffeq*. Version 0.2.2. URL: https://github.com/rtqichen/torchdiffeq (cit. on p. 113).

Chen, Ricky T. Q., Yulia Rubanova, Jesse Bettencourt, and David Duvenaud (2018). "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems 31*. Ed. by Samy Bengio, Hanna Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. Curran Associates, Inc., pp. 6571–6583 (cit. on pp. 29, 33, 35, 48, 55, 60, 76, 82, 92, 94, 146, 193, 201, 217, 218, 223–225, 245, 246).

Chen, Xi, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel (2016a). "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, pp. 2172–2180 (cit. on p. 194).

Chen, Xi, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel (2016b). "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett. Vol. 29. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2016/file/7c9d0b1f96aebd7b5eca8c3edaa19ebb-Paper.pdf (cit. on p. 46).

Chen, Yutian, Abram L Friesen, Feryal Behbahani, Arnaud Doucet, David Budden, Matthew Hoffman, and Nando de Freitas (2020). "Modular Meta-Learning with Shrinkage". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 2858–2869 (cit. on p. 118).

Chen, Zhengdao, Jianyu Zhang, Martin Arjovsky, and Léon Bottou (2020a). "Symplectic Recurrent Neural Networks". In: *International Conference on Learning Representations* (cit. on pp. 55, 194).

Chen, Zhengdao, Jianyu Zhang, Martin Arjovsky, and Léon Bottou (2020b). "Symplectic Recurrent Neural Networks". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 73).

Chen, Zhengdao, Jianyu Zhang, Martin Arjovsky, and Léon Bottou (2020c). "Symplectic Recurrent Neural Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=BkgYPREtPr (cit. on p. 103).

Cho, Kyunghyun, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (Oct. 2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natu-*

ral Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, pp. 1724–1734 (cit. on pp. 25, 66).

Chu, Mengyu and Nils Thuerey (July 2017). "Data-driven synthesis of smoke flows with CNN-based feature descriptors". In: *ACM Transactions on Graphics* 36.4, pp. 1–14. URL: http://dx.doi.org/10.1145/3072959.3073643 (cit. on p. 46).

Clavera, Ignasi, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn (2019). "Learning to Adapt in Dynamic, Real-World Environments through Meta-Reinforcement Learning". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HyztsoC5Y7 (cit. on p. 105).

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 (cit. on p. 104).

Cranmer, Miles, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho (2020). "Lagrangian Neural Networks". In: *ICLR 2020 Deep Differential Equations Workshop* (cit. on pp. 39, 78, 220).

Cybenko, G. (Dec. 1989). "Approximation by superpositions of a sigmoidal function". In: *Mathematics of Control, Signals, and Systems (MCSS)* 2.4, pp. 303–314. URL: http://dx.doi.org/10.1007/BF02551274 (cit. on p. 187).

Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov (2019). "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context". In: *ACL* (cit. on p. 27).

Daniels, Bryan C. and Ilya Nemenman (Mar. 2015). "Efficient Inference of Parsimonious Phenomenological Models of Cellular Dynamics Using S-Systems and Alternating Regression". In: *PLOS ONE* 10.3, pp. 1–14 (cit. on pp. 115, 259, 260).

de Bézenac, Emmanuel, Arthur Pajot, and Patrick Gallinari (2018c). "Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=By4HsfWAZ (cit. on p. 104).

Deng, Cai, Zhang Chiyuan, and He Xiaofei (2010). "Unsupervised Feature Selection for Multi-cluster Data". In: 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'10)}.2010 (cit. on p. 128).

Deng, Ruizhi, Bo Chang, Marcus A. Brubaker, Greg Mori, and Andreas M. Lehrmann (2020). "Modeling Continuous Stochastic Processes with Dynamic Normalizing Flows". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien

Lin. URL: https://proceedings.neurips.cc/paper/2020/hash/58c54802a9fb9526cd0923353a34a7ae-Abstract.html (cit. on p. 49).

Denton, Emily and Vighnesh Birodkar (2017). "Unsupervised Learning of Disentangled Representations from Video". In: *Advances in Neural Information Processing Systems 30*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Curran Associates, Inc., pp. 4414–4423 (cit. on pp. 54, 55, 65, 67, 192, 197–199).

Denton, Emily and Rob Fergus (July 2018). "Stochastic Video Generation with a Learned Prior". 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. 1174–1183 (cit. on pp. 28, 62, 63, 65, 68, 199, 203–205).

Diamantakis, Michail (2014). "The semi-Lagrangian technique in atmospheric modelling: current status and future challenges". In: *Seminar on Recent Developments in Numerical Methods for Atmosphere and Ocean Modelling, 2-5 September 2013*. ECMWF. Shinfield Park, Reading: ECMWF, pp. 183–200. URL: https://www.ecmwf.int/node/9054 (cit. on p. 241).

Dinh, Laurent, David Krueger, and Yoshua Bengio (2015). *NICE: Non-linear Independent Components Estimation*. arXiv: 1410.8516 [cs.LG] (cit. on p. 47).

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio (2017). "Density estimation using Real NVP". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. URL: https://openreview.net/forum?id=HkpbnH9lx (cit. on p. 47).

Dona, Jérémie and Patrick Gallinari (2021). "Differentiable Feature Selection, a Reparameterization Approach". In: *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. European Conference on Machine Learning, Principles, and Practice of Knowledge Discovery in Databases. Vienne, Austria (cit. on p. 125).

Donà, Jérémie, Marie Déchelle, Patrick Gallinari, and Marina Lévy (2022). "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction". In: *The Tenth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=gbe1zHyA73 (cit. on pp. 13, 71).

Donà, Jérémie, Jean-Yves Franceschi, Sylvain Lamprier, and Patrick Gallinari (May 2021). "PDE-Driven Spatiotemporal Disentanglement". In: *The Ninth International Conference on Learning Representations*. International Conference on Representation Learning. Vienne, Austria. URL: https://hal.archives-ouvertes.fr/hal-02911067 (cit. on pp. 10, 53).

Donahue, Jeff, Philipp Krähenbühl, and Trevor Darrell (2017). "Adversarial Feature Learning". In: *5th International Conference on Learning Representations, ICLR*

2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net. URL: https://openreview.net/forum?id=BJtNZAFgg (cit. on p. 46).

Donoho, David L. (2006). "Compressed sensing". In: *IEEE Trans. Inform. Theory* 52, pp. 1289–1306 (cit. on p. 129).

Dormand, John R and Peter J Prince (1980). "A family of embedded Runge-Kutta formulae". In: *Journal of computational and applied mathematics* 6.1, pp. 19–26 (cit. on p. 216).

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby (2021). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=YicbFdNTTy (cit. on p. 27).

Dosovitskiy, Alexey, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox (Dec. 2015). "FlowNet: Learning Optical Flow With Convolutional Networks". In: *The IEEE International Conference on Computer Vision (ICCV)*, pp. 2758–2766 (cit. on p. 54).

Doucet, Arnaud, Simon Godsill, and Christophe Andrieu (July 2000). "On Sequential Monte Carlo Sampling Methods for Bayesian Filtering". In: 10.3, pp. 197–208. URL: https://doi.org/10.1023/A:1008935410038 (cit. on p. 21).

Droniou, Jérôme (Apr. 2001). "Intégration et Espaces de Sobolev à Valeurs Vectorielles." working paper or preprint. URL: https://hal.archives-ouvertes.fr/hal-01382368 (cit. on p. 232).

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 (cit. on p. 73).

Dupont, Emilien, Arnaud Doucet, and Yee Whye Teh (2019). "Augmented Neural ODEs". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/21be9a4bd4f81549a9d1d241981cec3c-Paper.pdf (cit. on p. 36).

Duraisamy, Karthik, Gianluca Iaccarino, and Heng Xiao (2019). "Turbulence modeling in the age of data". In: *Annual Review of Fluid Mechanics* 51, pp. 357–377 (cit. on p. 104).

Eliasof, Moshe, Eldad Haber, and Eran Treister (2021). "PDE-GCN: Novel Architectures for Graph Neural Networks Motivated by Partial Differential Equations". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y.

Dauphin, P. Liang, and J. Wortman Vaughan. URL: https://openreview.net/forum?id=wWtk6GxJB2x (cit. on p. 145).

Engle, Robert F. (1982). "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation". In: *Econometrica* 50.4, pp. 987–1007. URL: http://www.jstor.org/stable/1912773 (cit. on p. 19).

Figurnov, Mikhail, Shakir Mohamed, and Andriy Mnih (2018). "Implicit Reparameterization Gradients". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc. (cit. on p. 128).

Finlay, Chris, Joern-Henrik Jacobsen, Levon Nurbekyan, and Adam Oberman (13–18 Jul 2020). "How to Train Your Neural ODE: the World of Jacobian and Kinetic Regularization". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 3154–3164. URL: https://proceedings.mlr.press/v119/finlay20a.html (cit. on p. 36).

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (June 2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep 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. 1126–1135. URL: http://proceedings.mlr.press/v70/finn17a.html (cit. on pp. 104, 116, 120, 251).

Finn, Chelsea, Ian Goodfellow, and Sergey Levine (2016). "Unsupervised Learning for Physical Interaction through Video Prediction". In: *Advances in Neural Information Processing Systems 29*. Ed. by Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett. Curran Associates, Inc., pp. 64–72 (cit. on p. 54).

Finzi, Marc, Max Welling, and Andrew Gordon Wilson (2021). "A Practical Method for Constructing Equivariant Multilayer Perceptrons for Arbitrary Matrix Groups". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 3318–3328. URL: http://proceedings.mlr.press/v139/finzi21a.html (cit. on p. 146).

Flennerhag, Sebastian, Andrei A. Rusu, Razvan Pascanu, Francesco Visin, Hujun Yin, and Raia Hadsell (2020). "Meta-Learning with Warped Gradient Descent". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rkeiQlBFPB (cit. on p. 121).

Fletcher, James and Warren Moors (Apr. 2014). "Chebyshev Sets". In: *Journal of the Australian Mathematical Society* 98, pp. 161–231 (cit. on p. 211).

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. URL: https://www.sciencedirect.com/science/article/pii/S1474667017429387 (cit. on p. 89).

Fourier, Jean Baptiste Joseph (1822). *Théorie analytique de la chaleur*. Didot, Firmin (cit. on p. 56).

Fraccaro, Marco, Simon Kamronn, Ulrich Paquet, and Ole Winther (2017). "A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning". In: *Advances in Neural Information Processing Systems 30*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Curran Associates, Inc., pp. 3601–3610 (cit. on p. 55).

Franceschi, Jean-Yves, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari (2020). "Stochastic Latent Residual Video Prediction". In: *arXiv preprint arXiv:2002.09219* (cit. on pp. 54, 55, 197).

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 (cit. on pp. 98, 239).

Frankle, Jonathan and Michael Carbin (2019). "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rJl-b3RcF7 (cit. on p. 112).

Fresca, Stefania, Andrea Manzoni, Luca Dedè, and Alfio Quarteroni (Oct. 2020). "Deep learning-based reduced order models in cardiac electrophysiology". In: *PloS one* 15.10, e0239416–e0239416. URL: https://pubmed.ncbi.nlm.nih.gov/33002014 (cit. on p. 104).

Fuller, Wayne A. (1976). *Introduction to statistical time series*. A Wiley publication in applied statistics. New York [u.a.]: Wiley. IX, 470 (cit. on p. 19).

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. URL: https://www.sciencedirect.com/science/article/pii/S089360800580125X (cit. on p. 27).

Garnelo, Marta, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Jimenez Rezende, and S. M. Ali Eslami (2018). "Conditional Neural Processes". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 1690–1699.

URL: http://proceedings.mlr.press/v80/garnelo18a.html (cit. on pp. 112, 121).

Gaultier, Lucile, Jacques Verron, Jean-Michel Brankart, Olivier Titaud, and Pierre Brasseur (2013). "On the inversion of submesoscale tracer fields to estimate the surface ocean circulation". In: *Journal of Marine Systems* 126, pp. 33–42 (cit. on p. 243).

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. URL: https://www.sciencedirect.com/science/article/pii/S0021999119307612 (cit. on p. 38).

Geneva, Nicholas and Nicholas Zabaras (2021). *Transformers for Modeling Physical Systems*. arXiv: 2010.03957 [cs.LG] (cit. on p. 27).

Gholaminejad, Amir, Kurt Keutzer, and George Biros (2019). "ANODE: Unconditionally Accurate Memory-Efficient Gradients for Neural ODEs". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by Sarit Kraus. ijcai.org, pp. 730–736. URL: https://doi.org/10.24963/ijcai.2019/103 (cit. on p. 35).

Gomez, Aidan N, Mengye Ren, Raquel Urtasun, and Roger B Grosse (2017). "The Reversible Residual Network: Backpropagation Without Storing Activations". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/f9be311e65d81a9ad8150a60844bb94c-Paper.pdf (cit. on p. 48).

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger. Curran Associates, Inc., pp. 2672–2680 (cit. on pp. 45, 55).

Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (June 2014). "Generative Adversarial Networks". In: *arXiv:1406.2661 [cs, stat]* (cit. on p. 274).

Goyal, Anirudh, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio (2016). "Professor Forcing: A New Algorithm for Training Recurrent Networks". In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., pp. 4608–4616 (cit. on p. 113).

Grathwohl, Will, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud (2019). "Scalable Reversible Generative Models with Free-form Continuous Dynamics". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rJxgknCcK7 (cit. on p. 48).

Gregor, Karol, George Papamakarios, Frederic Besse, Lars Buesing, and Theophane Weber (2019). "Temporal Difference Variational Auto-Encoder". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=S1x4ghC9tQ (cit. on p. 45).

Greydanus, Samuel, Misko Dzamba, and Jason Yosinski (2019). "Hamiltonian Neural Networks". In: *Advances in Neural Information Processing Systems 32*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Curran Associates, Inc., pp. 15379–15389 (cit. on pp. 39, 53, 55, 73, 78, 81, 82, 95, 103, 216, 217, 220, 225).

Gruver, Nate, Marc Anton Finzi, Samuel Don Stanton, and Andrew Gordon Wilson (2022). "Deconstructing the Inductive Biases of Hamiltonian Neural Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=EDeVYpT42oS (cit. on p. 145).

Gulrajani, Ishaan, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville (2017). "Improved Training of Wasserstein GANs". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., pp. 5769–5779 (cit. on p. 46).

Gur-Ari, Guy, Daniel A. Roberts, and Ethan Dyer (2019). *Gradient Descent Happens in a Tiny Subspace*. URL: https://openreview.net/forum?id=ByeTHsAqtX (cit. on p. 111).

Guyon, Isabelle and André Elisseeff (Mar. 2003). "An Introduction to Variable and Feature Selection". In: *J. Mach. Learn. Res.* 3, pp. 1157–1182 (cit. on pp. 126, 128).

Ha, David, Andrew Dai, and Quoc V. Le (2017a). "HyperNetworks". In: *International Conference on Learning Representation*. URL: https://openreview.net/pdf?id=rkpACe1lx (cit. on p. 133).

Ha, David, Andrew M. Dai, and Quoc V. Le (2017b). "HyperNetworks". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. URL: https://openreview.net/forum?id=rkpACe1lx (cit. on p. 109).

Haber, Eldad and Lars Ruthotto (Dec. 2017). "Stable architectures for deep neural networks". In: *Inverse Problems* 34.1, p. 014004 (cit. on pp. 36, 146, 193).

Haeberli, Wilfried, Martin Hoelzle, Frank Paul, and Michael Zemp (2007). "Integrated monitoring of mountain glaciers as key indicators of global climate change: the European Alps". en. In: *Annals of Glaciology* 46, pp. 150–160 (cit. on p. 126).

Hairer, E., S. P. Nørsett, and G. Wanner (2000). *Solving Ordinary Differential Equations I: Nonstiff problems*. Second. Berlin: Springer (cit. on p. 113).

Hairer, Ernst, Syvert P. Nørsett, and Gerhard Wanner (1993). "Solving Ordinary Differential Equations I: Nonstiff Problems". In: Berlin, Heidelberg: Springer Berlin Heidelberg. Chap. Runge-Kutta and Extrapolation Methods, pp. 129–353 (cit. on p. 196).

Hamilton, William Rowan (1835). "Second essay on a general method in dynamics". In: *Philosophical Transactions of the Royal Society* 125, pp. 95–144 (cit. on pp. 54, 56).

Han, Kai, Yunhe Wang, Chao Zhang, Chao Li, and Chao Xu (Oct. 2017). "AutoEncoder Inspired Unsupervised Feature Selection". In: *arXiv:1710.08310 [cs, stat]*. arXiv: 1710.08310 (cit. on pp. 126, 128).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778 (cit. on p. 137).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (June 2016). "Deep Residual Learning for Image Recognition". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778 (cit. on pp. 29, 32, 35, 48, 201).

He, Xiaofei, Deng Cai, and Partha Niyogi (2006). "Laplacian Score for Feature Selection". In: *Advances in Neural Information Processing Systems 18*. Ed. by Y. Weiss, B. Schölkopf, and J. C. Platt. MIT Press, pp. 507–514 (cit. on p. 128).

Hewamalage, Hansika, Christoph Bergmeir, and Kasun Bandara (2021). "Recurrent Neural Networks for Time Series Forecasting: Current status and future directions". In: *International Journal of Forecasting* 37.1, pp. 388–427. URL: https://www.sciencedirect.com/science/article/pii/S0169207020300996 (cit. on p. 27).

Higgins, Irina, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner (2017). "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *International Conference on Learning Representations* (cit. on p. 44).

Hinton, G. E. and R. R. Salakhutdinov (2006). "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313.5786, pp. 504–507. eprint: https://www.science.org/doi/pdf/10.1126/science.1127647 (cit. on p. 41).

Hinton, G.E. and R.R. Salakhutdinov (Aug. 2006). "Reducing the Dimensionality of Data with Neural Networks". In: *Science (New York, N.Y.)* 313, pp. 504–7 (cit. on p. 128).

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780 (cit. on p. 25).

Horn, Berthold K. P. and Brian G. Schunck (Aug. 1981). "Determining Optical Flow". In: *Artificial Intelligence* 17.1–3, pp. 185–203 (cit. on p. 54).

Hou, Chenping, Feiping Nie, Xuelong Li, Dongyun Yi, and Yi Wu (June 2014). "Joint Embedding Learning and Sparse Regression: A Framework for Unsupervised Feature Selection". In: *IEEE Transactions on Cybernetics* 44.6, pp. 793–804 (cit. on p. 128).

Hsieh, Jun-Ting, Bingbin Liu, De-An Huang, Li Fei-Fei, and Juan Carlos Niebles (2018). "Learning to Decompose and Disentangle Representations for Video Prediction". In: *Advances in Neural Information Processing Systems 31*. Ed. by Samy Bengio, Hanna Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. Curran Associates, Inc., pp. 517–526 (cit. on pp. 55, 65, 199).

Hsu, Wei-Ning, Yu Zhang, and James Glass (2017). "Unsupervised Learning of Disentangled and Interpretable Representations from Sequential Data". In: *Advances in Neural Information Processing Systems 30*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Curran Associates, Inc., pp. 1878–1889 (cit. on p. 55).

IPSL (2018). *IPSL CMA5.2 simulation*. URL: http://forge.ipsl.jussieu.fr/igcmg%5C_doc/wiki/DocHconfigAipslcm5a2 (cit. on p. 136).

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros (2017a). "Image-to-Image Translation with Conditional Adversarial Networks". In: *CVPR* (cit. on p. 243).

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros (Nov. 2016). "Image-to-Image Translation with Conditional Adversarial Networks". In: *arXiv:1611.07004 [cs]*. arXiv: 1611.07004 (cit. on pp. 137, 140, 270, 273, 274).

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros (2017b). "Image-to-Image Translation with Conditional Adversarial Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976 (cit. on p. 46).

Jacobsen, Jörn-Henrik, Arnold W.M. Smeulders, and Edouard Oyallon (2018). "i-RevNet: Deep Invertible Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HJsjkMb0Z (cit. on p. 48).

Jaderberg, Max, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu koray (2015). "Spatial Transformer Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2015/file/33ceb07bf4eeb3da587e268d663aba1a-Paper.pdf (cit. on pp. 97, 241).

Jang, Eric, Shixiang Gu, and Ben Poole (Apr. 2017). "Categorical Reparametrization with Gumbel-Softmax". In: *Proceedings International Conference on Learning Representations 2017*. OpenReviews.net (cit. on p. 128).

Jaques, Miguel, Michael Burke, and Timothy Hospedales (2020). "Physics-as-Inverse-Graphics: Unsupervised Physical Parameter Estimation from Video". In: *International Conference on Learning Representations* (cit. on p. 55).

Jia, Huabing, Wei Xu, Xiaoshan Zhao, and Zhanguo Li (Mar. 2008). "Separation of variables and exact solutions to nonlinear diffusion equations with x-dependent convection and absorption". In: *Journal of Mathematical Analysis and Applications* 339.2, pp. 982–995 (cit. on p. 57).

Jia, Xiaowei, Jared Willard, Anuj Karpatne, Jordan Read, Jacob Zwart, Michael S Steinbach, and Vipin Kumar (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 (cit. on pp. 39, 87, 88, 90, 245).

Jia, Xiaowei, Jared Willard, Anuj Karpatne, Jordan S. Read, Jacob A. Zwart, Michael Steinbach, and Vipin Kumar (May 2021). "Physics-Guided Machine Learning for Scientific Discovery: An Application in Simulating Lake Temperature Profiles". In: *ACM/IMS Trans. Data Sci.* 2.3. URL: https://doi.org/10.1145/3447814 (cit. on p. 27).

Johnson, Gordon G (1987). "A nonconvex set which has the unique nearest point property". In: *Journal of Approximation Theory* 51.4, pp. 289–332 (cit. on p. 211).

Kalman, R. E. (Mar. 1960). "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1, pp. 35–45. eprint: https://asmedigitalcollection.asme.org/fluidsengineering/article-pdf/82/1/35/5518977/35_1.pdf. URL: https://doi.org/10.1115/1.3662552 (cit. on p. 104).

Kalnins, E. G., Willard Miller Jr., and G. C. Williams (1992). "Recent Advances in the Use of Separation of Variables Methods in General Relativity". In: *Philosophical Transactions: Physical Sciences and Engineering* 340.1658, pp. 337–352 (cit. on p. 57).

Karl, Maximilian, Maximilian Sölch, Justin Bayer, and Patrick van der Smagt (2017). "Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data". In: *International Conference on Learning Representations*. Vol. abs/1605.06432 (cit. on p. 45).

Karras, Tero, Samuli Laine, and Timo Aila (2019). *A Style-Based Generator Architecture for Generative Adversarial Networks*. arXiv: 1812.04948 [cs.NE] (cit. on p. 133).

Kashinath, Karthik, Adrian Albert, Dragos Chirila, Mr Prabhat, and Heng Xiao (Dec. 2019). "Enforcing statistical constraints in generative adversarial networks for modeling chaotic dynamical systems". In: *Journal of Computational Physics* 406, p. 109209 (cit. on p. 46).

Kim, Hyunjik, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh (2019). "Attentive Neural Processes". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SkE6PjC9KX (cit. on p. 112).

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1412.6980 (cit. on pp. 116, 202, 261).

Kingma, Diederik P. and Max Welling (Dec. 2013). "Auto-Encoding Variational Bayes". In: *arXiv:1312.6114 [cs, stat]*. arXiv: 1312.6114 (cit. on p. 127).

Kingma, Diederik P. and Max Welling (2014a). "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1312.6114 (cit. on p. 42).

Kingma, Diederik P. and Max Welling (2014b). "Auto-Encoding Variational Bayes". In: *International Conference on Learning Representations* (cit. on p. 55).

Kingma, Diederik P. and Max Welling (2019) (cit. on p. 44).

Kingma, Durk P and Prafulla Dhariwal (2018). "Glow: Generative Flow with Invertible 1x1 Convolutions". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2018/file/d139db6a236200b21cc7f752979132d0-Paper.pdf (cit. on p. 47).

Kingma, Durk P, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling (2016). "Improved Variational Inference with Inverse Autoregressive Flow". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett. Vol. 29. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2016/file/ddeebdeefdb7e7e7a697e1c3e3d8ef54-Paper.pdf (cit. on p. 47).

Kirchmeyer, Matthieu, Yuan Yin, Jérémie Donà, Nicolas Baskiotis, Alain Rakotomamonjy, and Patrick Gallinari (Jan. 2022). "Generalizing to New Physical Systems via Context-Informed Dynamics Model". working paper or preprint. URL: https://hal.archives-ouvertes.fr/hal-03547546 (cit. on pp. 14, 103).

Klaasen, Gene A. and William C. Troy (1984). "Stationary Wave Solutions of a System of Reaction-Diffusion Equations Derived from the FitzHugh–Nagumo Equations". In: *SIAM Journal on Applied Mathematics* 44.1, pp. 96–110 (cit. on pp. 80, 217).

Kobyzev, I., S. D. Prince, and M. A. Brubaker (Nov. 2021). "Normalizing Flows: An Introduction and Review of Current Methods". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11, pp. 3964–3979 (cit. on p. 47).

Kochkov, Dmitrii, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer (2021). "Machine learning accelerated computational fluid dynamics". In: *Proceedings of the National Academy of Sciences* 118 (21). URL: https://www.pnas.org/content/118/21/e2101784118 (cit. on pp. 103, 104).

Kočiský, Tomáš, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann (Sept. 2016). "Semantic Parsing with Semi-Supervised Sequential Autoencoders". In: *arXiv:1609.09315 [cs]*. arXiv: 1609.09315 (cit. on pp. 128, 131).

Koller, Daphne and Mehran Sahami (Feb. 1996). *Toward Optimal Feature Selection*. Techreport (cit. on p. 128).

Kosiorek, Adam R., Hyunjik Kim, Yee Whye Teh, and Ingmar Posner (2018). "Sequential Attend, Infer, Repeat: Generative Modelling of Moving Objects". In: *Advances in Neural Information Processing Systems 31*. Ed. by Samy Bengio, Hanna Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. Curran Associates, Inc., pp. 8606–8616 (cit. on p. 55).

Kraskov, Alexander, Harald Stögbauer, and Peter Grassberger (June 2004). "Estimating mutual information". In: *Physical Review E* 69 (6), p. 066138 (cit. on p. 194).

Krishnapriyan, Aditi, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W. Mahoney (2021). "Characterizing possible failure modes in physics-informed neural networks". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan. URL: https://openreview.net/forum?id=a2Gr9gNFD-J (cit. on pp. 38, 146).

Krueger, David, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville (2021). *Out-of-Distribution Generalization via Risk Extrapolation (REx)*. arXiv: 2003.00688 [cs.LG]. URL: https://arxiv.org/pdf/2003.00688.pdf (cit. on p. 120).

Kutta, W. (1901). "Beitrag zur näherungsweisen Integration totaler Differentialgleichungen". In: *Zeit. Math. Phys.* 46, pp. 435–53 (cit. on p. 196).

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. URL: http://proceedings.mlr.press/v97/lange19a.html (cit. on p. 92).

Le Dret, Hervé and Brigitte Lucquin (2016). "Partial Differential Equations: Modeling, Analysis and Numerical Approximation". In: Cham: Springer International Publishing. Chap. The Heat Equation, pp. 219–251 (cit. on pp. 56, 190).

Le Guen, Vincent and Nicolas Thome (June 2020). "Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction". In: *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11474–11484 (cit. on pp. 55, 62, 63, 68, 198, 199, 220).

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (Nov. 1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324 (cit. on pp. 64, 197).

Lee, Alex X., Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine (2018). "Stochastic Adversarial Video Prediction". In: *arXiv preprint arXiv:1804.01523* (cit. on p. 204).

Lee, Alex X., Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine (2019). *Stochastic Adversarial Video Prediction*. URL: https://openreview.net/forum?id=HyEl3o05Fm (cit. on p. 28).

Lee, Kimin, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin (13–18 Jul 2020). "Context-aware Dynamics Model for Generalization in Model-Based Reinforcement Learning". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 5757–5766. URL: http://proceedings.mlr.press/v119/lee20g.html (cit. on p. 105).

Lee, Kwonjoon, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto (2019). "Meta-Learning With Differentiable Convex Optimization". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, pp. 10657–10665. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Lee%5C_Meta-Learning%5C_With%5C_Differentiable%5C_Convex%5C_Optimization%5C_CVPR%5C_2019%5C_paper.html (cit. on p. 251).

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 (cit. on p. 39).

Lee, Yoonho and Seungjin Choi (2018). "Gradient-based meta-learning with learned layerwise metric and subspace". In: *International Conference on Machine Learning*, pp. 2933–2942 (cit. on p. 120).

Lewis, Adrian S. and Jérôme Malick (2008). "Alternating Projections on Manifolds". In: *Mathematics of Operations Research* 33.1, pp. 216–234 (cit. on p. 92).

Li, Chunyuan, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski (2018). "Measuring the Intrinsic Dimension of Objective Landscapes". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=ryup8-WCW (cit. on p. 111).

Li, Hao, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein (2018). "Visualizing the Loss Landscape of Neural Nets". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc. (cit. on pp. 110, 111).

Li, Xuechen, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud (2020). "Scalable Gradients for Stochastic Differential Equations". In: *arXiv preprint arXiv:2001.01328* (cit. on p. 55).

Li, Zhenguo, Fengwei Zhou, Fei Chen, and Hang Li (2017). "Meta-SGD: Learning to Learn Quickly for Few Shot Learning". In: *CoRR* abs/1707.09835. arXiv: 1707.09835. URL: http://arxiv.org/abs/1707.09835 (cit. on pp. 116, 121).

Li, Zongyi, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar (2021). "Fourier Neural Operator for Parametric Partial Differential Equations". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=c8P9NQVtmnO (cit. on pp. 103, 104, 116, 261).

Lienen, Marten and Stephan Günnemann (2022). "Learning the Dynamics of Physical Systems from Sparse Observations with Finite Element Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HFmAukZ-k-2 (cit. on p. 145).

Linial, Ori, Neta Ravid, Danny Eytan, and Uri Shalit (2021). "Generative ODE modeling with known unknowns". In: *Proceedings of the Conference on Health, Inference, and Learning*, pp. 79–94 (cit. on pp. 39, 45, 87, 88, 90).

Liu, Zhijian, Jiajun Wu, Zhenjia Xu, Chen Sun, Kevin Murphy, William T. Freeman, and Joshua B. Tenenbaum (2019). "Modeling Parts, Structure, and System Dynamics via Predictive Learning". In: *International Conference on Learning Representations* (cit. on p. 55).

Ljung, Lennart (1998). "System Identification". In: *Signal Analysis and Prediction*. Ed. by Ales Procházka, Jan Uhlíř, P. W. J. Rayner, and N. G. Kingsbury. Boston, MA: Birkhäuser Boston. URL: https://doi.org/10.1007/978-1-4612-1768-8_11 (cit. on p. 31).

Locatello, Francesco, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem (June 2019). "Challenging Common

Assumptions in the Unsupervised Learning of Disentangled Representations". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 4114–4124 (cit. on p. 55).

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 (cit. on pp. 39, 55).

Long, Zichao, Yiping Lu, Xianzhong Ma, and Bin Dong (2018). "PDE-Net: Learning PDEs from Data". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 3214–3222. URL: http://proceedings.mlr.press/v80/long18a.html (cit. on pp. 39, 53, 55, 73, 111, 206).

Lorenz, Edward N. (1969). "Atmospheric Predictability as Revealed by Naturally Occurring Analogues". In: *Journal of Atmospheric Sciences* 26.4, pp. 636–646. URL: https://journals.ametsoc.org/view/journals/atsc/26/4/1520-0469_1969_26_636_aparbn_2_0_co_2.xml (cit. on p. 41).

Lotka, A.J. (1925). "Elements of Physical Biology". In: *Nature* 116.2917, pp. 461–461 (cit. on pp. 115, 259).

Louizos, Christos, Max Welling, and Diederik P. Kingma (Dec. 2017). "Learning Sparse Neural Networks through L_0 Regularization". In: *arXiv:1712.01312 [cs, stat]*. arXiv: 1712.01312 (cit. on p. 134).

Lu, Peter Y., Samuel Kim, and Marin Solja či ć (Sept. 2020). "Extracting Interpretable Physical Parameters from Spatiotemporal Systems Using Unsupervised Learning". In: *Phys. Rev. X* 10 (3), p. 031056 (cit. on p. 45).

Lu, Yiping, Aoxiao Zhong, Quanzheng Li, and Bin Dong (Oct. 2018). "Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 3276–3285. URL: https://proceedings.mlr.press/v80/lu18d.html (cit. on pp. 36, 201).

Luce, R. Duncan (1959). *Individual choice behavior*. Individual choice behavior. Oxford, England: John Wiley (cit. on p. 128).

Luong, Thang, Hieu Pham, and Christopher D. Manning (Sept. 2015). "Effective Approaches to Attention-based Neural Machine Translation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 1412–1421. URL: https://aclanthology.org/D15-1166 (cit. on p. 27).

Lusch, Bethany, J. Nathan Kutz, and Steven L. Brunton (2018). "Deep learning for universal linear embeddings of nonlinear dynamics". In: *Nature Commu-*

nications 9.1, p. 4950. URL: https://doi.org/10.1038/s41467-018-07210-0 (cit. on p. 42).

Maddison, Chris J., Andriy Mnih, and Yee Whye Teh (Nov. 2016). "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: *arXiv:1611.00712 [cs, stat]*. arXiv: 1611.00712 (cit. on pp. 128, 129, 137, 268–271).

Madec, Gurvan (2008). *NEMO ocean engine*. Note du Pôle de modélisation, Institut Pierre-Simon Laplace (IPSL), France, No 27 (cit. on pp. 63, 238).

Makhzani, Alireza and Brendan Frey (Dec. 2013). "k-Sparse Autoencoders". In: *arXiv:1312.5663 [cs]*. arXiv: 1312.5663 (cit. on p. 128).

Makhzani, Alireza, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow (2016). "Adversarial Autoencoders". In: *International Conference on Learning Representations*. URL: http://arxiv.org/abs/1511.05644 (cit. on p. 44).

Maldonado, Sebastián and Richard Weber (June 2009). "A wrapper method for feature selection using Support Vector Machines". In: *Information Sciences*. Special Section on High Order Fuzzy Sets 179.13, pp. 2208–2217 (cit. on p. 128).

Manohar, Krithika, Bingni W. Brunton, J. Nathan Kutz, and Steven L. Brunton (June 2018). "Data-Driven Sparse Sensor Placement for Reconstruction: Demonstrating the Benefits of Exploiting Known Patterns". In: *IEEE Control Systems Magazine* 38.3, pp. 63–86 (cit. on p. 129).

Mao, Xudong, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley (Oct. 2017). "Least Squares Generative Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 46).

Mathieu, Michael, Camille Couprie, and Yann LeCun (Jan. 2016a). "Deep multi-scale video prediction beyond mean square error". English (US). In: 4th International Conference on Learning Representations, ICLR 2016 ; Conference date: 02-05-2016 Through 04-05-2016 (cit. on p. 46).

Mathieu, Michael, Camille Couprie, and Yann LeCun (2016b). "Deep multi-scale video prediction beyond mean square error". In: *International Conference on Learning Representations* (cit. on p. 204).

McPhaden, M. J., G. Meyers, K. Ando, Y. Masumoto, V. S. N. Murty, M. Ravichandran, F. Syamsudin, J. Vialard, L. Yu, and W. Yu (Apr. 2009). "RAMA: The Research Moored Array for African–Asian–Australian Monsoon Analysis and Prediction*". In: *Bulletin of the American Meteorological Society* 90.4, pp. 459–480 (cit. on p. 126).

Mehta, Viraj, Ian Char, Willie Neiswanger, Youngseog Chung, Andrew Oakleigh Nelson, Mark D Boyer, Egemen Kolemen, and Jeff Schneider (2020). "Neural Dynamical Systems: Balancing Structure and Flexibility in Physical Prediction". In: *arXiv preprint arXiv:2006.12682* (cit. on pp. 39, 82, 84, 220, 221).

Micikevicius, Paulius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu (2018). "Mixed Precision Training". In: *International Conference on Learning Representations* (cit. on p. 199).

Miller Jr., Willard (1983). "The technique of variable separation for partial differential equations". In: *Nonlinear Phenomena*. Ed. by Kurt Bernardo Wolf. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 184–208 (cit. on p. 57).

Miller Jr., Willard (1988). "Mechanisms for variable separation in partial differential equations and their relationship to group theory". In: *Symmetries and Nonlinear Phenomena: Proceedings of the International School on Applied Mathematics*. Ed. by Decio Levi and Pavel Winternitz. Singapore: World Scientific, pp. 188–221 (cit. on pp. 54, 57).

Minderer, Matthias, Chen Sun, Ruben Villegas, Forrester Cole, Kevin Murphy, and Honglak Lee (2019). "Unsupervised learning of object structure and dynamics from videos". In: *Advances in Neural Information Processing Systems 32*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Curran Associates, Inc., pp. 92–102 (cit. on p. 55).

Mirza, Mehdi and Simon Osindero (Nov. 2014). "Conditional Generative Adversarial Nets". In: *arXiv:1411.1784 [cs, stat]* (cit. on p. 140).

Mishra, Nikhil, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel (2018). "A Simple Neural Attentive Meta-Learner". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=B1DmUzWAW (cit. on p. 105).

Moore, Andrew M (1991). "Data assimilation in a quasi-geostrophic open-ocean model of the Gulf Stream region using the adjoint method". In: *Journal of Physical Oceanography* 21.3, pp. 398–427 (cit. on p. 12).

Neic, Aurel, Fernando Otaviano Campos, Anton J. Prassl, Steven A. Niederer, Martin J. Bishop, Edward J. Vigmond, and Gernot Plank (2017). "Efficient computation of electrograms and ECGs in human whole heart simulations using a reaction-eikonal model". In: *J. Comput. Phys.* 346, pp. 191–211. URL: https://doi.org/10.1016/j.jcp.2017.06.020 (cit. on p. 104).

Neumann, John von (1950). *Functional Operators (AM-22), Volume 2: The Geometry of Orthogonal Spaces*. Princeton University Press (cit. on p. 92).

Onken, Derek and Lars Ruthotto (2020a). *Discretize-Optimize vs. Optimize-Discretize for Time-Series Regression and Continuous Normalizing Flows*. arXiv: 2005.13420 [cs.LG] (cit. on p. 35).

Onken, Derek and Lars Ruthotto (2020b). "Discretize-optimize vs. optimize-discretize for time-series regression and continuous normalizing flows". In: *arXiv preprint arXiv:2005.13420* (cit. on p. 92).

Oord, Aäron van den, Nal Kalchbrenner, and Koray Kavukcuoglu (2016). "Pixel Recurrent Neural Networks". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Ed. by Maria-Florina Balcan and Kilian Q. Weinberger. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 1747–1756. URL: http://proceedings.mlr.press/v48/oord16.html (cit. on p. 22).

Oreshkin, Boris N., Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio (2020). "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting". In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 224, 225).

Ott, Myle, Sergey Edunov, David Grangier, and Michael Auli (2018). "Scaling Neural Machine Translation". In: arXiv: 1806.00187 [cs.CL] (cit. on p. 27).

Ouala, Said, Ananda Pascual, and Ronan Fablet (2019). "Residual Integration Neural Network". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3622–3626 (cit. on p. 36).

P., Langevin (1908). "Sur la théorie du mouvement brownien". In: *Compte Rendu de l'Académie des Sciences* (cit. on p. 48).

Pan, Feilai, Jun Li, Bendong Tan, Ciling Zeng, Xinfan Jiang, Li Liu, and Jun Yang (2018). "Stacked-GRU Based Power System Transient Stability Assessment Method". In: *Algorithms* 11.8. URL: https://www.mdpi.com/1999-4893/11/8/121 (cit. on p. 24).

Park, Jeong Joon, Peter Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove (2019). "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, pp. 165–174. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Park%5C_DeepSDF%5C_Learning%5C_Continuous%5C_Signed%5C_Distance%5C_Functions%5C_for%5C_Shape%5C_Representation%5C_CVPR%5C_2019%5C_paper.html (cit. on p. 121).

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2013). "On the difficulty of training recurrent neural networks." In: *ICML (3)*. Vol. 28. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 1310–1318 (cit. on p. 23).

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019a). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer,

Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Curran Associates, Inc., pp. 8026–8037 (cit. on pp. 199, 241).

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019b). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. cite arxiv:1912.01703Comment: 12 pages, 3 figures, NeurIPS 2019. URL: http://arxiv.org/abs/1912.01703 (cit. on p. 216).

Pearson, John E. (1993). "Complex Patterns in a Simple System". In: *Science* 261.5118, pp. 189–192. eprint: https://www.science.org/doi/pdf/10.1126/science.261.5118.189. URL: https://www.science.org/doi/abs/10.1126/science.261.5118.189 (cit. on pp. 115, 260).

Perez, Ethan, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville (2018). "FiLM: Visual Reasoning with a General Conditioning Layer". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger. AAAI Press, pp. 3942–3951. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16528 (cit. on pp. 116, 253).

Pfaff, Tobias, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia (2021). "Learning Mesh-Based Simulation with Graph Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=roNqYL0_XP (cit. on p. 40).

Polyanin, Andrei D. (July 2019). "Functional separable solutions of nonlinear convection–diffusion equations with variable coefficients". In: *Communications in Nonlinear Science and Numerical Simulation* 73, pp. 379–390 (cit. on p. 57).

Polyanin, Andrei D. (2020). "Functional Separation of Variables in Nonlinear PDEs: General Approach, New Solutions of Diffusion-Type Equations". In: *Mathematics* 8.1, p. 90 (cit. on p. 57).

Polyanin, Andrei D. and Alexei I. Zhurov (Feb. 2020). "Separation of variables in PDEs using nonlinear transformations: Applications to reaction–diffusion type equations". In: *Applied Mathematics Letters* 100, p. 106055 (cit. on p. 57).

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. eprint: https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.690381003. URL: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003 (cit. on p. 39).

Radford, Alec, Luke Metz, and Soumith Chintala (2016). "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1511.06434 (cit. on p. 200).

Raghu, Aniruddh, Maithra Raghu, Samy Bengio, and Oriol Vinyals (2020). "Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rkgMkCEtPB (cit. on pp. 118, 120, 251).

Raissi, M., P. Perdikaris, and G. 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. URL: https://www.sciencedirect.com/science/article/pii/S0021999118307125 (cit. on p. 111).

Raissi, Maziar (2018). "Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations". In: *Journal of Machine Learning Research* 19.25, pp. 1–24 (cit. on pp. 38, 58, 192).

Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019a). "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 (cit. on p. 37).

Raissi, Maziar, Paris Perdikaris, and George Em Karniadakis (2019b). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 473, pp. 686–707 (cit. on pp. 73, 78, 146).

Raissi, Maziar, Alireza Yazdani, and George Em Karniadakis (2020). "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations". In: *Science* 367.6481, pp. 1026–1030 (cit. on pp. 38, 56).

Ramachandran, Prajit, Barret Zoph, and Quoc V. Le (2018). *Searching for Activation Functions*. URL: https://openreview.net/forum?id=SkBYYyZRZ (cit. on p. 261).

Rebuffi, Sylvestre-Alvise, Hakan Bilen, and Andrea Vedaldi (2017). "Learning multiple visual domains with residual adapters". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. (cit. on p. 121).

Rebuffi, Sylvestre-Alvise, Hakan Bilen, and Andrea Vedaldi (2018). "Efficient Parametrization of Multi-Domain Deep Neural Networks". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Com-

puter Society, pp. 8119–8127. URL: http://openaccess.thecvf.com/content%5C_cvpr%5C_2018/html/Rebuffi%5C_Efficient%5C_Parametrization%5C_of%5C_CVPR%5C_2018%5C_paper.html (cit. on p. 121).

Rehmer, Alexander and Andreas Kroll (2020). "On the vanishing and exploding gradient problem in Gated Recurrent Units". In: *IFAC-PapersOnLine* 53, pp. 1243–1248 (cit. on p. 25).

Reichstein, Markus, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and & Prabhat (2019). "Deep learning and process understanding for data-driven Earth system science". In: *Nature* 566, pp. 195–204 (cit. on pp. 73, 75, 104).

Renardy, Michael and Robert C Rogers (2006). *An introduction to partial differential equations*. Vol. 13. Springer Science & Business Media (cit. on p. 31).

Requeima, James, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E. Turner (2019). "Fast and Flexible Multi-Task Classification using Conditional Neural Adaptive Processes". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, pp. 7957–7968. URL: https://proceedings.neurips.cc/paper/2019/file/1138d90ef0a0848a542e57d1595f58ea-Paper.pdf (cit. on p. 121).

Rezende, Danilo Jimenez and Shakir Mohamed (2015). "Variational Inference with Normalizing Flows". In: ICML'15. Lille, France: JMLR.org (cit. on p. 47).

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (June 2014). "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research. Bejing, China: PMLR, pp. 1278–1286 (cit. on p. 55).

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 (cit. on pp. 11, 39).

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. Cham: Springer International Publishing, pp. 234–241 (cit. on p. 200).

Rubanova, Yulia, Ricky T. Q. Chen, and David K Duvenaud (2019). "Latent Ordinary Differential Equations for Irregularly-Sampled Time Series". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A.

Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/42a6845a557bef704ad8ac9cb4461d43-Paper.pdf (cit. on pp. 36, 42, 55, 62).

Ruder, Sebastian (2017). "An Overview of Multi-Task Learning in Deep Neural Networks". In: *CoRR* abs/1706.05098. arXiv: 1706.05098. URL: http://arxiv.org/abs/1706.05098 (cit. on p. 251).

Rumelhart, David E. and James L. McClelland (1987). "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, pp. 318–362 (cit. on p. 41).

Rusu, Andrei A., Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell (2019). "Meta-Learning with Latent Embedding Optimization". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL: https://openreview.net/forum?id=BJgklhAcK7 (cit. on p. 116).

Ryder, Tom, Andrew Golightly, A. Stephen McGough, and Dennis Prangle (July 2018). "Black-Box Variational Inference for Stochastic Differential Equations". 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. 4423–4432 (cit. on p. 55).

Saemundsson, Steindor, Alexander Terenin, Katja Hofmann, and Marc Deisenroth (2020). "Variational integrator networks for physically structured embeddings". In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 3078–3087 (cit. on pp. 39, 45).

Sagawa, Shiori, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang (2020). "Distributionally Robust Neural Networks". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=ryxGuJrFvS (cit. on p. 120).

Saha, Priyabrata, Saurabh Dash, and Saibal Mukhopadhyay (2020a). "PhICNet: Physics-Incorporated Convolutional Recurrent Neural Networks for Modeling Dynamical Systems". In: *arXiv preprint arXiv:2004.06243* (cit. on pp. 63, 195).

Saha, Priyabrata, Saurabh Dash, and Saibal Mukhopadhyay (2020b). "PhICnet: Physics-incorporated convolutional recurrent neural networks for modeling dynamical systems". In: *arXiv preprint arXiv:2004.06243* (cit. on p. 39).

San, Omer and Romit Maulik (Apr. 2018). "Machine learning closures for model order reduction of thermal fluids". In: *Applied Mathematical Modelling* 60 (cit. on p. 39).

Schüldt, Christian, Ivan Laptev, and Barbara Caputo (Aug. 2004). "Recognizing Human Actions: A Local SVM Approach". In: *Proceedings of the 17th Inter-*

national Conference on Pattern Recognition, 2004. ICPR 2004. Vol. 3, pp. 32–36 (cit. on p. 204).

Schuster, Mike and Kuldip Paliwal (Dec. 1997). "Bidirectional recurrent neural networks". In: *Signal Processing, IEEE Transactions on* 45, pp. 2673–2681 (cit. on p. 26).

Seo, Younggyo, Kimin Lee, Ignasi Clavera Gilaberte, Thanard Kurutach, Jinwoo Shin, and Pieter Abbeel (2020). "Trajectory-wise Multiple Choice Learning for Dynamics Generalization in Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 12968–12979 (cit. on p. 73).

Sepulchre, Pierre, Arnaud Caubel, Jean-Baptiste Ladant, Laurent Bopp, Olivier Boucher, Pascale Braconnot, Patrick Brockmann, Anne Cozic, Yannick Donnadieu, Victor Estella-Perez, Christian Ethé, Frédéric Fluteau, Marie-Alice Foujols, Guillaume Gastineau, Josefine Ghattas, Didier Hauglustaine, Frédéric Hourdin, Masa Kageyama, Myriam Khodri, Olivier Marti, Yann Meurdesoif, Juliette Mignot, Anta-Clarisse Sarr, Jérôme Servonnat, Didier Swingedouw, Sophie Szopa, and Delphine Tardif (Dec. 2019). "IPSL-CM5A2. An Earth System Model designed for multi-millennial climate simulations". English. In: *Geoscientific Model Development Discussions*, pp. 1–57. (Visited on 02/05/2020) (cit. on p. 136).

Shaier, Sagi, Maziar Raissi, and Padmanabhan Seshaiyer (2021). "Data-driven approaches for predicting spread of infectious diseases through DINNs: Disease Informed Neural Networks". In: (cit. on p. 104).

Shi, Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun WOO (2015). "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf (cit. on pp. 27, 53, 65, 68).

Simon, Noah, Jerome Friedman, Trevor Hastie, and Robert Tibshirani (Apr. 2013). "A Sparse-Group Lasso". In: *Journal of Computational and Graphical Statistics* 22.2, pp. 231–245 (cit. on p. 128).

Simonyan, Karen and Andrew Zisserman (2015). "Very deep convolutional networks for large-scale image recognition". In: *International Conference on Learning Representations* (cit. on p. 200).

Sirignano, Justin and Konstantinos Spiliopoulos (2018). "DGM: A deep learning algorithm for solving partial differential equations". In: *Journal of Computa-*

tional Physics 375 (Dms 1550918), pp. 1339–1364 (cit. on pp. 37, 58, 73, 82, 104, 192).

Srivastava, Nitish, Elman Mansimov, and Ruslan Salakhudinov (July 2015). "Unsupervised Learning of Video Representations using LSTMs". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, pp. 843–852 (cit. on pp. 64, 197).

Steenkiste, Sjoerd van, Michael Chang, Klaus Greff, and Jürgen Schmidhuber (2018). "Relational Neural Expectation Maximization: Unsupervised Discovery of Objects and their Interactions". In: *International Conference on Learning Representations* (cit. on p. 55).

Stokes, G. G. (Jan. 1851). "On the Effect of the Internal Friction of Fluids on the Motion of Pendulums". In: *Transactions of the Cambridge Philosophical Society* 9, p. 8 (cit. on pp. 115, 260).

Tait, Daniel J and Theodoros Damoulas (2020). "Variational Autoencoding of PDE Inverse Problems". In: *arXiv preprint arXiv:2006.15641* (cit. on pp. 39, 45).

Takeishi, Naoya, Yoshinobu Kawahara, and Takehisa Yairi (2017). "Learning Koopman Invariant Subspaces for Dynamic Mode Decomposition". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/3a835d3215755c435ef4fe9965a3f2a0-Paper.pdf (cit. on p. 42).

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 (cit. on p. 39).

Thrun, Sebastian and Lorien Y. Pratt (1998). "Learning to Learn: Introduction and Overview". In: *Learning to Learn*. Ed. by Sebastian Thrun and Lorien Y. Pratt. Springer, pp. 3–17. URL: https://doi.org/10.1007/978-1-4615-5529-2_1 (cit. on p. 104).

Tibshirani, Robert (1996). "Regression Shrinkage and Selection Via the Lasso". en. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1, pp. 267–288 (cit. on p. 126).

Tompson, Jonathan, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin (Aug. 2017a). "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. URL: http://proceedings.mlr.press/v70/tompson17a.html (cit. on p. 40).

Tompson, Jonathan, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin (Aug. 2017b). "Accelerating Eulerian Fluid Simulation With Convolutional Net-

works". 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. International Convention Centre, Sydney, Australia: PMLR, pp. 3424–3433 (cit. on p. 56).

Toth, Peter, Danilo J. Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins (2020). "Hamiltonian Generative Networks". In: *International Conference on Learning Representations* (cit. on pp. 55, 82, 217).

Tulyakov, Sergey, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz (June 2018). "MoCoGAN: Decomposing Motion and Content for Video Generation". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1526–1535 (cit. on p. 55).

Um, Kiwon, Robert Brand, Fei Yun, Philipp Holl, and Nils Thuerey (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 (cit. on p. 90).

Ummenhofer, Benjamin, Lukas Prantl, Nils Thuerey, and Vladlen Koltun (2020). "Lagrangian Fluid Simulation with Continuous Convolutions". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 73).

Unterthiner, Thomas, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly (2018). "Towards accurate generative models of video: A new metric & challenges". In: *arXiv preprint arXiv:1812.01717* (cit. on p. 204).

Urain, J., M. Ginesi, D. Tateo, and J. Peters (2020). "ImitationFlow: Learning Deep Stable Stochastic Dynamic Systems by Normalizing Flows". In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5231–5237 (cit. on p. 49).

Van Den Berg, Rianne, Leonard Hasenclever, Jakub M. Tomczak, and Max Welling (2018). "Sylvester normalizing flows for variational inference". English. In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Ed. by Amir Globerson, Amir Globerson, and Ricardo Silva. 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018. 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018 ; Conference date: 06-08-2018 Through 10-08-2018. Association For Uncertainty in Artificial Intelligence (AUAI), pp. 393–402 (cit. on p. 47).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (cit. on p. 27).

Villar, Soledad, David W Hogg, Kate Storey-Fisher, Weichi Yao, and Ben Blum-Smith (2021). "Scalars are universal: Equivariant machine learning, structured like classical physics". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan. URL: https://openreview.net/forum?id=ba27-RzNaIv (cit. on p. 146).

Villegas, Ruben, Arkanath Pathak, Harini Kannan, Dumitru Erhan, Quoc V. Le, and Honglak Lee (2019). "High Fidelity Video Prediction with Large Stochastic Recurrent Neural Networks". In: *Advances in Neural Information Processing Systems 32*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Curran Associates, Inc., pp. 81–91 (cit. on p. 203).

Villegas, Ruben, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee (2017a). "Decomposing motion and content for natural video sequence prediction". In: *International Conference on Learning Representations* (cit. on p. 55).

Villegas, Ruben, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee (Aug. 2017b). "Learning to Generate Long-term Future via Hierarchical Prediction". 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. International Convention Centre, Sydney, Australia: PMLR, pp. 3560–3569 (cit. on p. 55).

Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol (2008). "Extracting and Composing Robust Features with Denoising Autoencoders". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. event-place: Helsinki, Finland. New York, NY, USA: ACM, pp. 1096–1103 (cit. on p. 128).

Vogels, Thijs, Sai Praneeth Karimireddy, and Martin Jaggi (2019). "PowerSGD: Practical Low-Rank Gradient Compression for Distributed Optimization". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/d9fbed9da256e344c1fa46bb46c34c5f-Paper.pdf (cit. on p. 112).

Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba (2016). "Generating Videos with Scene Dynamics". In: *Advances in Neural Information Processing Systems 29*. Ed. by Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett. Curran Associates, Inc., pp. 613–621 (cit. on p. 55).

Wan, Zhong, Pantelis Vlachas, Petros Koumoutsakos, and Themistoklis Sapsis (Mar. 2018). "Data-assisted reduced-order modeling of extreme events in complex dynamical systems". In: *PLOS ONE* 13 (cit. on p. 27).

Wandel, Nils, Michael Weinmann, and Reinhard Klein (2021a). "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 (cit. on p. 40).

Wandel, Nils, Michael Weinmann, and Reinhard Klein (2021b). "Learning Incompressible Fluid Dynamics from Scratch - Towards Fast, Differentiable Fluid Models that Generalize". In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. URL: https://openreview.net/forum?id=KUDUoRsEphu (cit. on p. 104).

Wang, Haoxiang, Han Zhao, and Bo Li (2021). "Bridging Multi-Task Learning and Meta-Learning: Towards Efficient Training and Effective Adaptation". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 10991–11002. URL: http://proceedings.mlr.press/v139/wang21ad.html (cit. on p. 121).

Wang, Yi-Jen and Chin-Teng Lin (1998). "Runge-Kutta neural network for identification of dynamical systems in high accuracy". In: *IEEE Transactions on Neural Networks* 9.2, pp. 294–307 (cit. on p. 35).

Wang, Jindong, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin (2021). "Generalizing to Unseen Domains: A Survey on Domain Generalization". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*. Ed. by Zhi-Hua Zhou. ijcai.org, pp. 4627–4635. URL: https://doi.org/10.24963/ijcai.2021/628 (cit. on p. 104).

Wang, Qi, Feng Li, Yi Tang, and Yan Xu (2019). "Integrating model-driven and data-driven methods for power system frequency stability assessment and control". In: *IEEE Transactions on Power Systems* 34.6, pp. 4557–4568 (cit. on pp. 220, 221).

Wang, Rui, Robin Walters, and Rose Yu (2021). "Meta-Learning Dynamics Forecasting Using Task Inference". In: *CoRR* abs/2102.10271. arXiv: 2102.10271. URL: https://arxiv.org/abs/2102.10271 (cit. on pp. 105, 121).

Wang, Sifan, Hanwen Wang, and Paris Perdikaris (2021a). "On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks". In: *Computer Methods in Applied Mechanics and Engineering* 384, p. 113938. URL: https://www.sciencedirect.com/science/article/pii/S0045782521002759 (cit. on p. 38).

Wang, Sifan, Xinling Yu, and Paris Perdikaris (2021b). "When and why PINNs fail to train: A neural tangent kernel perspective". In: *Journal of Computational*

Physics, p. 110768. URL: https://www.sciencedirect.com/science/article/pii/S002199912100663X (cit. on pp. 38, 146).

Wang, Yunbo, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S Yu (July 2018). "PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 5123–5132. URL: https://proceedings.mlr.press/v80/wang18b.html (cit. on pp. 28, 68, 82, 198).

Wang, Yunbo, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei (2019a). "Eidetic 3D LSTM: A Model for Video Prediction and Beyond". In: *International Conference on Learning Representations* (cit. on p. 68).

Wang, Yunbo, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu (2017). "PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs". In: *Advances in Neural Information Processing Systems 30*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Curran Associates, Inc., pp. 879–888 (cit. on p. 68).

Wang, Yunbo, Jianjing Zhang, Hongyu Zhu, Mingsheng Long, Jianmin Wang, and Philip S. Yu (2019b). "Memory in Memory: A Predictive Neural Network for Learning Higher-Order Non-Stationarity From Spatiotemporal Dynamics". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9146–9154 (cit. on pp. 28, 63, 65, 68, 82, 84, 199).

Weissenborn, Dirk, Oscar Täckström, and Jakob Uszkoreit (2020). "Scaling Autoregressive Video Models". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rJgsskrFwH (cit. on pp. 27, 203).

Williams, Ronald J. (1992). "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning". In: *Machine Learning*, pp. 229–256 (cit. on p. 127).

Williams, Ronald J. and Jing Peng (Dec. 1990). "An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories". In: *Neural Computation* 2.4, pp. 490–501. eprint: https://direct.mit.edu/neco/article-pdf/2/4/490/812062/neco.1990.2.4.490.pdf. URL: https://doi.org/10.1162/neco.1990.2.4.490 (cit. on p. 23).

Wu, Yan, Mihaela Rosca, and Timothy Lillicrap (May 2019). "Deep Compressed Sensing". en. In: *International Conference on Machine Learning*, pp. 6850–6860 (cit. on p. 129).

Xie, Saining, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He (2017). "Aggregated Residual Transformations for Deep Neural Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Hon-*

olulu, HI, USA, July 21-26, 2017. IEEE Computer Society, pp. 5987–5995. URL: https://doi.org/10.1109/CVPR.2017.634 (cit. on p. 36).

Xing, Eric P., Michael I. Jordan, and Richard M. Karp (2001). "Feature Selection for High-Dimensional Genomic Microarray Data". In: *In Proceedings of the Eighteenth International Conference on Machine Learning*. Morgan Kaufmann, pp. 601–608 (cit. on p. 128).

Yang, Sheng, Rui Zhang, Feiping Nie, and Xuelong Li (May 2019). "Unsupervised Feature Selection Based on Reconstruction Error Minimization". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. ISSN: 1520-6149, pp. 2107–2111 (cit. on pp. 126, 128).

Yin, Yuan, Ibrahim Ayed, Emmanuel de Bézenac, Nicolas Baskiotis, and Patrick Gallinari (2021a). "LEADS: Learning Dynamical Systems that Generalize Across Environments". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan. URL: https://openreview.net/forum?id=HD6CxZtbmIx (cit. on pp. 105, 116, 121, 251, 261).

Yin, Yuan, Vincent Le Guen, Jérémie Donà, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari (2021b). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *The Ninth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=kmG8vRXTFv (cit. on pp. 13, 71, 94, 95, 104, 111, 242, 245).

Yingzhen, Li and Stephan Mandt (July 2018). "Disentangled Sequential Autoencoder". 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. 5670–5679 (cit. on p. 55).

Yıldız, Cagatay, Markus Heinonen, and Harri Lahdesmaki (2019). "ODE^{2}VAE: Deep generative second order ODEs with Bayesian neural networks". In: *Advances in Neural Information Processing Systems 32*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Curran Associates, Inc., pp. 13412–13421 (cit. on pp. 54, 55).

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 (cit. on p. 39).

Yu, Lantao, Weinan Zhang, Jun Wang, and Yong Yu (2017). "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. Ed. by Satinder P. Singh and Shaul Markovitch. AAAI Press, pp. 2852–2858. URL: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14344 (cit. on p. 46).

Yu, Lei and Huan Liu (Jan. 2003). "Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution". In: vol. 2, pp. 856–863 (cit. on p. 128).

Yuan, Ming and Yi Lin (2006). "Model selection and estimation in regression with grouped variables". In: *Journal of the Royal Statistical Society, Series B* 68, pp. 49–67 (cit. on pp. 128, 131).

Zaeemzadeh, Alireza, Nazanin Rahnavard, and Mubarak Shah (2020). "Norm-Preservation: Why Residual Networks Can Become Extremely Deep?" In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* (cit. on p. 32).

Zhang, Junbo, Yu Zheng, and Dekang Qi (2017). "Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI'17. San Francisco, California, USA: AAAI Press, pp. 1655–1661 (cit. on pp. 67, 198).

Zhou, Jie, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun (2020). "Graph neural networks: A review of methods and applications". In: *AI Open* 1, pp. 57–81. URL: https://www.sciencedirect.com/science/article/pii/S2666651021000012 (cit. on p. 40).

Zhou, Yang, Rong Jin, and Steven Chu-Hong Hoi (Mar. 2010). "Exclusive Lasso for Multi-task Feature Selection". en. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 988–995 (cit. on p. 128).

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros (Oct. 2017). "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 46).

Zhu, Pengfei, Wangmeng Zuo, Lei Zhang, Qinghua Hu, and Simon C. K. Shiu (Feb. 2015). "Unsupervised feature selection by regularized self-representation". en. In: *Pattern Recognition* 48.2, pp. 438–446 (cit. on p. 128).

Zhuang, Juntang, Nicha Dvornek, Xiaoxiao Li, Sekhar Tatikonda, Xenophon Papademetris, and James Duncan (13–18 Jul 2020). "Adaptive Checkpoint Adjoint Method for Gradient Estimation in Neural ODE". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 11639–11649. URL: https://proceedings.mlr.press/v119/zhuang20a.html (cit. on p. 35).

Zintgraf, Luisa, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson (Sept. 2019). "Fast Context Adaptation via Meta-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. 7693–7702. URL: http://proceedings.mlr.press/v97/zintgraf19a.html (cit. on pp. 112, 116, 121, 251).

Part VIII

APPENDICES

APPENDIX A

NEURAL NETWORKS: A SHORT INTRODUCTION

We here provide a synthetic overview of Neural Network (NN) models for supervised learning. Supervised learning refers to learning settings in which we are interested in predicting a variable y, given covariables x in particular with a particular clas of parametric models: NN.

For simplicity, we consider Multi-Layer Perceptron (MLP), that consists in alternating the application of matrix multiplication and a fixed non-linearity called *activation* function. Consider a $(p+1)$-layer MLP, it is defined by its weights $\theta = (W^1, ..., W^p)$, where $\forall i,\ W_i$ is a weight matrix, and by its non-linear activation function denoted σ. A NN defines a computational graph defined by the following equations:

$$\begin{cases} A^0 = x & \text{(A.1a)} \\ A^i = \sigma(W^i.A^{i-1}),\ \forall i \in \{1, ..., p-1\} & \text{(A.1b)} \\ O = \psi(W^p A^{p-1}) & \text{(A.1c)} \end{cases}$$

eq. (A.1a) is the input layer and x is simply our data. eq. (A.1b) defines the hidden layer computations, A^i are called the activations. Finally, eq. (A.1c) defines the output layer and ψ is a predefined function (sigmoid, softmax, tanh, etc.).

With these notations, the neurons' outputs simply are gated activations of the matrix multiplication, i.e the output of the neuron j at depth i is $A^i_j = \sigma(A^{i-1^T}.W^i_j)$, where W^i_j is the j^{th} line of W^i.

A noticeable property of such a MLP is the *universal approximation theorem* that states that with a single hidden layer neural network (with enough neurons, i.e. with the dimension of W^1 large enough) one can approximate uniformly any continuous function (over a compact set), see (Cybenko 1989).

In general, NN are optimized by minimizing the conditional negative log-likelihood (or equivalently maximizing the conditional log-likelood of the model): The likelihood function writes as:

$$L(y; x; \theta) = p_{nn}(y|x) \tag{A.2}$$

In this setting a training set, denoted D_T is made of n-pairs, so that $D_T = (y_i; x_i)$, we can rewrite it as:

$$L(y_1; ...., y_n; x_1, ..., x_n; \theta) = p_{nn}(y_1, ..., y_n | x_1, ..., x_n) = \Pi_{i=1}^{n} p_{nn}(y_i | x_i). \tag{A.3}$$

Taking the log out of the last term, we obtain:

$$\log(L)(y_1; ...., y_n; x_1, ..., x_n; \theta) = \sum_{i=1}^{n} \log p_{nn}(y_i | x_i) \approx E_{x,y \sim p_{data}} \log p_{nn}(y_i | x_i) \tag{A.4}$$

Let $f(x; \theta)$ a NN obeying the computation of eq. (A.1), and using a simple Gaussian prior for p_{nn} i.e, so that $p_{nn}(y|x) \sim \mathcal{N}(y; f(x, \theta); \sigma^2)$, eq. (A.4) writes as:

$$\log(L)(y_1; ...., y_n; x_1, ..., x_n; \theta) = -\lambda \times \left\| y - f(x; \theta) \right\|^2 + constant \tag{A.5}$$

where λ is a proportionality constant independant of the NN ψ (and so is the constant).

Therefore maximizing the loglikelihood $\log L$, assuming a Gaussian setting amounts to minimize the $\| \ . \ \|$ between the model prediction $f(x, \theta)$ and the observed output y. Given the differentiability of f with respect to the parameters θ, we can compute the gradients of $-\log L$, and optimize the θ in the steepest direction. This training procedure is called Gradient Descent (GD). For compuational memory and robustness reasons, practical optimization evaluate the gradients $\frac{\partial \log L}{\partial \theta}$ on small subset of the training set D_T defining an optimization procedure called Stochastic Gradient Descent (SGD).

APPENDIX B

APPENDICES TO SPATIOTEMPORAL DISENTANGLEMENT

B.1 Proofs

B.1.1 Resolution of the Heat Equation

In this section, we succinctly detail a proof for the existence and uniqueness for the solution to the two-dimensional heat equation. It shows that product-separable solutions allow to build the entire solution space for this problem, highlighting our interest in the research of separable solutions.

Existence through separation of variables. Consider the heat equation problem:

$$\frac{\partial u}{\partial t} = c^2 \frac{\partial^2 u}{\partial x^2}, \qquad u(0,t) = u(L,t) = 0, \qquad u(x,0) = f(x). \tag{B.1}$$

Assuming product separability of u with $u(x,t) = u_1(x)u_2(t)$ in eq. (B.1) gives:

$$c^2 \frac{u_1''(x)}{u_1(x)} = \frac{u_2'(t)}{u_2(t)}. \tag{B.2}$$

Both sides being independent of each other variables, they are equal to a constant denoted by $-\alpha$. If α is negative, solving the right side of eq. (B.2) results to non-physical solutions with exponentially increasing temperatures, and imposing border condition of eq. (B.1) makes this solution collapse to the null trivial solution. Therefore, we consider that $\alpha > 0$.

Both sides of eq. (B.2) being equal to a constant leads to a second-order ODE on u_1 and a first-order ODE on u_2, giving the solution shapes, with constants A, B and D:

$$\begin{cases} u_1(x) &= A\cos\left(\sqrt{\alpha}\,x\right) + B\sin\left(\sqrt{\alpha}\,x\right) \\ u_2(t) &= D\mathrm{e}^{-\alpha c^2 t} \end{cases}. \tag{B.3}$$

Link with initial and boundary conditions. We now link the above equation to the boundary conditions of the problem. Because our separation is multiplicative, we can omit D for non-trivial solutions and set it without loss of generality to 1, as it only scales the values of A and B.

Boundary condition $u(0,t) = u(L,t) = 0$, along with the fact that for all $t > 0$, $u_2(t) \neq 0$, give:

$$A = 0, \qquad B\mathrm{e}^{-\alpha c^2 t}\sin\left(\sqrt{\alpha}L\right) = 0, \tag{B.4}$$

which means that, for a non-trivial solution (i.e., $B \neq 0$), we have for some $n \in \mathbb{N}$: $\sqrt{\alpha} = n\pi/L$. We can finally express our product-separable solution to the heat equation without initial conditions as:

$$u(x,t) = B\sin\left(\frac{n\pi}{L}x\right)\exp\left(-\left(\frac{cn\pi}{L}\right)^2 t\right). \tag{B.5}$$

Considering the superposition principle, because the initial problem is homogeneous, all linear combinations of eq. (B.5) are solutions of the heat equation without initial conditions. Therefore, any following function is a solution of the heat equation without initial conditions.

$$u(x,t) = \sum_{n=0}^{+\infty} B_n \sin\left(\frac{n\pi}{L}x\right)\exp\left(-\left(\frac{cn\pi}{L}\right)^2 t\right). \tag{B.6}$$

Finally, considering the initial condition $u(x,0) = f(x)$, a Fourier decomposition of f allows to choose appropriate values for all coefficients B_n, showing that, for any initial condition f, there exists a solution to eq. (B.1) of the form of eq. (B.6).

Uniqueness. We present here elements of proof for establishing the uniqueness of the solutions of eq. (B.1) that belong to $\mathcal{C}^2\left([0,1]\times\mathbb{R}_+\right)$. Detailed and rigorous proofs are given by Le Dret and Lucquin (2016).

The key element consists in establishing the so-called Maximum Principle which states that, considering a sufficiently smooth solution, the minimum value of the solution is reached on the boundary of the space and time domains.

For null border condition as in our case, this means that the norm of the solution u is given by the norm of the initial condition f. Finally, let us consider two smooth solutions U_1 and U_2 of eq. (B.1). Then, their difference $v = U_1 - U_2$ follows the heat equation with null border and initial conditions (i.e, $v(x,0) = 0$). Because v is as regular as U_1 and U_2, it satisfies the previous fact about the norm of the solutions,

i.e, the norm of v equals the norm of its initial condition: $\|v\| = 0$. Therefore, v is null and so is $U_1 - U_2 = 0$, showing the uniqueness of the solutions.

Therefore, this shows that solutions of the form of eq. (B.6) shape the whole set of smooth solutions of eq. (B.1).

B.1.2 Heat Equation with Advection Term

Consider the heat equation with a complementary advection term, for $x \in (-1, 1)$, $t \in (0, T)$ and a constant $c \in \mathbb{R}_+$.

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = \chi\frac{\partial^2 u}{\partial x^2}, \quad . \tag{B.7}$$

We give here details for the existence of product-separable solutions of eq. (B.7). To this end, let us choose real constants α and β, and consider the following change of variables for u:

$$u(x,t) = v(x,t)\mathrm{e}^{\alpha x+\beta t}. \tag{B.8}$$

The partial derivatives from eq. (B.7) can be rewritten as functions of the new variable v:

$$\frac{\partial u}{\partial t} = \frac{\partial v}{\partial t}\,\mathrm{e}^{\alpha x+\beta t} + \beta v\mathrm{e}^{\alpha x+\beta t} \tag{B.9}$$

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial x}\,\mathrm{e}^{\alpha x+\beta t} + \alpha v\mathrm{e}^{\alpha x+\beta t} \tag{B.10}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{\partial^2 v}{\partial x^2}\,\mathrm{e}^{\alpha x+\beta t} + 2\alpha\,\frac{\partial v}{\partial x}\,\mathrm{e}^{\alpha x+\beta t} + \alpha^2 v\mathrm{e}^{\alpha x+\beta t} \tag{B.11}$$

Using these expressions in eq. (B.7) and dividing it by $\mathrm{e}^{\alpha x+\beta t}$ lead to:

$$\frac{\partial v}{\partial t} + \big(\beta + c\alpha - \alpha^2\chi\big)v + (c - 2\alpha\chi)\frac{\partial v}{\partial x} = \nu\frac{\partial^2 v}{\partial x^2}. \tag{B.12}$$

α and β can then be set such that:

$$\beta + c\alpha - \alpha^2\chi = 0 \qquad\qquad c - 2\alpha\chi = 0, \tag{B.13}$$

to retrieve the standard two-dimensional heat equation of eq. (B.1) given by:

$$\frac{\partial v}{\partial t} = \chi\frac{\partial^2 v}{\partial x^2}, \tag{B.14}$$

which is known to have product-separable solutions as explained in the previous section. This more generally shows that all solutions of eq. (B.7) can be retrieved from solutions to eq. (B.1).

B.2 Accessing Time Derivatives of S and Deriving a Feasible Weaker Constraint

Explicitly constraining the time derivative of $E_S\big(V_\tau(t)\big)$ as explained in section 6.4.4 is a difficult matter in practice. Indeed, E_S does not take as input neither the time coordinate t nor spatial coordinates x and y as done by Maziar Raissi 2018 and Sirignano and Spiliopoulos 2018, which allows them to directly estimate the networks derivative thanks to automatic differentiation. In our case, E_S rather takes as input a finite number of observations, making this derivative impractical to compute.

To discretize eq. (6.10) and find a weaker constraint, we chose to leverage the Cauchy-Schwarz inequality. We presented and used a version where we applied this inequality on the whole integration domain, i.e., from t_0 to $t_1 - \tau\Delta t$. We highlight that this inequality can also be applied on subintervals of the integration domain, generalizing our proposition. Indeed, let $p \in \mathbb{N}^*$ and consider a sequence of $t^{(k)}$ for $k \in [\![0, p]\!]$ such that $t_0 = t^{(0)} \leq t^{(1)} \leq \ldots \leq t^{(p)} = t_1 - \tau\Delta t$. Then, using the Cauchy-Schwarz inequality, we obtain:

$$\begin{aligned}
\int_{t_0}^{t_1-\tau\Delta t} \left\| \frac{\partial E_S\big(V_\tau(t)\big)}{\partial t} \right\|_2^2 \mathrm{d}t &= \sum_{k=0}^{k=p} \int_{t^{(k-1)}}^{t^{(k)}} \left\| \frac{\partial E_S\big(V_\tau(t)\big)}{\partial t} \right\|_2^2 \mathrm{d}t \\
&\geq \sum_{k=0}^{k=p} \frac{1}{t^{(k)} - t^{(k-1)}} \left\| \int_{t^{(k-1)}}^{t^{(k)}} \frac{\partial E_S\big(V_\tau(t)\big)}{\partial t} \,\mathrm{d}t \right\|_2^2 \\
&\geq \sum_{k=0}^{k=p} \frac{1}{t^{(k)} - t^{(k-1)}} \left\| E_S\Big(V_\tau\big(t^{(k)}\big)\Big) - E_S\Big(V_\tau\big(t^{(k-1)}\big)\Big) \right\|_2^2 .
\end{aligned} \tag{B.15}$$

Our constraint is a special case of this development, with $p = 1$. Nevertheless, we experimentally found that our simple penalty is sufficient to achieve state-of-the-art performance at a substantially reduced computational cost. We notice that other invariance constraints such as the one of Denton and Birodkar 2017 can also be derived thanks to framework, showing the generality of our approach.

B.3 Of Spatiotemporal Disentanglement

B.3.1 Modeling Spatiotemporal Phenomena with Differential Equations

Besides their increasing popularity to model spatiotemporal phenomena (see section 6.2), the ability of residual networks to facilitate learning (Haber and Ruthotto 2017) as well as the success of their continuous counterpart (R. T. Q. Chen et al. 2018) motivate our choice. Indeed, learning ODEs or discrete approximations as residual networks has become standard for a variety of tasks such as classification, inpainting, and generative models. Consequently, their application to forecasting physical processes and videos is only a natural extension of its already broad applicability discussed in section 6.2. Furthermore, they present interesting properties, as detailed below.

B.3.2 Separation of Variables Preserves the Mutual Information of S and T through Time

Invertible Flow of an ODE

We first highlight that the general ODE eq. (6.7) admits, according to the Cauchy–Lipschitz theorem, exactly one solution for a given initial condition, since f is implemented with a standard neural network (see appendix B.5), making it Lipschitz-continuous. Consequently, the flow of this ODE, denoted by Φ_t and defined as:

$$\begin{aligned}\Phi\colon \mathbb{R}\times\mathbb{R}^p &\to \mathbb{R}^p\\ (t_0, T_{t_0}) &\mapsto \Phi_t(T_{t_0}) = T_{t_0+t}\end{aligned}$$

is a bijection for all t. Indeed, let T_{t_0} be fixed and t_0, t_1 be two timesteps; thanks to the existence and unicity of the solution to the ODE with this initial condition: $\Phi_{t_0+t_1} = \Phi_{t_0}\circ\Phi_{t_1} = \Phi_{t_1}\circ\Phi_{t_0}$. Therefore, Φ_t is a bijection and $\Phi_t^{-1} = \Phi_{-t}$. Moreover, the flow is differentiable if f is continuously differentiable as well, which is not a restrictive assumption if it is implemented by a neural network with differentiable activation functions.

Preservation of Mutual Information by Invertible Mappings

A proof of the following result is given by Kraskov et al. 2004. We indicate below the major steps of the proof. Let X and Y be two random variables with marginal densities μ_X, μ_Y. Let F be a diffeomorphism acting on Y, $Y' = F(Y)$. If J_F is the determinant of the Jacobian of F, we have:

$$\mu'(x, y') = \mu(x, y) J_F(y').$$

Then, expressing the mutual information I in integral form, with the change of variables $y' = F(y)$ (F being a diffeomorphism), results in:

$$\begin{aligned} I(X, Y') &= \iint_{x,y'} \mu'(x, y') \log \frac{\mu'(x, y')}{\mu_X(x) \times \mu_{Y'}(y')} \, \mathrm{d}x \, \mathrm{d}y' \\ &= \iint_{x,y} \mu(x, y) \log \frac{\mu(x, y)}{\mu_X(x) \times \mu_Y(y)} \, \mathrm{d}x \, \mathrm{d}y \\ I(X, Y') &= I(X, Y). \end{aligned}$$

B.3.3 Ensuring Disentanglement at any Time

As noted by X. Chen et al. 2016a and Achille and Soatto 2018, mutual information I is a key metric to evaluate disentanglement. We show that our model logically preserves the mutual information between S and T through time thanks to the flow of the learned ODE on T. Indeed, with the result of mutual information preservation by diffeomorphisms, and Φ_t being a diffeomorphism as demonstrated above, we have, for all t and t':

$$I(S, T_t) = I\big(X, \Phi_{t'-t}(T_t)\big) = I(S, T_{t'}). \tag{B.16}$$

Hence, if S and T_t are disentangled, then so are S and $T_{t'}$.

The flow Φ_t being dicretized in practice, its invertibility can no longer be guaranteed in general. Some numerical schemes (Z. Chen et al. 2020a) or residual networks with Lipschitz-constrained residual blocks (Behrmann et al. 2019b) provide sufficient conditions to concretely reach this invertibility. In our case, we did not observe the need to enforce invertibility. We can also leverage the data processing inequality to show that, for any $t \geq t_0$:

$$I(S, T_{t_0}) \geq I(S, T_t), \tag{B.17}$$

since T_t is a deterministic function of T_{t_0}. Since we constrain the very first T value T_{t_0} (i.e., we do not need to go back in time), there is no imperative need

to enforce the invertibility of Φ_t in practice: the inequality also implies that, if S and T_{t_0} are disentangled, then so are S and T_t for $t \geq t_0$. Nevertheless, should the need to disentangle for $t < t_0$ appear, the aforementioned mutual information conservation properties could allow, with further practical work to ensure the effective invertibility of Φ_t, to still regularize T_{t_0} only. This is, however, out of the scope of this paper.

B.4 Datasets

B.4.1 WaveEq and WaveEq-100

These datasets are based on the two-dimensional wave equation on a functional $w(x, y, t)$:

$$\frac{\partial^2 w}{\partial t^2} = c^2 \nabla^2 w + f(x, y, t), \tag{B.18}$$

where ∇^2 is the Laplacian operator, c denotes the wave celerity, and f is an arbitrary time-dependent source term. It has several application in physics, modeling a wide range of phenomena ranging from mechanical oscillations to electromagnetism. Note that the homogeneous equation, where $f = 0$, admits product-separable solutions.

We build the WaveEq dataset by solving eq. (B.18) for $t \in [0, 0.298]$ and $x, y \in [0, 63]$. Sequences are generated using c drawn uniformly at random in $[300, 400]$ for each sequence to imitate the propagation of acoustic waves, with initial and Neumann boundary conditions:

$$w(x, y, 0) = w(0, 0, t) = w(32, 32, t) = 0, \tag{B.19}$$

and, following Saha et al. 2020a, we make use of the following source term:

$$f(x, y, t) = \begin{cases} f_0 \mathrm{e}^{-\frac{t}{T_0}} & \text{if } (x, y) \in \mathcal{B}\big((32, 32), 5\big) \\ 0 & \text{otherwise} \end{cases}, \tag{B.20}$$

with $T_0 = 0.05$ and $f_0 \sim \mathcal{U}\big([1, 30]\big)$. The source term is taken non-null in a circular central zone only in order to avoid numerical differentiation problems in the case of a punctual source.

We generate 300 sequences of 64×64 frames of length 150 from this setting by assimilating pixel $(i, j) \in [\![0, 63]\!] \times [\![0, 63]\!]$ to a point $(x, y) \in [0, 63] \times [0, 63]$ and selecting a frame per time interval of size 0.002. This discretization is used to solve eq. (B.18) as its spatial derivatives are estimated thanks to finite differences;

once computed, they are used in an ODE numerical solver to solve eq. (B.18) on t. Spatial derivatives are estimated with finite differences of order 5, and the ODE solver is the fourth-order Runge-Kutta method with the $3/8$ rule (Kutta 1901; Ernst Hairer et al. 1993) and step size 0.001. The data are finally normalized following a min-max $[0, 1]$ scaling per sequence.

The dataset is then split into training (240 sequences) and testing (60 sequences) sets. Sequences sampled during training are random chuncks of length $\nu+1 = 25$, including $\tau+1 = 5$ conditioning frames, of full-size training sequences. Sequences used during testing are all possible chunks of length $\tau+1+40 = 45$ from full-size testing sequences.

Finally, WaveEq-100 is created from WaveEq by selecting 100 pixels uniformly at random. The extracted pixels are selected before training and are fixed for both training and testing. Therefore, train and test sequences for WaveEq-100 consist of vector of size 100 extracted from WaveEq frames. Training and testing sequences are chosen to be the same as those of WaveEq.

B.4.2 Sea Surface Temperature

SST is composed of sea surface temperatures of the Atlantic ocean generated using E.U. Copernicus Marine Service Information thanks to the state-of-the-art simulation engine NEMO. The use of a so-called reanalysis procedure implies that these data accurately represent the actual temperature measures. For more information, we refer to the complete description of the data by Bézenac et al. 2018a. The data history of this engine is available online.[1] Unfortunately, due to recent maintenance, data history is limited to the last three years; prior histories should be manually requested.

The dataset uses daily temperature acquisitions from Thursday 28$^{\text{th}}$ December, 2006 to Wednesday 5$^{\text{th}}$ April, 2017 of a 481×781 zone, from which 29 zones of size 64×64 zones are extracted. We follow the same setting as Bézenac et al. 2018a by training all models with $\tau + 1 = 4$ conditioning steps and $\nu - \tau = 6$ steps to predict, and evaluating them only on zones 17 to 20. These zones are particularly interesting since they are the places where cold waters meet warm waters, inducing more pronounced motion.

We normalize the data in the same manner as Bézenac et al. 2018a. Each daily acquisition of a zone is first normalized using the mean and standard deviation of measured temperatures in this zone computed for all days with the same date of the year from the available data (daily history climatological normalization).

1. https://resources.marine.copernicus.eu/?option=com_csw&view=details&product_id=GLOBAL_ANALYSIS_FORECAST_PHY_001_024.

Each zone is then normalized so that the mean and variance over all acquisitions correspond to those of a standard Gaussian distribution. These normalized data are finally fed to the model; MSE scores reported in table 6.1 are computed once the performed normalization of the data and model prediction is reverted to the original temperature measurement space, in order to compute physically meaningful scores.

Training sequences correspond to randomly selected chunks of length $\nu = 10$ in the first 2987 acquisitions (corresponding to 80% of total acquisitions), and testing sequences to all possible chunks of length $\nu = 10$ in the remaining 747 acquisitions.

B.4.3 Moving MNIST

This dataset involves two MNIST digits (LeCun et al. 1998) of size 28×28 that linearly move within 64×64 frames and deterministically bounce against frame borders following reflection laws. We use the modified version of the dataset proposed by Franceschi et al. 2020 instead of the original one (Srivastava et al. 2015). We train all models in the same setting as Denton and Birodkar 2017, with $\tau + 1 = 5$ conditioning frames and $\nu - \tau = 10$ frames to predict, and test them to predict either 10 or 95 frames ahead. Training data consist in trajectories of digits from the MNIST training set, randomly generated on the fly during training. Test data are produced by computing a trajectory for each digit of the MNIST testing set, and randomly pairwise combining them, thus producing 5000 sequences.

To evaluate disentanglement with content swapping, we report PSNR and SSIM metrics between the swapped sequence produced by our model and a ground truth. However, having two digits in the image, there is an ambiguity as to in which order target digits should be swapped in the ground truth. To account for this ambiguity and thanks to the synthetic nature of the dataset, we instead build two ground truth sequences for both possible digit swap permutations, and report the lowest metric between the generated sequence and both ground truths (i.e., we choose the closest ground truth to compare to with respect to the considered metric).

B.4.4 3D Warehouse Chairs

This multi-view dataset introduced by Aubry et al. 2014 contains 1393 three-dimensional models of chairs seen under the same periodic angles. We resize

the original 600×600 images by center-cropping them to 400×400 images, and downsample them to 64×64 frames using the Lanczos filter of the Pillow library. [2]

We create sequences from this dataset for our model by assembling the views of each chair to simulate its rotation from right to left until it reaches its initial position. This process is repeated for each existing angle to serve as initial position for all chairs. We chose this dataset instead of Denton and Birodkar 2017's multi-view chairs dataset because the latter contains too few objects to allow both tested methods to generalize on the testing set, preventing us to draw any conclusion from the experiment. We train models on this dataset with $\tau + 1 = 5$ conditioning frames and $\nu - \tau = 10$ frames to predict, and test them to predict 15 frames within the content swap experiment. Training and testing data are constituted by randomly selecting 85% of the chairs for training and 15% of the remaining ones for testing. Disentanglement metrics are computed similarly to the ones on Moving MNIST, but with only one reference ground truth corresponding to the chair given as content input at the position of the chair given as dynamic input.

B.4.5 TaxiBJ

This crowd flow dataset provided by Zhang et al. 2017 consists in two-channel 32×32 frames representing the inflow and outflow of taxis in Beijing, each pixel corresponding to a square region of the city. Observations are registered every thirty minutes. It is highly structured as the flows are dependent on the infrastructure of the city, and complex since methods have to account for non-local dependencies and model subtle changes in the evolution of the flows.

We follow the preprocessing steps of Y. Wang et al. 2018 and Le Guen and Thome 2020 by performing a min-max normalization of the data to the $[0, 1]$ range. We train models on this dataset with $\tau + 1 = 4$ conditioning frames and $\nu - \tau = 4$ frames to predict, and test them to predict 4 frames like our competitors on the last four weeks of data which are excluded from the training set. MSE on this dataset is reported in the $[0, 1]$-normalized space and multiplied by a hundred times the dimensionality of a frame, i.e. by $100 \times 32 \times 32 \times 2$.

B.5 Training Details

Along with the code in the supplementary material, we provide in this section sufficient details in order to replicate our results.

2. https://pillow.readthedocs.io/

B.5.1 Reproduction of Baselines

PKnl. We retrained PKnl (Bézenac et al. 2018a) on SST using the official implementation and the indicated hyperparameters.

SVG, MIM and DDPAE. We trained SVG (Denton and Fergus 2018), MIM (Y. Wang et al. 2019b) and DDPAE (Hsieh et al. 2018) on our version of Moving MNIST using the official implementation and the same hyperparameters that the authors used for the original version of Moving MNIST.

We trained MIM on SST using the recommended hyperparameters of the authors, and SVG by retaining the same hyperparameters as those used on KTH.

DrNet. We trained DrNet (Denton and Birodkar 2017) on our version of Moving MNIST using the same hyperparameters originally used for the alternative version of the dataset on which it was originally trained (with digits of different colors). To this end, we reimplemented the official Lua implementation into a Python code in order to train it with a more recent infrastucture. We also trained DrNet on 3D Warehouse Chairs using the same hyperparameters used by its authors on the smaller multi-view chairs dataset on which they trained their method.

PhyDNet. We trained PhyDNet (Le Guen and Thome 2020) on SST and our version of Moving MNIST using the official implementation and the same hyperparameters that the authors used for SST and the original version of Moving MNIST. We removed the skip connections used by the authors on the Moving MNIST dataset in order to perform a fairer comparison with other models, such as ours, in our experimental study that do not incorporate skip connections on this dataset.

B.5.2 Model Specifications

Implementation

We used Python 3.8.1 and PyTorch 1.4.0 (Paszke et al. 2019a) to implement our model. Each model was trained on an Nvidia GPU with CUDA 10.1. Training is done with mixed-precision training (Micikevicius et al. 2018) thanks to the Apex library. [3]

3. https://github.com/nvidia/apex.

Architecture

Combination of S and T. As explained in section 6.4, the default choice of combination of S and T as decoder inputs is the concatenation of both vectorial variables: it is generic, and allows the decoder to learn an appropriate combination function ζ as in eq. (6.4).

Nonetheless, further knowledge of the studied dataset can help to narrow the choices of combination functions. Indeed, we choose to multiply S and T before giving them as input to the decoder for both datasets WaveEq and WaveEq-100, given the knowledge of the existence of product-separable solutions to the homogeneous version of equation (i.e., without source). This shows that it is possible to change the combination function of S and T, and that existing combination functions in the PDE literature could be leveraged for other datasets.

Encoders E_S and E_T, and decoder D. For WaveEq, the encoder and decoder outputs are considered to be vectors; images are thus flattened before encoding and reshaped after decoding to 64×64 frames. The encoder is a MultiLayer Perceptron (MLP) with two hidden layers of size 1200 and internal ReLU activation functions. The decoder is an MLP with three hidden layers of size 1200, internal ReLU activation functions, and a final sigmoid activation function for the decoder. The encoder and decoder used for WaveEq-100 are similar to those used for WaveEq, but with two hidden layers each, of respective sizes 2400 and 150.

We used for SST a VGG16 architecture (Simonyan and Zisserman 2015), mirrored between the encoder and the decoder, complemented with skip connections integrated into S (Ronneberger et al. 2015) from all internal layers of the encoder to corresponding decoder layers, also leveraged by Bézenac et al. 2018a in their PKnl model. We adapted this VGG16 architecture without skip connections for the 32×32 frames of TaxiBJ by removing the shallowest upsampling and downsampling operations in the VGG encoder and decoder. For Moving MNIST, the encoder and its mirrored decoder are shaped with the DCGAN discriminator and generator architecture (Radford et al. 2016), with an additional sigmoid activation after the very last layer of the decoder; this encoder and decoder DCGAN architecture is also used by DrNet and DDPAE. We highlight that we leveraged in both SST and Moving MNIST architectural choices that are also used in compared baselines, enabling fair comparisons.

For the two-dimensional latent space experiments on SST (see appendix B.6.3), we use a modified version of the VGG encoder / decoder network by removing the two deepest maximum pooling layers, thus preserving the two-dimensional latent structures. The decoder mirrors the encoder complemented with skip connections.

Regarding 3D Warehouse Chairs, we also followed the same architectural choices as DrNet with a ResNet18-like architecture for the encoders and a DCGAN architecture, followed by a sigmoid activation after the last layer for the decoder.

Encoders E_S and E_T taking as input multiple observations, we combine them by either concatenating them for the vectorial observations of WaveEq-100, or grouping them on the color channel dimensions for the other datasets where observations are frames. Each encoder and decoder layer was initialized from a normal distribution with standard deviation 0.02 (except for biases initialized to 0, and batch normalizations weights drawn from a Gaussian distribution with unit mean and a standard deviation of 0.02).

ODE solver. Following the recent line of work assimilating residual networks (K. He et al. 2016) with ODE solvers (Y. Lu et al. 2018; R. T. Q. Chen et al. 2018), we use a residual network as an integrator for eq. (6.7). This residual network is composed of a given number K of residual blocks, each block $i \in [\![1, K]\!]$ implementing the application $\mathrm{id} + g_i$, where g_i is an MLP with a two hidden layers of size H and internal ReLU activation functions. The parameter values for each dataset are:

- WaveEq and WaveEq-100: $K = 3$ and $H = 512$;
- SST (with linear latent states): $K = 3$ and $H = 1024$;
- Moving MNIST, 3D Warehouse Chairs and TaxiBJ: $K = 1$ and $H = 512$.

Each MLP is orthogonally initialized with the following gain for each dataset:

- WaveEq, WaveEq-100, SST (with linear latent states), 3D Warehouse Chairs and TaxiBJ: 0.71;
- Moving MNIST: 1.41.

For SST with two-dimensional states, the MLPs are replaced by convolutional layers with kernel size 3, padding 1 and a number of hidden channels equal to $H = 128$. We set $K = 2$ and an orthogonal initialization gain of 0.2. ReLU activations are replaced by Leaky ReLU activations and preceded by batch normalization layers.

Latent variable sizes. S and T have the following vectorial dimensions for each dataset:

- WaveEq and WaveEq-100: 32;

- SST, respectively $196 \times 16 \times 16$ and $64 \times 16 \times 16$; for the linear version, both are set to 256.
- Moving MNIST and TaxiBJ: respectively, 128 and 20;
- 3D Warehouse Chairs: respectively, 128 and 10.

Note that, in order to perform fair comparisons, the size of T for baselines without static component S is chosen to be the sum of the vectorial sizes of S and T in the full model. The skip connections of S for SST cannot, however, be integrated into T, as its evolution is only modeled in the latent space, and it is out of the scope of this paper to leverage low-level dynamics.

B.5.3 Optimization

Optimization is performed using the Adam optimizer (Diederik P. Kingma and Ba 2015) with initial learning rate 4×10^{-4} for WaveEq, WaveEq-100, Moving MNIST, 3D Warehouse Chairs and SST and 4×10^{-5} for TaxiBJ, and with decay rates $\beta_1 = 0.9$ (except for the experiments on Moving MNIST where we choose $\beta_1 = 0.5$) and $\beta_2 = 0.99$.

Loss function. Chosen coefficients values of λ_{pred}, λ_{AE}, λ^S_{reg}, and λ^T_{reg} are the following:

- $\lambda_{\text{pred}} = 45$;
- $\lambda_{\text{AE}} = 45$ for TaxiBJ; 10 for SST (linear)and Moving MNIST; 1 for WaveEq, WaveEq-100 and 3D Warehouse Chairs; 0.1 for SST;
- $\lambda^S_{\text{reg}} = 100$ for SST; $\lambda^S_{\text{reg}} = 45$ for WaveEq, WaveEq-100, SST (linear) and Moving MNIST; 1 for 3D Warehouse Chairs; 0.0001 for TaxiBJ;
- $\lambda^T_{\text{reg}} = \frac{1}{2}p \times 10^{-3}$ for WaveEq, WaveEq-100, Moving MNIST, 3D Warehouse Chairs and TaxiBJ (where p is the dimension of T); $\frac{1}{2}p \times 10^{-2}$ for SST (linear); 5×10^{-6} for SST.

The batch size is chosen to be 128 for WaveEq, WaveEq-100, Moving MNIST and 3D Warehouse Chairs, and 100 for SST and TaxiBJ.

Training length. The number of training epochs for each dataset is:

- WaveEq and WaveEq-100: 250 epochs;

- SST: 30 epochs; SST (linear): 80 epochs;
- Moving MNIST: 800 epochs, with an epoch corresponding to $200\,000$ trajectories (the dataset being infinite), and with the learning rate successively divided by 2 at epochs 300, 400, 500, 600, and 700;
- 3D Warehouse Chairs: 120 epochs;
- TaxiBJ: 550 epochs, with the learning rate divided by 5 at epochs 250, 300, 350, 400 and 450.

B.5.4 Prediction Offset for SST

Using the formalism of our work, our algorithm trains to reconstruct $v = (v_{t_0}, \dots, v_{t_1})$ from conditioning frames $V_\tau(t_0)$. Therefore, it first learns to reconstruct $V_\tau(t_0)$.

However, the evolution of SST data is chaotic and predicting above an horizon of 6 with coherent and sharp estimations is challenging. Therefore, for the SST dataset only, we chose to supervise the prediction from $t = t_0 + (\tau + 1)\Delta t$, i.e, our algorithm trains to forecast $v_{t_0+(\tau+1)\Delta t}, \dots, v_{t_1}$ from $V_\tau(t_0)$. It simply consists in making the temporal representation $E_T\big(V_\tau(t_0)\big)$ match the observation $v_{t_0+(\tau+1)\Delta t}$ instead of v_{t_0}. This index offset does not change our interpretation of spatiotemporal disentanglement through separation of variables.

B.6 Additional Results and Samples

B.6.1 Ablation Study on Moving MNIST

We report in table B.1 the results of an ablation study of our model on Moving MNIST, that we comment in section 6.5.2.

B.6.2 Preliminary Results on KTH

The application of our method to natural videos is an interesting perspective, but would motivate further adaptation of the model (see perspectives in the conclusion), in particular regarding the integration of stochastic dynamics. Indeed, there is a consensus in the literature (e.g.: Denton and Fergus 2018; Villegas et al. 2019; Weissenborn et al. 2020) indicating that human motion datasets require

Table B.1. – Prediction and content swap PSNR and SSIM scores of variants of our model.

Models	Pred. $(t+10)$		Pred. $(t+95)$		Swap $(t+10)$		Swap $(t+95)$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Ours	**21.70**	**0.9088**	**17.50**	**0.7990**	**18.42**	**0.8368**	**16.50**	**0.7713**
Ours (without S)	20.46	0.8867	14.95	0.6707	—	—	—	—
Ours ($\lambda_{\text{AE}} = 0$)	21.61	0.9058	16.58	0.7611	18.21	0.8309	15.79	0.7399
Ours ($\lambda^{S}_{\text{reg}} = 0$)	15.99	0.6900	12.31	0.5702	13.73	0.5476	12.07	0.5556
Ours ($\lambda^{T}_{\text{reg}} = 0$)	15.63	0.7369	14.02	0.7253	14.91	0.7154	13.95	0.7234
Ours (GRU)	21.66	0.9088	15.45	0.4888	17.70	0.8178	14.77	0.4718

Table B.2. – FVD score of compared models on KTH. The bold score indicates the best performing method.

Ours	PhyDNet	SVG	DrNet
330	384	375	383

stochastic modeling because of the inherently highly random events occurring in these videos. Tackling this issue would require to incorporate stochasticity in our model, for example leveraging variational autoencoders like Denton and Fergus 2018, or supplement it with adversarial losses on the image space, for instance like Mathieu et al. 2016b and A. X. Lee et al. 2018. These changes are feasible, but are out of the scope of this paper.

Nonetheless, we investigate the realistic video dataset KTH (Schüldt et al. 2004), which is an action recognition video database featuring various subjects performing actions in front of different backgrounds. We trained our model, SVG, DrNet and PhyDNet on this dataset. DrNet and PhyDNet are powerful deterministic approaches, while SVG is a standard stochastic video prediction model. We compare all models in terms of FVD (Unterthiner et al. 2018, lower is better), which is a metric based on deep features that evaluates the realism of the generated videos.

Results are reported in table B.2. We observe that our model substantially outperforms the considered baselines. These significant results against powerful deterministic baselines, and even the standard stochastic method SVG, confirm our advantage at modeling complex dynamics and support our claim that our model lays the foundations for domain-specific methods, such as a stochastic version for natural videos.

Reproductibility. We use the following training parameters for KTH:

- we follow the same dataset processing and evaluation procedure as Denton and Fergus 2018;
- we train our model on 125 epochs with batch size 100, with an epoch being defined as $100\,000$ training sequences;
- we set the learning rate to 2×10^{-4} and the same optimizer parameters as for SST;
- $\lambda_{\text{pred}} = 45$, $\lambda_{\text{AE}} = 10 = \lambda^S_{\text{reg}} = 10$, $\lambda^T_{\text{reg}} = p \times 10^{-4}$;
- the size of S and T are respectively 128 and 50;
- the ODE is solved with a flat latent architecture and parameters $K = 1$ and $H = 512$;
- the encoder and decoder architecture is VGG16 with skip connections integrated into S from E_S to D, and with the decoder output being given to a final sigmoid activation.

We reproduced SVG, DrNet and PhyDNet using the recommended hyperparameters of their authors. We trained PhyDNet for 125 epochs, like our model, to obtain a fair evaluation despite its low efficiency (six times slower than ours).

B.6.3 Modeling SST with Separation of Variables

We present in table B.3 results of table 6.1 for SST, complemented with an alternative version of our model obtained using vectorial representation for S and T and MLPs to compute the derivative of T. The latter setting corresponds to a strictly enforced separation of spatial and dynamical variables, with results significantly outperforming powerful methods PhyDNet, PKnl and SVG thanks to this separation, as attested by the corresponding ablation without a static component.

However, sea surface temperature exhibits highly local structure that can be assimilated to a flow in a coarse approximation. For example, there is transport of large bodies of hot and cold water. Accordingly, performances may be enhanced by considering local dependencies in the dynamics, as also implemented by MIM and PhyDNet. We propose to do so by considering like the latter methods two-dimensional latent states for the static S and the dynamical T, and convolutional networks to model the derivative of T.

Table B.3. – Forecasting performance on SST of PKnl, PhyDNet and our model with respect to indicated prediction horizons. Bold scores indicate the best performing method.

Models	MSE		SSIM	
	$t+6$	$t+10$	$t+6$	$t+10$
PKnl	1.28	2.03	0.6686	0.5844
PhyDNet	1.27	1.91	0.5782	0.4645
SVG	1.51	2.06	0.6259	0.5595
MIM	0.91	1.45	0.7406	0.6525
Ours	**0.86**	**1.43**	**0.7466**	**0.6577**
Ours (without S)	0.95	1.50	0.7204	0.6446
Ours (linear)	1.15	1.80	0.6837	0.5984
Ours (linear, without S)	1.46	2.19	0.6200	0.5456

Accounting for such locality in the dynamics amounts to implementing another separation than the usual separation between t and spatial variables. Indeed, it rather excludes unknown content variables from the dynamics. The resulting dynamics is then a PDE over time t and the observation coordinates x and y that we implement using convolutional neural networks, following Long et al. 2018 and Ayed et al. 2020. This different kind of separation of variables simplifies learning by estimating a PDE that is simpler than the original one, since it acts on fewer variables. It highlights the generality of our intuition of using the separation of variables, which may be used in other settings that strict spatiotemporal disentanglement. This approach, while still maintaining disentangling properties, significantly improves prediction performances.

Note that our proposition remains computationally much lighter than the alternatives MIM, PhyDNet and SVG.

B.6.4 Additional Samples

WaveEq

We provide in fig. B.1 a sample for the WaveEq dataset, highlighting the long-term consistency in the forecasts of our algorithm.

We also show in fig. B.2 the effect in forecasting of changing the spatial code S from the one of another sequence.

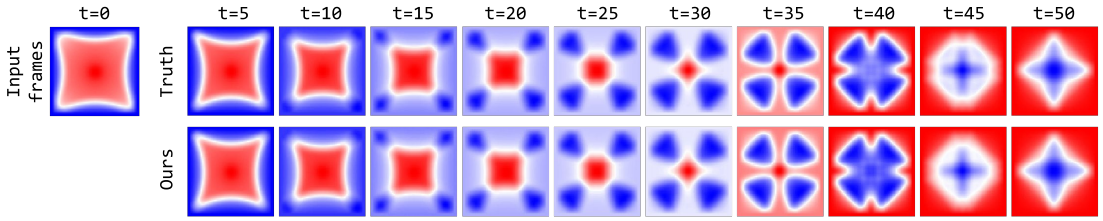


Figure B.1. – Example of predictions of our model on WaveEq.

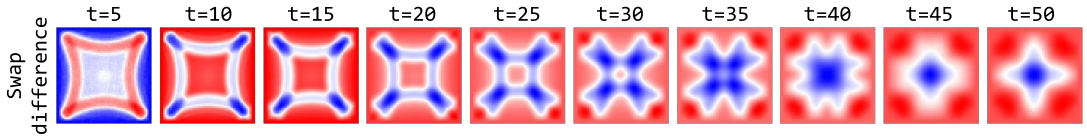


Figure B.2. – Evolution of the scaled difference between the forecast of a sequence and the same forecast with a spatial code coming from another sequence for the WaveEq dataset.

SST

We provide an additional sample for SST in fig. B.3.

Moving MNIST

We provide two additional samples for Moving MNIST in figs. B.4 and B.5.

3D Warehouse Chairs

We provide a qualitative comparison for the content swap experiment between our model and DrNet for 3D Warehouse Chairs in fig. B.7. We notice that DrNet produces substantially more blurry samples than our model and has difficulties to capture the exact dynamic of the chairs.

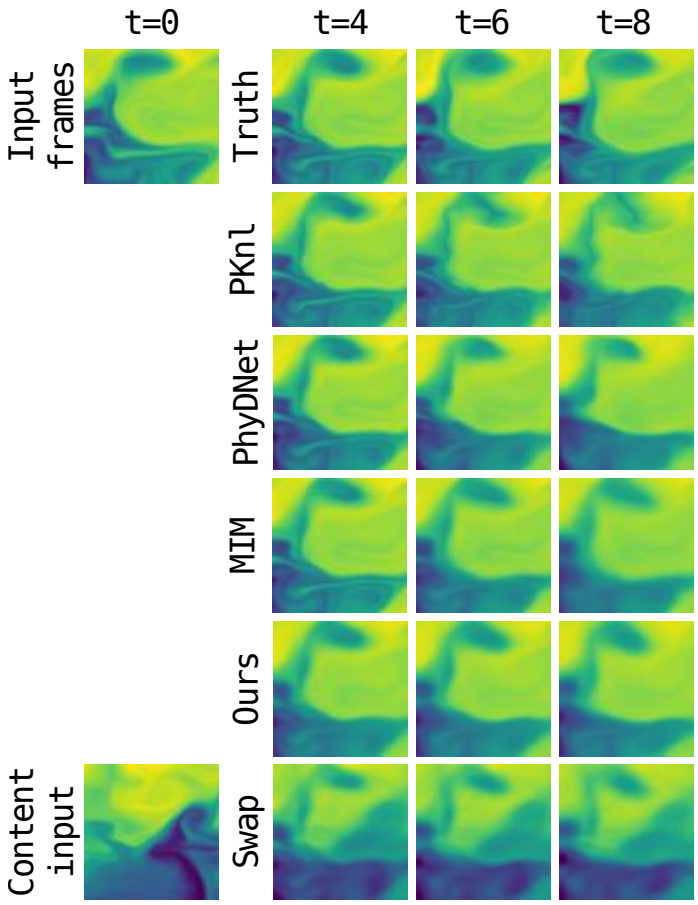


Figure B.3. – Example of predictions of compared models on SST.

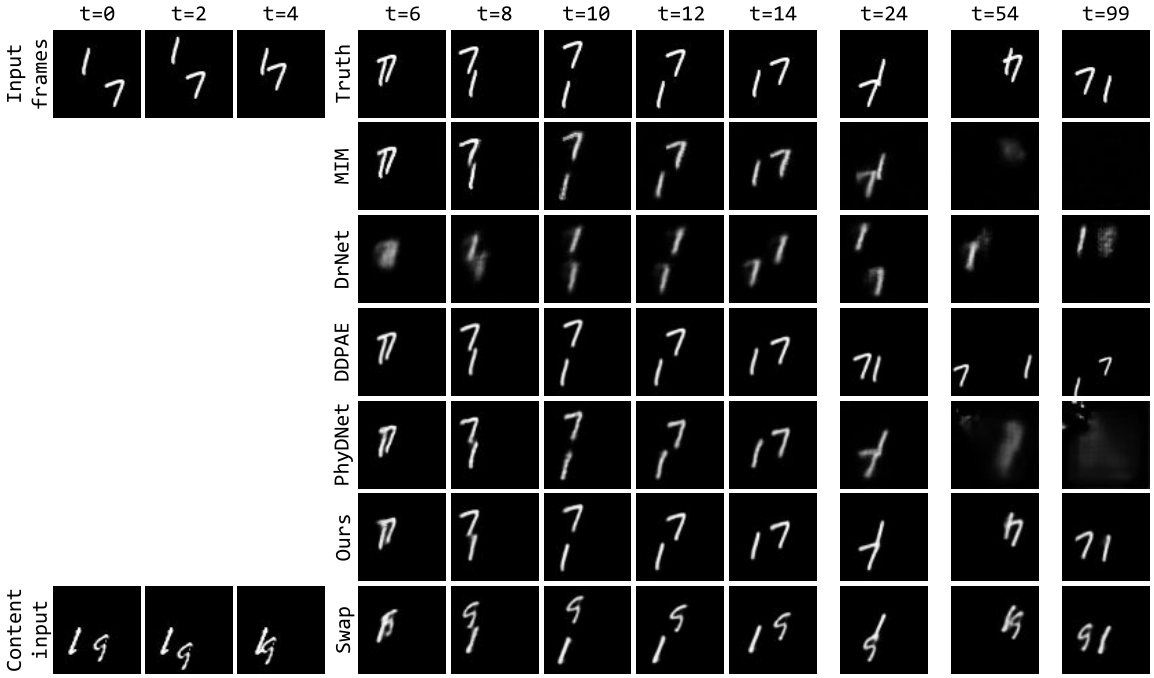


Figure B.4. – Example of predictions of compared models on Moving MNIST.

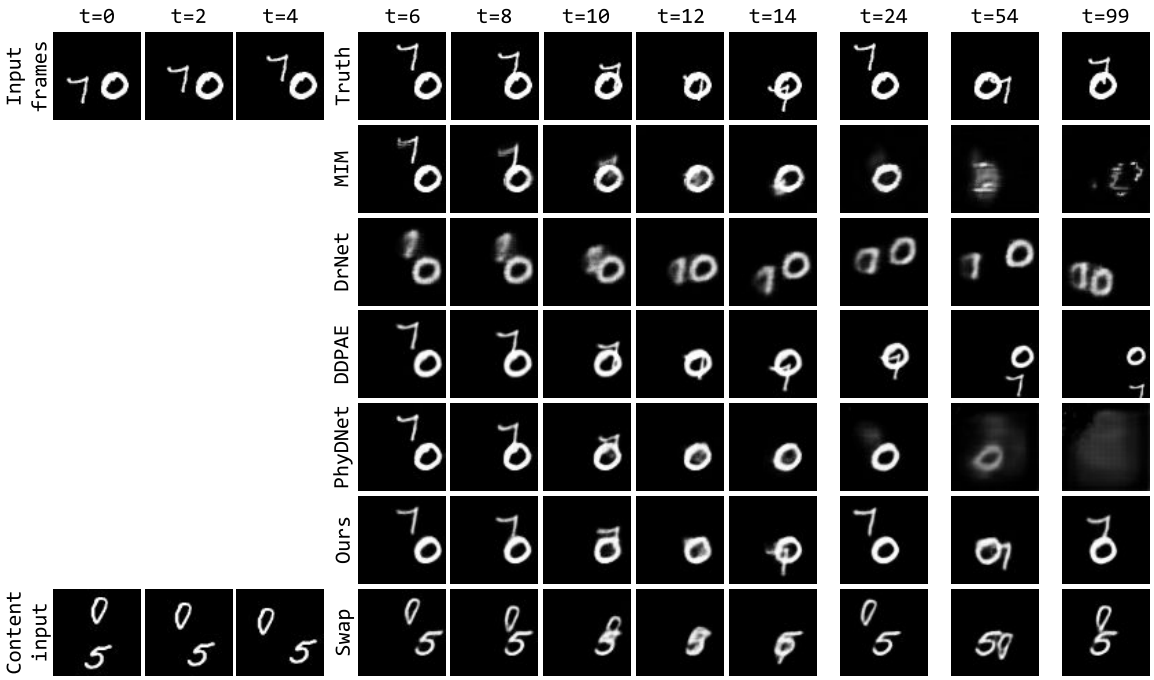


Figure B.5. – Example of predictions of compared models on Moving MNIST.

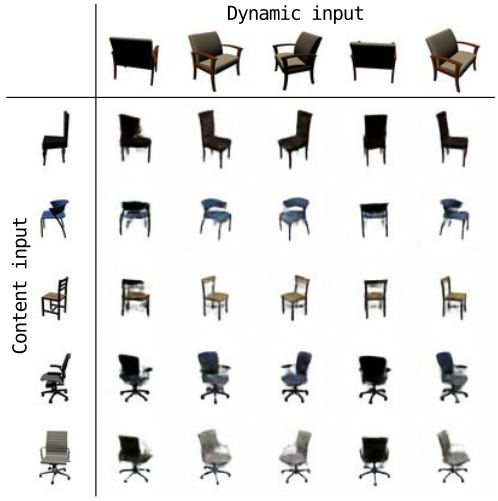


Figure B.6. – Fusion of content and dynamic variables in Dr Net on 3D Warehouse Chairs.

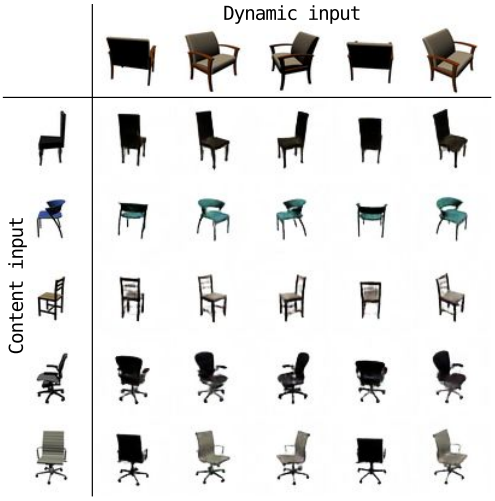


Figure B.7. – Fusion of content and dynamic variables in Our model on 3D Warehouse Chairs.

APPENDIX C

APPENDICES TO APHYNITY

C.1 A reminder on Chebyshev sets

We begin by giving a definition of Chebyshev sets, taken from (Fletcher and Moors 2014):

Definition C.1. A *Chebyshev set* of a normed space $(E, \|\cdot\|)$ is a subset $\mathcal{C} \subset E$ such that every $x \in E$ admits a unique nearest point in $\mathcal{C}$.

In Euclidean spaces, Chebyshev sets are simply the closed convex subsets. The question of knowing whether it is the case that all Chebyshev sets are closed convex sets in infinite dimensional Hilbert spaces is still an open question. In general, there exists examples of non-convex Chebyshev sets, a famous one being presented in (Johnson 1987) for a non-complete inner-product space.

Given the importance of this topic in approximation theory, finding necessary conditions for a set to be Chebyshev and studying the properties of those sets have been the subject of many efforts. Some of those properties are summarized below:

- The metric projection on a boundedly compact Chebyshev set is continuous.
- If the norm is strict, every closed convex space, in particular any finite dimensional subspace is Chebyshev.
- In a Hilbert space, every closed convex set is Chebyshev.

C.2 Proof of Existance and Uniqueness

We prove the following result: We prove the following result which implies both propositions in the article:

Proposition C.2. *The optimization problem:*

$$\min_{f_k \in \mathcal{H}_k, f_u \in \mathcal{F}} \|f_u\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{\mathrm{d}X_t}{\mathrm{d}t} = (f_k + f_u)(X_t) \tag{C.1}$$

is equivalent a metric projection onto $\mathcal{H}_k$.

If $\mathcal{H}_k$ is proximinal, (C.1) admits a minimizing pair.

If $\mathcal{H}_k$ is Chebyshev, (C.1) admits a unique minimizing pair which F_p is the metric projection.

Proof. The idea is to reconstruct the full functional from the trajectories of $\mathcal{D}$. By definition, $\mathcal{A}$ is the set of points reached by trajectories in $\mathcal{D}$ so that:

$$\mathcal{A} = \{x \in \mathbb{R}^d \mid \exists X_\cdot \in \mathcal{D}, \exists t,\ X_t = x\}$$

Then let us define a function $F^{\mathcal{D}}$ in the following way: For $a \in \mathcal{A}$, we can find $X_\cdot \in \mathcal{D}$ and t_0 such that $X_{t_0} = a$. Differentiating X at t_0, which is possible by definition of $\mathcal{D}$, we take:

$$F^{\mathcal{D}}(a) = \left.\frac{dX_t}{dt}\right|_{t=t_0}$$

For any (f_k, f_u) satisfying the constraint in (C.1), we then have that $(f_k + f_u)(a) = \frac{dX_t}{dt}_{|t_0} = F^{\mathcal{D}}(a)$ for all $a \in \mathcal{A}$. Conversely, any pair such that $(f_k, f_u) \in \mathcal{H}_k \times \mathcal{F}$ and $f_k + f_u = F^{\mathcal{D}}$, verifies the constraint.

As $\mathcal{H}_k$ is a Chebyshev set, the optimization problem:

$$\min_{f_k \in \mathcal{H}_k} \|F^{\mathcal{D}} - f_k\|$$

has a unique minimum which is the projection of $F^{\mathcal{D}}$ on $\mathcal{H}_k$ and which we denote $f_k^\star$. Taking $f_u^\star = F^{\mathcal{D}} - f_k^\star$, we have that $f_k^\star + f_u^\star = F^{\mathcal{D}}$ so that $(f_k^\star, f_u^\star)$ verifies the constraint of (8.2). Moreover, if there is (f_k, f_u) satisfying the constraint of eq. (8.2), we have that $f_k + f_u = F^{\mathcal{D}}$ by what was shown above and $\|f_u\| = \|F^{\mathcal{D}} - f_k\| \geq \|F^{\mathcal{D}} - f_k^\star\|$ by definition of $f_k^\star$. This shows that $(f_k^\star, f_u^\star)$ is minimal. Finally, by uniqueness of the projection, if $f_k \neq f_k^\star$ then $\|f_u\| > \|f_u^\star\|$. Thus the minimal pair is unique. □

C.3 Parameter Estimation in Incomplete Physical Models

Classically, when a set $\mathcal{H}_k \subset \mathcal{F}$ summarising the most important properties of a system is available, this gives a *simplified model* of the true dynamics and the

adopted problem is then to fit the trajectories using this model as well as possible, solving:

$$\min_{f_k \in \mathcal{H}_k} \mathbb{E}_{X \sim \mathcal{D}} L(\widetilde{X}^{X_0}, X) \tag{C.2}$$

$$\text{subject to } \forall g \in \mathcal{I},\ \widetilde{X}_0^g = g \ \text{ and } \ \forall t,\ \frac{d\widetilde{X}_t^g}{dt} = f_k(\widetilde{X}_t^g) \tag{C.3}$$

where L is a discrepancy measure between trajectories. Recall that $\widetilde{X}^{X_0}$ is the result trajectory of an ODE solver taking X_0 as initial condition. In other words, we try to find a function f_k which gives trajectories as close as possible to the ones from the dataset. While estimation of the function becomes easier, there is then a residual part which is left unexplained and this can be a non negligible issue in at least two ways:

- When the dynamics $f \notin \mathcal{H}_k$, the loss is strictly positive at the minimum. This means that reducing the space of functions makes us lose in terms of accuracy. [1]

- The obtained function f_k might not even be the most meaningful function from $\mathcal{H}_k$ as it would try to capture phenomena which are not explainable with functions in $\mathcal{H}_k$, thus giving the wrong bias to the calculated function. For example, if one is considering a dampened periodic trajectory where only the period can be learned in $\mathcal{H}_k$ but not the dampening, the estimated period will account for the dampening and will thus be biased.

This is confirmed in section 8.4: the incomplete physical models augmented with APHYNITY get different and experimentally better physical identification results than the physical models alone.

Let us compare our approach with this one on the linearized damped pendulum to show how estimates of physical parameters can differ. The equation is the following:

$$\frac{d^2\theta}{dt^2} + \omega_0^2\theta + \lambda\frac{d\theta}{dt} = 0$$

We take the same notations as in the article and parametrize the simplified physical models as:

$$f_k^a : X \mapsto (\frac{d\theta}{dt}, -a\theta)$$

1. This is true in theory, although not necessarily in practice when f overfits a small dataset.

where $a > 0$ corresponds to ω_0^2. The corresponding solution for an initial state X_0, which we denote X^a, can then written explicitly as:

$$\theta_t^a = \theta_0 \cos \sqrt{a}\, t$$

Let us consider damped pendulum solutions X written as:

$$\theta_t = \theta_0 e^{-t} \cos t$$

which corresponds to:

$$F : X \mapsto (\frac{d\theta}{dt}, 2(\theta - \frac{d\theta}{dt}))$$

It is then easy to see that the estimate of a with the physical model alone can be obtained by minimizing:

$$\int_0^T |e^{-t} \cos t - \cos \sqrt{a}\, t|^2$$

This expression depends on T and thus, depending on the chosen time interval and the way the integral is discretized will almost always give biased estimates. In other words, the estimated value of a won't give us the desired solution $t \mapsto \cos t$.

On the other hand, for a given a, in the APHYNITY framework, the residual must be equal to:

$$F_r^a : X \mapsto (0, (2 - a)\theta - 2\frac{d\theta}{dt})$$

in order to satisfy the fitting constraint. Here a corresponds to $1 + \omega_0^2$ not to ω_0^2 as in the simplified case. Minimizing its norm, we obtain $a = 2$ which gives us the desired solution:

$$\theta_t = \theta_0 e^{-t} \cos t$$

with the right period.

C.4 Discussion of Supervision over Derivatives

In order to find the appropriate decomposition (f_k, f_u), we use a trajectory-based error by solving:

$$\min_{f_k \in \mathcal{H}_k, f_u \in \mathcal{F}} \|f_u\| \tag{C.4}$$

$$\text{subject to: } \forall g \in \mathcal{I},\ \widetilde{X}_0^g = g \text{ and } \forall t,\ \frac{d\widetilde{X}_t^g}{dt} = (f_k + f_u)(\widetilde{X}_t^g)$$

$$\text{subject to: } \forall X \in \mathcal{D},\ L(X, \widetilde{X}^{X_0}) = 0$$

In the continuous setting where the data is available at all times t, this problem is in fact equivalent to the following one:

$$\min_{f_k \in \mathcal{H}_k} \mathop{\mathbb{E}}_{X \sim \mathcal{D}} \int \| \frac{dX_t}{dt} - f_k(X_t) \| \tag{C.5}$$

where the supervision is done directly over derivatives, obtained through finite-difference schemes. This echoes the proof in appendix C.2 of the Supplementary where f can be reconstructed from the continuous data.

However, in practice, data is only available at discrete times with a certain time resolution. While eq. (C.5) is indeed equivalent to eq. (C.4) in the continuous setting, in the practical discrete one, the way error propagates isn't anymore: For eq. (C.4) it is controlled over integrated trajectories while for eq. (C.5) the supervision is over the approximate derivatives of the trajectories from the dataset. We argue that the trajectory-based approach is more flexible and more robust for the following reasons:

- In (C.4), if f_u is appropriately parameterized, it is possible to perfectly fit the data trajectories at the sampled points.

- The use of finite differences schemes to estimate f as is done in (C.5) necessarily induces a non-zero discretization error.

- This discretization error is explosive in terms of divergence from the true trajectories.

This last point is quite important, especially when time sampling is sparse (even though we do observe this adverse effect empirically in our experiments with relatively finely time-sampled trajectories). The following gives a heuristical reasoning as to why this is the case. Let $\widetilde{F} = f + \epsilon$ be the function estimated from the sampled points with an error ϵ such that $\|\epsilon\|_\infty \leq \alpha$. Denoting $\widetilde{X}$ the corresponding trajectory generated by $\widetilde{F}$, we then have, for all $X \in \mathcal{D}$:

$$\forall t, \ \frac{d(X - \widetilde{X})_t}{dt} = f(X_t) - f(\widetilde{X}_t) - \epsilon(\widetilde{X}_t)$$

Integrating over $[0, T]$ and using the triangular inequality as well as the mean value inequality, supposing that f has uniformly bounded spatial derivatives:

$$\forall t \in [0, T], \ \|(X - \widetilde{X})_t\| \leq \|\nabla f\|_\infty \int_0^t \|X_s - \widetilde{X}_s\| + \alpha t$$

which, using a variant of the Grönwall lemma, gives us the inequality:

$$\forall t \in [0, T],\ \|X_t - \widetilde{X}_t\| \leq \frac{\alpha}{\|\nabla f\|_\infty}(\exp(\|\nabla f\|_\infty t) - 1)$$

When α tends to 0, we recover the true trajectories X. However, as α is bounded away from 0 by the available temporal resolution, this inequality gives a rough estimate of the way $\widetilde{X}$ diverges from them, and it can be an equality in many cases. This exponential behaviour explains our choice of a trajectory-based optimization.

C.5 Implementation Details

We describe here the three use cases studied in the paper for validating APHYNITY. All experiments are implemented with PyTorch Paszke et al. 2019b and the differentiable ODE solvers with the adjoint method implemented in `torchdiffeq`.[2]

C.5.1 Non-linear pendulum

We consider the non-linear damped pendulum problem, governed by the ODE

$$\frac{d^2\theta}{dt^2} + \omega_0^2 \sin\theta + \lambda \frac{d\theta}{dt} = 0$$

where $\theta(t)$ is the angle, $\omega_0 = \frac{2\pi}{T_0}$ is the proper pulsation (T_0 being the period) and λ is the damping coefficient. With the state $X = (\theta, \frac{d\theta}{dt})$, the ODE can be written as $\frac{dX_t}{dt} = f(X_t)$ with $f : X \mapsto (\frac{d\theta}{dt}, -\omega_0^2 \sin\theta - \lambda \frac{d\theta}{dt})$.

Dataset For each train / validation / test split, we simulate a dataset with 25 trajectories of 40 timesteps (time interval $[0, 20]$, timestep $\delta t = 0.5$) with fixed ODE coefficients ($T_0 = 12, \lambda = 0.2$) and varying initial conditions. The simulation integrator is Dormand-Prince Runge-Kutta method of order (4)5 (DOPRI5) Dormand and Prince 1980. We also add a small amount of white gaussian noise ($\sigma = 0.01$) to the state. Note that our pendulum dataset is much more challenging than the ideal frictionless pendulum considered in Greydanus et al. 2019.

Neural network architectures We detail in Table C.1 the neural architectures used for the damped pendulum experiments. All data-driven augmentations

2. https://github.com/rtqichen/torchdiffeq

for approximating the mapping $X_t \mapsto f(X_t)$ are implemented by multi-layer perceptrons (MLP) with 3 layers of 200 neurons and ReLU activation functions (except at the last layer: linear activation). The Hamiltonian Greydanus et al. 2019 is implemented by a MLP that takes the state X_t and outputs a scalar estimation of the Hamiltonian $\mathcal{H}$ of the system: the derivative is then computed by an in-graph gradient of $\mathcal{H}$ with respect to the input: $f(X_t) = \left(\frac{\partial \mathcal{H}}{\partial(d\theta/dt)}, -\frac{\partial \mathcal{H}}{d\theta}\right)$.

Table C.1. – Neural network architectures for the damped pendulum experiments. n/a corresponds to non-applicable cases.

Method	Physical model	Data-driven model
Neural ODE R. T. Q. Chen et al. 2018	n/a	MLP(in=2, units=200, layers=3, out=2)
Hamiltonian Greydanus et al. 2019; Toth et al. 2020	MLP(in=2, units=200, layers=3, out=1)	n/a
APHYNITY Hamiltonian	MLP(in=2, units=200, layers=3, out=1)	MLP(in=2, units=200, layers=3, out=2)
Param ODE (ω_0)	1 trainable parameter ω_0	n/a
APHYNITY Param ODE (ω_0)	1 trainable parameter ω_0	MLP(in=2, units=200, layers=3, out=2)
Param ODE (ω_0, λ)	2 trainable parameters ω_0, λ	n/a
APHYNITY Param ODE (ω_0, λ)	2 trainable parameters ω_0, λ	MLP(in=2, units=200, layers=3, out=2)

Optimization hyperparameters The hyperparameters of the APHYNITY optimization algorithm ($Niter, \lambda_0, \tau_1, \tau_2$) were cross-validated on the validation set and are shown in Table C.2. All models were trained with a maximum number of 5000 steps with early stopping.

Table C.2. – Hyperparameters of the damped pendulum experiments.

Method	Niter	λ_0	τ_1	τ_2
APHYNITY Hamiltonian	5	1	1	0.1
APHYNITY ParamODE (ω_0)	5	1	1	10
APHYNITY ParamODE (ω_0, λ)	5	1000	1	100

C.5.2 Reaction-diffusion equations

The system is driven by a 2D FitzHugh-Nagumo type PDE Klaasen and Troy 1984

$$\frac{\partial u}{\partial t} = a\Delta u + R_u(u, v; k), \frac{\partial v}{\partial t} = b\Delta v + R_v(u, v)$$

where a and b are respectively the diffusion coefficients of u and v, Δ is the Laplace operator. The local reaction terms are $R_u(u, v; k) = u - u^3 - k - v, R_v(u, v) = u - v$.

The state $X = (u, v)$ is defined over a compact rectangular domain $\Omega = [-1, 1]^2$ with periodic boundary conditions. Ω is spatially discretized with a 32×32 2D uniform square mesh grid. The periodic boundary condition is implemented with

Table C.3. – ConvNet architecture in reaction-diffusion and wave equation experiments, used as data-driven derivative operator in APHYNITY and Neural ODE (R. T. Q. Chen et al. 2018).

Module	Specification
2D Conv.	3×3 kernel, 2 input channels, 16 output channels, 1 pixel zero padding
2D Batch Norm.	No average tracking
ReLU activation	—
2D Conv.	3×3 kernel, 16 input channels, 16 output channels, 1 pixel zero padding
2D Batch Norm.	No average tracking
ReLU activation	—
2D Conv.	3×3 kernel, 16 input channels, 2 output channels, 1 pixel zero padding

circular padding around the borders. Δ is systematically estimated with a 3×3 discrete Laplace operator.

Dataset Starting from a randomly sampled initial state $X_{\text{init}} \in [0,1]^{2\times 32\times 32}$, we generate states by integrating the true PDE with fixed a, b, and k in a dataset ($a = 1 \times 10^{-3}, b = 5 \times 10^{-3}, k = 5 \times 10^{-3}$). We firstly simulate high time-resolution ($\delta t_{\text{sim}} = 0.001$) sequences with explicit finite difference method. We then extract states every $\delta t_{\text{data}} = 0.1$ to construct our low time-resolution datasets.

We set the time of random initial state to $t = -0.5$ and the time horizon to $t = 2.5$. 1920 sequences are generated, with 1600 for training/validation and 320 for test. We take the state at $t = 0$ as X_0 and predict the sequence until the horizon (equivalent to 25 time steps) in all reaction-diffusion experiments. Note that the sub-sequence with $t < 0$ are reserved for the extensive experiments in Section C.7.1 of the Supplementary.

Neural network architectures Our f_u here is a 3-layer convolution network (ConvNet) in the APHYNITY. The two input channels are (u, v) and two output ones are $(\frac{\partial u}{\partial t}, \frac{\partial v}{\partial t})$. The purely data-driven Neural ODE uses such ConvNet as its F. The detailed architecture is provided in table C.3.

The estimated physical parameters θ_k in f_k are simply a trainable vector $(a, b) \in \mathbb{R}^2_+$ or $(a, b, k) \in \mathbb{R}^3_+$.

Optimization hyperparameters We choose to apply the same hyperparameters for all the reaction-diffusion experiments: $Niter = 1, \lambda_0 = 1, \tau_1 = 1 \times 10^{-3}, \tau_2 = 1 \times 10^3$.

C.5.3 Wave equations

The damped wave equation is defined by

$$\frac{\partial^2 w}{\partial t^2} - c^2 \Delta w + k \frac{\partial w}{\partial t} = 0$$

where c is the wave speed and k is the damping coefficient. The state is $X = (w, \frac{\partial w}{\partial t})$.

We consider a compact spatial domain Ω represented as a 64×64 grid and discretize the Laplacian operator similarly. Δ is implemented using a 5×5 discrete Laplace operator in simulation whereas in the experiment is a 3×3 Laplace operator. Null Neumann boundary condition are imposed for generation.

Dataset δt was set to 0.001 to respect Courant number and provide stable integration. The simulation was integrated using a 4th order finite difference Runge-Kutta scheme for 300 steps from an initial Gaussian state, i.e for all sequence at $t = 0$, we have:

$$w(x, y, t = 0) = C \times \exp^{\frac{(x-x_0)^2 + (y-y_0)^2}{\sigma^2}} \tag{C.6}$$

The amplitude C is fixed to 1, and $(x_0, y_0) = (32, 32)$ to make the Gaussian curve centered for all sequences. However, σ is different for each sequence and uniformly sampled in $[10, 100]$. The same δt was used for train and test. All initial conditions are Gaussian with varying amplitudes. 250 sequences are generated, 200 are used for training while 50 are reserved as a test set. In the main paper setting, $c = 330$ and $k = 50$. As with the reaction diffusion case, the algorithm takes as input a state $X_{t_0} = (w, \frac{dw}{dt})(t_0)$ and predicts all states from $t_0 + \delta t$ up to $t_0 + 25\delta t$.

Neural network architectures The neural network for f_u is a 3-layer convolution neural network with the same architecture as in table C.3. For f_k, the parameter(s) to be estimated is either a scalar $c \in \mathbb{R}_+$ or a vector $(c, k) \in \mathbb{R}_+^2$. Similarly, Neural ODE networks are build as presented in table C.3.

Optimization hyperparameters We use the same hyperparameters for the experiments: $Niter = 3, \lambda_0 = 1, \tau_1 = 1 \times 10^{-4}, \tau_2 = 1 \times 10^2$.

C.6 Ablation study

We conduct ablation studies to show the effectiveness of APHYNITY's adaptive optimization and trajectory-based learning scheme.

C.6.1 Ablation to vanilla MB/ML cooperation

In Table C.4, we consider the ablation case with the vanilla augmentation scheme found in (Le Guen and Thome 2020; Q. Wang et al. 2019; Mehta et al. 2020), which does not present any proper decomposition guarantee. We observe that the APHYNITY cooperation scheme outperforms this vanilla scheme in all case, both in terms of forecasting performances (e.g. log MSE= -0.35 vs. -3.97 for the Hamiltonian in the pendulum case) and parameter identification (e.g. Err Param=8.4% vs. 2.3 for Param PDE $(a, b$ for reaction-diffusion). It confirms the crucial benefits of APHYNITY's principled decomposition scheme.

C.6.2 Detailed ablation study

We conduct also two other ablations in Table C.5:

- *derivative supervision*: in which $f_k + f_u$ is trained with supervision over approximated derivatives on ground truth trajectory, as performed in (Greydanus et al. 2019; Cranmer et al. 2020). More precisely, APHYNITY's $\mathcal{L}_{traj}$ is here replaced with $\mathcal{L}_{deriv} = \|\frac{\mathrm{d}X_t}{\mathrm{d}t} - f(X_t)\|$ as in eq. (C.5), where $\frac{\mathrm{d}X_t}{\mathrm{d}t}$ is approximated by finite differences on X_t.

- *non-adaptive optim.*: in which we train APHYNITY by minimizing $\|f_u\|$ without the adaptive optimization of λ shown in Algorithm 8.1. This case is equivalent to $\lambda = 1, \tau_2 = 0$.

We highlight the importance to use a principled adaptive optimization algorithm (APHYNITY algorithm described in paper) compared to a non-adpative optimization: for example in the reaction-diffusion case, log MSE= -4.55 vs. -5.10 for Param PDE (a, b). Finally, when the supervision occurs on the derivative, both forecasting and parameter identification results are systematically lower than with APHYNITY's trajectory based approach: for example, log MSE=-1.16 vs. -4.64 for Param PDE (c) in the wave equation. It confirms the good properties of the APHYNITY training scheme.

Table C.4. – Ablation study comparing APHYNITY to the vanilla augmentation scheme (Q. Wang et al. 2019; Mehta et al. 2020) for the reaction-diffusion equation, wave equation and damped pendulum.

Dataset	Method	log MSE	%Err Param.	$\|f_u\|^2$
Reaction-diffusion	Param. PDE (a, b) with vanilla aug.	-4.56±0.52	8.4	(7.5±1.4)e1
	APHYNITY Param. PDE (a, b)	**-5.10±0.21**	**2.3**	(6.7±0.4)e1
	Param. PDE (a, b, k) with vanilla aug.	-8.04±0.03	25.4	(1.5±0.2)e-2
	APHYNITY Param. PDE (a, b, k)	**-9.35±0.02**	**0.096**	(1.5±0.4)e-6
	True PDE with vanilla aug.	-8.12±0.05	n/a	(6.1±2.3)e-4
	APHYNITY True PDE	**-9.17±0.02**	n/a	(1.4±0.8)e-7
Wave equation	Param PDE (c) with vanilla aug.	-3.90 ± 0.27	0.51	88.66
	APHYNITY Param PDE (c)	**-4.64±0.25**	**0.31**	71.0
	Param PDE (c, k) with vanilla aug.	-5.96 ± 0.10	0.71	25.1
	APHYNITY Param PDE (c, k)	**-6.09±0.28**	**0.70**	4.54
Damped pendulum	Hamiltonian with vanilla aug.	-0.35±0.1	n/a	837±117
	APHYNITY Hamiltonian	**-3.97±1.2**	n/a	623±68
	Param ODE (ω_0) with vanilla aug.	-7.02±1.7	4.5	148±49
	APHYNITY Param ODE (ω_0)	**-7.86±0.6**	**4.0**	132±11
	Param ODE (ω_0, α) with vanilla aug.	-7.60±0.6	4.65	35.5±6.2
	APHYNITY Param ODE (ω_0, α)	**-8.31±0.3**	**0.39**	8.5±2.0
	Augmented True ODE with vanilla aug.	**-8.40±0.2**	n/a	3.4±0.8
	APHYNITY True ODE	**-8.44±0.2**	n/a	2.3±0.4

C.7 Additional experiments

C.7.1 Reaction-diffusion systems with varying diffusion parameters

We conduct an extensive evaluation on a setting with varying diffusion parameters for reaction-diffusion equations. The only varying parameters are diffusion coefficients, i.e. individual a and b for each sequence. We randomly sample $a \in [1 \times 10^{-3}, 2 \times 10^{-3}]$ and $b \in [3 \times 10^{-3}, 7 \times 10^{-3}]$. k is still fixed to 5×10^{-3} across the dataset.

In order to estimate a and b for each sequence, we use here a ConvNet encoder E to estimate parameters from 5 reserved frames ($t < 0$). The architecture of the encoder E is similar to the one in Table C.3 except that E takes 5 frames (10 channels) as input and E outputs a vector of estimated $(\tilde{a}, \tilde{b})$ after applying a sigmoid activation scaled by 1×10^{-2} (to avoid possible divergence). For the baseline Neural ODE, we concatenate a and b to each sequence as two channels.

Table C.5. – Detailed ablation study on supervision and optimization for the reaction-diffusion equation, wave equation and damped pendulum.

Dataset	Method	log MSE	%Err Param.	$\|f_u\|^2$
Reaction-diffusion	Augmented Param. PDE (a, b) derivative supervision	-4.42±0.25	12.6	(6.8±0.6)e1
	Augmented Param. PDE (a, b) non-adaptive optim.	-4.55±0.11	7.5	(7.6±1.0)e1
	APHYNITY Param. PDE (a, b)	**-5.10±0.21**	**2.3**	(6.7±0.4)e1
	Augmented Param. PDE (a, b, k) derivative supervision	-4.90±0.06	11.7	(1.9±0.3)e-1
	Augmented Param. PDE (a, b, k) non-adaptive optim.	-9.10±0.02	0.21	(5.5±2.9)e-7
	APHYNITY Param. PDE (a, b, k)	**-9.35±0.02**	**0.096**	(1.5±0.4)e-6
	Augmented True PDE derivative supervision	-6.03±0.01	n/a	(3.1±0.8)e-3
	Augmented True PDE non-adaptive optim.	-9.01±0.01	n/a	(1.5±0.8)e-6
	APHYNITY True PDE	**-9.17±0.02**	n/a	(1.4±0.8)e-7
Wave equation	Augmented Param PDE (c) derivative supervision	-1.16±0.48	12.1	0.00024
	Augmented Param PDE (c) non-adaptive optim.	-2.57±0.21	3.1	43.6
	APHYNITY Param PDE (c)	**-4.64±0.25**	**0.31**	71.0
	Augmented Param PDE (c, k) derivative supervision	-4.19±0.36	7.2	0.00012
	Augmented Param PDE (c, k) non-adaptive optim.	-4.93±0.51	1.32	0.054
	APHYNITY Param PDE (c, k)	**-6.09±0.28**	**0.70**	4.54
	Augmented True PDE derivative supervision	-4.42 ± 0.33	n/a	6.02e-5
	Augmented True PDE non-adaptive optim.	-4.97±0.49	n/a	0.23
	APHYNITY True PDE	**-5.24±0.45**	n/a	0.14
Damped pendulum	Augmented Hamiltonian derivative supervision	-0.83±0.3	n/a	642±121
	Augmented Hamiltonian non-adaptive optim.	-0.49±0.58	n/a	165±30
	APHYNITY Hamiltonian	**-3.97±1.2**	n/a	623±68
	Augmented Param ODE (ω_0) derivative supervision	-1.02±0.04	5.8	136±13
	Augmented Param ODE (ω_0) non-adaptive optim.	-4.30±1.3	4.4	90.4±27
	APHYNITY Param ODE (ω_0)	**-7.86±0.6**	**4.0**	132±11
	Augmented Param ODE (ω_0, α) derivative supervision	-2.61±0.2	5.0	3.2±1.7
	Augmented Param ODE (ω_0, α) non-adaptive optim.	-7.69±1.3	1.65	4.8±7.7
	APHYNITY Param ODE (ω_0, α)	**-8.31±0.3**	**0.39**	8.5±2.0
	Augmented True ODE derivative supervision	-2.14±0.3	n/a	4.1±0.6
	Augmented True ODE non-adaptive optim.	**-8.34±0.4**	n/a	1.4±0.3
	APHYNITY True ODE	**-8.44±0.2**	n/a	2.3±0.4

In Table C.6, we observe that combining data-driven and physical components outperforms the pure data-driven one. When applying APHYNITY to Param PDE (a, b), the prediction precision is significantly improved (log MSE: -1.32 vs. -4.32) with a and b respectively reduced from 55.6% and 54.1% to 11.8% and 18.7%. For complete physics cases, the parameter estimations are also improved for Param PDE (a, b, k) by reducing over 60% of the error of b (3.10 vs. 1.23) and 10% to 20% of the errors of a and k (resp. 1.55/0.59 vs. 1.29/0.39).

The extensive results reflect the same conclusion as shown in the main article: APHYNITY improves the prediction precision and parameter estimation. The same decreasing tendency of $\|F_a\|$ is also confirmed.

Table C.6. – Results of the dataset of reaction-diffusion with varying (a, b). $k = 5 \times 10^{-3}$ is shared across the dataset.

	Method	log MSE	%Err a	%Err b	%Err k	$\|F_a\|^2$
Data-driven	Neural ODE (R. T. Q. Chen et al. 2018)	-3.61±0.07	n/a	n/a	n/a	n/a
Incomplete physics	Param PDE (a, b)	-1.32±0.02	55.6	54.1	n/a	n/a
	APHYNITY Param PDE (a, b)	**-4.32±0.32**	**11.8**	**18.7**	n/a	(4.3±0.6)e1
Complete physics	Param PDE (a, b, k)	**-5.54±0.38**	1.55	3.10	0.59	n/a
	APHYNITY Param PDE (a, b, k)	**-5.72±0.25**	**1.29**	**1.23**	**0.39**	(5.9±4.3)e-1
	True PDE	**-8.86±0.02**	n/a	n/a	n/a	n/a
	APHYNITY True PDE	**-8.82±0.15**	n/a	n/a	n/a	(1.8±0.6)e-5

C.7.2 Additional results for the wave equation

We conduct an experiment where each sequence is generated with a different wave celerity. This dataset is challenging because both c and the initial conditions vary across the sequences. For each simulated sequence, an initial condition is sampled as described previously, along with a wave celerity c also sampled uniformly in $[300, 400]$. Finally our initial state is integrated with the same Runge-Kutta scheme. 200 of such sequences are generated for training while 50 are kept for testing.

For this experiment, we also use a ConvNet encoder to estimate the wave speed c from 5 consecutive reserved states $(w, \frac{\partial w}{\partial t})$. The architecture of the encoder E is the same as in Table C.3 but with 10 input channels. Here also, k is fixed for all sequences and $k = 50$. The hyper-parameters used in these experiments are the same than described in the Section C.5.3.

The results when multiple wave speeds c are in the dataset are consistent with the one present when only one is considered. Indeed, while prediction performances are slightly hindered, the parameter estimation remains consistent for both c and k. This extension provides elements attesting for the robustness and adaptability of our method to more complex settings. Finally the purely data-driven Neural-ODE fails to cope with the increasing difficulty.

Table C.7. – Results for the damped wave equation when considering multiple c sampled uniformly in $[300, 400]$ in the dataset, k is shared across all sequences and $k = 50$.

	Method	log MSE	%Error c	%Error k	$\|f_u\|^2$
Data-driven	Neural ODE	0.056±0.34	n/a	n/a	n/a
Incomplete physics	Param PDE (c)	-1.32±0.27	23.9	n/a	n/a
	APHYNITY Param PDE (c)	**-4.51±0.38**	3.2	n/a	171
Complete physics	Param PDE (c, k)	-4.25±0.28	3.54	1.43	n/a
	APHYNITY Param PDE (c, k)	**-4.84±0.57**	2.41	0.064	3.64
	True PDE (c, k)	**-4.51±0.29**	n/a	n/a	n/a
	APHYNITY True PDE (c, k)	**-4.49±0.22**	n/a	n/a	0.0005

C.7.3 Damped pendulum with varying parameters

To extend the experiments conducted in the paper (section 8.4) with fixed parameters ($T_0 = 6, \alpha = 0.2$) and varying initial conditions, we evaluate APHYNITY on a much more challenging dataset where we vary both the parameters (T_0, α) and the initial conditions between trajectories.

We simulate 500/50/50 trajectories for the train/valid/test sets integrated with DOPRI5. For each trajectory, the period T_0 (resp. the damping coefficient α) are sampled uniformly in the range $[3, 10]$ (resp. $[0, 0.5]$).

We train models that take the first 20 steps as input and predict the next 20 steps. To account for the varying ODE parameters between sequences, we use an encoder that estimates the parameters based on the first 20 timesteps. In practice, we use a recurrent encoder composed of 1 layer of 128 GRU units. The output of the encoder is fed as additional input to the data-driven augmentation models and to an MLP with final softplus activations to estimate the physical parameters when necessary ($\omega_0 \in \mathbb{R}_+$ for Param ODE (ω_0), $(\omega_0, \alpha) \in \mathbb{R}_+^2$ for Param ODE (ω_0, α)).

In this varying ODE context, we also compare to the state-of-the-art univariate time series forecasting method N-Beats (Oreshkin et al. 2020).

Results shown in Table C.8 are consistent with those presented in the paper. Pure data-driven models Neural ODE (R. T. Q. Chen et al. 2018) and N-Beats (Oreshkin et al. 2020) fail to properly extrapolate the pendulum dynamics. Incomplete physical models (Hamiltonian and ParamODE (ω_0)) are even worse since they do not account for friction. Augmenting them with APHYNITY significantly and consistently improves forecasting results and parameter identification.

Table C.8. – Forecasting and identification results on the damped pendulum dataset with different parameters for each sequence. log MSEs are computed over 20 predicted time-steps. For each level of incorporated physical knowledge, equivalent best results according to a Student t-test are shown in bold. n/a corresponds to non-applicable cases.

	Method	log MSE	%Error T_0	%Error α	$\|f_u\|^2$
data-driven	Neural ODE (R. T. Q. Chen et al. 2018)	-4.35±0.9	n/a	n/a	n/a
	N-Beats (Oreshkin et al. 2020)	-4.57±0.5	n/a	n/a	n/a
Incomplete physics	Hamiltonian (Greydanus et al. 2019)	-1.31±0.4	n/a	n/a	n/a
	APHYNITY Hamiltonian	**-4.72±0.4**	n/a	n/a	5.6±0.6
	Param ODE (ω_0)	-2.66±0.9	21.5±19	n/a	n/a
	APHYNITY Param ODE (ω_0)	**-5.94±0.7**	**5.0±1.8**	n/a	0.49±0.1
Complete physics	Param ODE (ω_0, α)	**-5.71±0.4**	4.08±0.8	152±129	n/a
	APHYNITY Param ODE (ω_0, α)	**-6.22±0.7**	**3.26±0.6**	**62±27**	(5.39±0.1)e-10
	True ODE	**-8.58±0.1**	n/a	n/a	n/a
	APHYNITY True ODE	**-8.58±0.1**	n/a	n/a	(2.15±1.6)e-4

APPENDIX D

APPENDICES TO CONSTRAINED PHYSICAL-STATISTICS MODELS FOR DYNAMICAL SYSTEM IDENTIFICATION AND PREDICTION

D.1 Distance

D.1.1 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:

$$\mathrm{d}(u, v) = \mathbb{E}_{X\sim p_X} \left\| u(X) - v(X) \right\|_2 \tag{D.1}$$

Naturally, eq. (D.1) 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 $\lambda \in [0, 1]$:

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

Hence the convexity of $\mathrm{d}(., f)$. This consideration suffices to ensure the convexity of $\mathcal{S}_k$ and $\mathcal{S}_u$ defined in section 9.3.

D.1.2 Distance Between Flows

Consider the ODE with $X(t), X_0 \in \mathbb{R}^p$:

$$\frac{dX(t)}{dt} = f(X(t)), \quad X(t=0) = X_0 \tag{D.2}$$

Equation (D.2) admits a unique solution as soon as f is Lipschitz. We note $X^\star$ this solution. Then, we can defined the flow ϕ_f of such ODE as :

$$\begin{aligned} [0,T]\times\mathbb{R}^p &\to \mathbb{R}^p \\ t, \quad X_0 &\to \phi_f(t, X_0) = X^\star(t) \end{aligned} \tag{D.3}$$

With the definition of eq. (D.3), we can define the distance between two flows of ODE as:

$$d_\phi(\phi_u, \phi_f) = \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_f(t, X_0)\|\, dt \tag{D.4}$$

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

$$\begin{aligned} d_\phi(\phi_u, \phi_f) &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_f(t, X_0)\|\, dt \\ &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_v(t, X_0) + \phi_v(t, X_0) + \phi_f(t, X_0)\|\, dt \\ &\leq \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_v(t, X_0)\| + \|\phi_v(t, X_0) + \phi_f(t, X_0)\|\, dt \\ &\leq d_\phi(\phi_v, \phi_v) + d_\phi(\phi_v, \phi_f) \end{aligned}$$

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

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

However, in this case the convexity is not ensured with respect to u and vThis is the reason why for theoretical investigations, we consider the distance d instead of d_ϕ.

Nonetheless, $\mathrm{d}_\phi(\phi_u, \phi_f) = 0 \implies \phi_u = \phi_f \implies u = f$.

D.2 Remark on Additive Decomposition

First, note that in the case of a metric space the decomposition as defined in eq. (9.1) always exists.

We now detail an intuition for the well-posedness of such decomposition.

Let $\mathcal{H}_k$ be a closed convex subset of functions of an Hilbert space $(E, <, >)$, and f the function we want to approximate with partial knowledge (represented by the space of hypothesis $\mathcal{H}_k$). Then, thanks to Hilbert projection lemma, we have the uniqueness of the minimizer of $\min_{g \in \mathcal{H}_k} \|f - g\|$, i.e it exists one unique $h_k \in \mathcal{H}_k$ such that: $\forall g \in \mathcal{H}_k, \|f - h_k\| \leq \|f - g\|$.

Finally, the additive decomposition hypothesis presents a remarkable advantage in the case of a vector space. Indeed, if $\mathcal{H}_k$ is a (closed) vector space, let $\mathcal{H}_k^{\perp}$ be its supplementary in E, then we have the uniqueness in the decomposition: $f = f_{\mathcal{H}_k} + f_{\mathcal{H}_k^{\perp}}$, where $f_{\mathcal{H}_k^{\perp}} \in \mathcal{H}_k^{\perp}$ and $f_{\mathcal{H}_k} \in \mathcal{H}_k$.

The existence and uniqueness flowing directly from the additive decomposition hypothesis, this can explain why such assumption is common when bridging ML and MB hypothesis.

D.3 Upper Bounds

D.3.1 Derivation of Equation (9.3)

The first upper bound is a simple use of the triangle inequality:

$$\begin{aligned} \mathrm{d}(h, f) &= \mathrm{d}(h, f) + \mathrm{d}(h_k, f) - \mathrm{d}(h_k, f) \\ &\leq \mathrm{d}(h_k, f) + |\mathrm{d}(h, f) - \mathrm{d}(h_k, f)| \\ &\leq \mathrm{d}(h_k, f) + \mathrm{d}(h, h_k) \end{aligned}$$

D.3.2 Derivation of Equation (9.4)

To derive the second upper bound, we assume that f_k^{pr} comes from an overall dynamics f^{pr} obeying the additive decomposition hypothesis of eq. (9.1) so that f^{pr} and f_k^{pr} verifies: $f^{pr} = f_k^{pr} + f_u^{pr}$. First, with computations similar to eq. (9.3), we have:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, f^{pr}) + \mathrm{d}(f^{pr}, f) \tag{D.5}$$

Then:

$$\begin{aligned}
\mathrm{d}(h, f^{pr}) &= \mathrm{d}(h, f^{pr}) + \mathrm{d}(h_k, f_k^{pr}) - \mathrm{d}(h_k, f_k^{pr}) \\
&\leq \mathrm{d}(h_k, f_k^{pr}) + |\,\mathrm{d}(h, f^{pr}) - \mathrm{d}(h_k, f_k^{pr})| \\
&\leq \mathrm{d}(h_k, f_k^{pr}) + |\,\mathrm{d}(h, f^{pr}) - \mathrm{d}(h, f_k^{pr}) - \mathrm{d}(h, f_k^{pr}) + \mathrm{d}(h_k, f_k^{pr})| \\
&\leq \mathrm{d}(h_k, f_k^{pr}) + |\,\mathrm{d}(h, f^{pr}) - \mathrm{d}(h, f_k^{pr})| + |\,\mathrm{d}(h_k, f_k^{pr}) - \mathrm{d}(h, h_k)| \\
&\leq \mathrm{d}(h_k, f_k^{pr}) + \mathrm{d}(f^{pr}, f_k^{pr}) + \mathrm{d}(h, h_k) \qquad \text{(D.6)}
\end{aligned}$$

Combining Equations (D.5) and (D.6), we retrieve eq. (9.4):

$$\mathrm{d}(h, f) \leq \mathrm{d}(h_k, f_k^{pr}) + \mathrm{d}(h, h_k) + \mathrm{d}(f^{pr}, f_k^{pr}) + \mathrm{d}(f^{pr}, f) \qquad \text{(D.7)}$$

and we have: $\Gamma = \mathrm{d}(f^{pr}, f_k^{pr}) + \mathrm{d}(f^{pr}, f)$. Γ is a constant of the problem that cannot be optimized.

D.3.3 Upper-Bound Using Auxiliary Dynamics f^{pr}

Let f^{pr} be the dynamics of model data, we can link up the error made by h on true data (following dynamics f) and the error made by h on model data (with dynamics f^{pr}) via:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, f^{pr}) + \mathrm{d}(f^{pr}, f) \qquad \text{(D.8)}$$

Thus a pre-training on auxiliary data of dynamics f^{pr} amounts to control the term $\mathrm{d}(h, f^{pr})$ in the upper-bound of eq. (D.8).

D.3.4 Self-Supervision

Let $h = h_k + h_u$ be the function to learn and G_ψ the recognition network providing an estimate $\hat{\theta}_k^i$ of the parameters from an initial sequence $(X_{t_0}^i, \ldots, X_{t_0+k\Delta t}^i)$. This learning setting corresponds to how velocity fields are learned from consecutive measurements of the tracer fields T in section 9.4.2.

To compute $\mathrm{d}(h_k, f_k^{pr})$ in the case where $f^{pr} = h^\star$, where $h^\star = h_k^\star + h_u^\star$ is a learned model, we rely on the computed θ_k associated to $h_k^\star$ (thanks for example to the algorithm of section 9.3.2 associated to eq. (9.3)) to generate a synthetic dataset with achievable supervision in the space of the parameters θ_k.

From a real initial sequence $(X_{t_0}^i, \ldots, X_{t_0+k\Delta t}^i)$, we can estimate the unknown parameter θ_k^i associated to sequence i with the recognition network $G_\psi^\star$ learned with $h^\star$, i.e $\theta_k^i = G_\psi^\star(X_{t_0}^i, \ldots, X_{t_0+k\Delta t}^i)$. Then, integrating from the initial condition $X_{t_0}^i$, we generate a trajectory of known parameters θ_k^i with dynamics $h^\star$ denoted

by: $\tilde{X}^i = (\tilde{X}^i_{t_0}, \ldots, \tilde{X}^i_{t_n})$. Sampling the space of initial conditions, we obtain a synthetic dataset: $((\tilde{X}^1, \theta^1_k), \ldots, (\tilde{X}^m, \theta^m_k))$ enabling us to perform self-supervision for G_ψ. Let $\hat{\theta}^i_k$ be the parameters estimated by G_ψ from the simulated $(\tilde{X}^i_t, \ldots, \tilde{X}^i_{t+k\Delta t})$, we make the following approximation:

$$\mathrm{d}(h_k, f_k) \approx \frac{1}{m} \sum_{i=1}^{m} \left\| \hat{\theta}^i_k - \theta^i_k \right\|_2 \tag{D.9}$$

D.4 Proofs to Propositions

D.4.1 Note on the Convexity

Convexity of $\mathcal{S}_k$

Proof. Let $u, v \in \mathcal{S}_k$:

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

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

Convexity of $\mathcal{S}_u$

Proof. Let $t \in [0, 1]$ and $u, v \in \mathcal{S}_u$.

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

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

D.4.2 ODE Identification

Consider the following set: $\mathcal{S}_A = \{X(t) \in \mathcal{C}^1([0, T], \mathbb{R}^p)$ 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:

$$\mathcal{L}(A) = \sum_t \|(X^s(t+1) - (A\Delta t + Id)X^A(t))\|_2^2, \tag{D.10}$$

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.

D.4.3 Proof for Well-posedness of 9.6

We set ourselves in the Hilbert space of squared integrable functions with the canonical scalar product $(\mathcal{L}^2(\mathbb{R}^p, \mathbb{R}^p), <,>)$. For further consideration on such functional space we refer to (Droniou 2001).

We assume that $\mathcal{H}_k$ hence $\mathcal{S}_k$ is convex and a relatively compact family of functions.

Convexity of $\mathcal{S}_k + \mathcal{S}_u$ Let $\mathcal{S} = \mathcal{S}_k + \mathcal{S}_u = \{f \ | \exists f_k \in \mathcal{S}_k, f_u \in \mathcal{S}_u, f = f_k + f_u\}$.

Let $f, g \ \in \mathcal{S}$ and $\lambda \in]0,1[$:

$$\lambda f + (1-\lambda)g = \lambda f_k + (1-\lambda)g_k + \lambda f_u + (1-\lambda)g_u \in \mathcal{S}_k + \mathcal{S}_u$$

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

Closeness of $\mathcal{S}_u$ We show that $\mathcal{S}_u$ is a closed set. Indeed, $\mathcal{S}_u = g^{-1}([0, \mu_u])$, where $g(u) = \|u\|$, Because g is 1-Lipschitz (using the triangle inequality), g is continuous. Therefore $\mathcal{S}_u$ 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 $f_u^n \in \mathcal{S}_u$ and $f_k^n \in \mathcal{S}_k$.

Thanks to the relative compactness of $\mathcal{S}_k$, we can extract a converging sub-sequence, of indexes n_j, from f_k^n so that $f_k^{n_j} \to f_k \in \mathcal{S}_k$.

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 $\mathcal{S}_u$ 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 $\mathcal{S}_u$ is a closed set $f_u^{n_j}$ converges in $\mathcal{S}_u$. We write f_u its limit. Therefore, $f_u^{n_j} = f^{n_j} - f_k^{n_j} \to f - f_k = f_u \in \mathcal{S}_u$. Hence, $f = f_u + f_k$ with $f_u \in \mathcal{S}_u$ and $f_k \in \mathcal{S}_k$.

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. (9.6).

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 $A(x) = \{f(x) | f \in \mathcal{F}\}$.

D.4.4 Proof of Theorem 9.2

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

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. □

D.4.5 Algorithm in Linear Setting

We detail in Algorithm D.1 the alternate projection algorithm in a linear setting. We denote $Y = (X^i_{t_0+\Delta t}, X^i_{t_0+n\Delta t})$ and $X = (X^i_{t_0}, X^i_{t_0+(n-1).\Delta t})$. For readability purposes we set $\Delta t = 1$.

Algorithm D.1 Alternate estimation: Linear Setting

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

D.4.6 Proof to Theorem 9.3

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:

$$\mathcal{L}_D = \|Y - XA^n - D\|_2^2 + \lambda\|D\|_2^2 \tag{D.11}$$

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

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

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:

$$\mathcal{L}_A = \|Y - XA - D\|_2^2 + \gamma\|Y - XA\|_2^2 \tag{D.12}$$

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

$$\begin{aligned}\frac{\partial \mathcal{L}_A}{\partial a_j} = 0 \Leftrightarrow & -2 * \sum_{i=1}^{Q} x_{i,j}(y_i - a_0 x_{i,0} + \cdots + a_p x_{i,p} - d) \\ & -2\gamma * \sum_{i=1}^{Q} x_{i,j}(y_i - a_0 x_{i,0} + \cdots + a_p x_{i,p}) = 0 \\ \Leftrightarrow & -2X^t(Y - XA - D) - 2\gamma X^t(Y - XA) = 0 \\ \Leftrightarrow & (1+\gamma)X^tXA - X^t(Y - D) - \gamma X^tY = 0 \\ \Leftrightarrow & (1+\gamma)X^tXA = X^t(\gamma Y + (Y - D)) \\ \Leftrightarrow & A = \frac{B^{-1}X^t}{1+\gamma}((1+\gamma)Y - D) \end{aligned} \tag{D.13}$$

where $B = X^tX$. Equation (D.13) indicates that as soon a D converges, A^n converges. Thus, we now prove the convergence of (D^n). Then, for $n > 1$ consider:

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

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. □

D.5 Datasets

In this section, we provide exhaustive simulation details for the damped pendulum, Lotka-Volterra, and both geophysical datasets.

D.5.1 Damped-Pendulum

For the damped pendulum data, eq. (9.14) is integrated with $\Delta t = 0.2s$ using a Runge-Kutta 4-5 scheme from $t = 0$ up to $t = 10s$. Both the pulsation ω_0 and the damping coefficient k are fixed across the dataset. We generate 100/50/50 sequences respectively for train, validation and test sampling over the initial conditions so that $(\theta, \dot{\theta}) \sim \mathcal{U}([-\pi/2, \pi/2] \times [-0.1, 0.1])$.

Small Oscillations To linearize the pendulum, we consider the small oscillations regime and take the initial conditions so that : $(\theta, \dot{\theta}) \sim \mathcal{U}([-\pi/6, \pi/6] \times [-0.1, 0.1])$. In that case eq. (9.14) writes as:

$$\frac{d}{dt}\begin{pmatrix}\dot{\theta}\\ \theta\end{pmatrix} = \begin{pmatrix}-\lambda & \frac{g}{L}\\ 1 & 0\end{pmatrix}\begin{pmatrix}\dot{\theta}\\ \theta\end{pmatrix} \tag{D.14}$$

and following notations of section 9.3.3, we have: $D_A = 0$ and $A = \begin{pmatrix}-\lambda & \frac{g}{L}\\ 1 & 0\end{pmatrix}$

D.5.2 Lotka-Volterra

For Lotka-Volterra data, eq. (9.15) is integrated with $\Delta t = 0.05$ using a Runge-Kutta 4-5 scheme from $t = 0$ up to $t = 20$. All parameters $\alpha, \beta, \gamma, \delta$ are set to 1 across the dataset. We generate 100/50/50 sequences respectively for train, validation and test sampling over the initial prey and predators populations so that $(x, y) \sim \mathcal{U}([0, 2]^2)$.

Practical Issues and Adaptation Assuming that α and γ have positive values makes the following problem arises: the trajectories defined by h_k for the prey are unbounded, whereas the trajectories defined by eq. (9.15) are. Minimizing $\mathrm{d}(h_k, f)$ over long term horizon will lead in an underestimation of α to match the bounded behaviour of true data. Therefore, we enforce $\mathrm{d}(h_k, f)$ on the prey component as soon as the number of predator is small. In practice, we set this threshold to 0.15.

D.5.3 Geophysical Datasets

We present in this section introductory tools for the understanding of the fluid dynamics data presented in section 9.4.2. We first introduce the physical modeling

of ocean dynamics. Then, we outline the Adv+S dataset simulation which draws from ocean modeling. Finally, we introduce the Natl dataset and the proxy data used in the experiments.

Introduction To Ocean Modeling The increase in ocean observations thanks to satellites and floats enabled a great development in Earth modeling over the last decades. The ocean circulation, that is the current velocity fields dynamics, are now realistically modeled in tri-dimensional structured models such as NEMO (Madec 2008).

Such models rely on in-depth physical knowledge of the studied system and its representation through partial differential equations. Integrated over depth, the equations associated to the transport of the Sea Surface Temperature T by a time-varying horizontal velocity field U can be written as:

$$\frac{\partial T}{\partial t} = -\nabla.(TU) + D^T + F^T \tag{D.15}$$

$$\frac{\partial U}{\partial t} = -(U.\nabla)U + f \wedge U - g'\nabla h + D^U + F^U \tag{D.16}$$

where f 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 $g' \approx g.10^{-3}$. In a two-dimensional setting, $\nabla.(TU)$ refers to the advection of a scalar quantity T by a velocity field $U = (u, v)$ and writes as : $\nabla.(TU) = \frac{\partial T}{\partial x}u + \frac{\partial T}{\partial y}v$. The mixing terms, referred to as $D^{T/U}$ and the forcings $F^{T/U}$, are not known.

In the context of the presented work, the physical state $Z_t = (T_t, U_t)$, f_X and f_Y from eq. (7.1) can be interpreted as follows: f_X represents the dynamics of the observed T, i.e. $f_X(T) = -\nabla.(TU) + D^T + F^T$ in eq. (D.15). f_Y represents the dynamics of U in eq. (D.16), i.e. $f_Y(U, h) = -(U.\nabla)U + f \wedge U - g'\nabla h + D^U + F^U$.

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 hypothesis such as stationarity ($\frac{\partial U}{\partial t} = 0$), incompressibility ($(U.\nabla)U = 0$)), forcings can be omited. In this case, eq. (D.16) can be rewritten into

$$f \wedge U = -g'\nabla h \tag{D.17}$$

When projected onto x and y axis, eq. (D.17) becomes

$$-fv = -g'\frac{\partial h}{\partial x}, \qquad fu = -g'\frac{\partial h}{\partial y}, \tag{D.18}$$

Note that eq. (D.17) and eq. (D.18) do not hold at fine scales as the stationarity and incompressibility assumptions only hold at large scale. In this case, the SSH h can be regarded as a stream function.

Both datasets considered in the paper follow the same equations approximating the tracer equation (eq. (9.16)) inspired by eq. (D.15):

$$\frac{\partial T}{\partial t} = -\nabla.(TU) + S \tag{D.19}$$

We study the equations D.15 and D.16 in an incremental approach. In the following parts, we describe how T, U and S are computed in both datasets Adv+S and Natl.

Adv+S

We first investigate a dataset generated following simplifying assumptions (Adv+S). We don't rely on true U and S, we instead build them so that they correspond to our hypothesis.

Building a Velocity Field U Under stationarity and incompressibility hypothesis, 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):

$$\mathcal{H}(x, y, t) = -\tanh[\frac{y - B(t) \times \cos kx}{\sqrt{1 + k^2 B(t)^2 \times sin^2 kx}}] + cy, \tag{D.20}$$

As introduced precedently (see eq. (D.17)), eq. (D.16) can be simplified and we compute $U = (u, v)$ so that it follows:

$$u = -\frac{\partial \mathcal{H}}{\partial y}, \quad v = \frac{\partial \mathcal{H}}{\partial x} \tag{D.21}$$

Note that B varies periodically with time according to $B = B_0 + \epsilon \cos(\omega t + \phi)$. We compute 10 different velocity fields sampling random parameters $B_0, k, c, \omega, \epsilon, \phi$.

Building a Source Term S In eq. (D.15), the diffusion term D^T is omitted. We generate the source term S so that it represents the forcing term F^T in eq. (D.19). 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:

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

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

Dataset Generation Using computed U and S, we integrate eq. (D.19) with $\Delta t = 8640s$ over 30 days, using a Semi-Lagrangian scheme (see explanations below). We generate 800/100/200 sequences respectively for train, validation and test sampling over the initial conditions, which are images of size 64×64 sampled from Natl dataset. Finally, for integration, we impose East-West periodic conditions, implying that what comes out the left part re-enters at the right, and reciprocally. We also impose velocity to be null on both top and bottom parts of the image.

Semi-Lagrangian Integration Unlike Eulerian scheme, relying on time discretization of the derivative, the semi Lagrangian scheme relies on the constancy of the solution of a PDE along *a characteristic curve*. Consider a solution to the advection equation, i.e. eq. (D.19) with $S = 0$. The method of characteristics consists in exhibiting curves $(x(s), t(s))$ along which the derivative of the solution T is simple, i.e $\frac{\partial T}{\partial s}(x(s), t(s)) = 0$. For a 1D constant advection scheme, computations lead to:

$$\begin{aligned} \frac{dt}{ds} &= 1 \implies s = t \\ \frac{dx}{ds} &= U \implies x = x_0 + Ut \end{aligned}$$

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 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:

$$\begin{aligned} \frac{dt}{ds} &= 1 \implies s = t \\ \frac{dx}{ds} &= U(x,t) \end{aligned} \tag{D.22}$$

A great deal in the semi-Lagrangian literature involves solving correctly eq. (D.22). We use the conventional mid-point integration rule and the semi-Lagrangian is implemented using Pytorch function `gridsample`, following in (Jaderberg et al. 2015). Further developments can be found for example in (Diamantakis 2014).

Natl

This second dataset depicts the actual ocean circulation, i.e. we consider both eq. (D.15) and eq. (D.16). In this case, no assumptions are made on U and S represents both diffusion terms D^T and forcing terms F^T. We access daily data over a year of ocean surface temperature of the North Atlantic observations model resulting from (Ajayi et al. 2019) [1]. The dataset covers a 2300km $\times$ 2560km zone at 1.5km resolution, in the North Atlantic Ocean.

In this real-life dataset, sea surface height (SSH) partial derivative with respect to x and y serves as proxies to the (unobserved) velocity fields U. Indeed, recall that simplifying hypotheses led us to eq. (D.18).

We divide the Natl zone into 270 patches of size 64×64. For each region, we extract sea surface temperatures, velocity fields, source terms and height variables. We sample 200/20/50 sequences of 1 year, for respectively train, validation and test. In this case, $\Delta t = 86400s$ (1 day).

D.6 Training Details

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

Hyper-Parameters Interpretation From eq. (9.3), two independent terms appear justifying an alternate projections approach.

First, we highlight that strictly minimizing $\mathrm{d}(h_k, f)$ biases our estimation of h_k. However, it may yield a good estimation of h_k provided that f_k contributes significantly to the prediction of f. Hence, we interpret this loss as an *initialization* loss. Thus, in most applications, we progressively decrease its magnitude along training as detailed in appendices D.6.1 to D.6.3.

On the other hand, $\mathrm{d}(h_u, 0)$ aims at constraining the free form function h_u to make its action as small as possible. We interpret this loss as a *stability* penalty.

1. Details available at : https://meom-group.github.io/swot-natl60/access-data.html

Finally, aiming to recover exact trajectories of observations, we proceed as suggested in (Yin et al. 2021b) progressively increasing the hyper-parameters associated to $\mathrm{d}(h, f)$.

The practical implementation is summarized in the following algorithm:

Algorithm D.2 Alternate estimation: Practical Setting

Initialization: $\theta_u^0 = 0,\ \theta_k^0 = \min_{h_k \in \mathcal{H}_k} \mathrm{d}(h_k, f)$, λ_h, λ_{h_k}, λ_{h_u}
for $epoch = 1 : N_{epochs}$ **do**
  for $batch = 1 : B_k$ **do**

$$\theta_k^{n+1} = \theta_k^n - \tau_1 \nabla_{\theta_k} [\lambda_h \, \mathrm{d}(h, f) + \lambda_{h_k} \ell(h_k)]$$

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

$$\theta_u^{n+1} = \theta_u^n - \tau_1 \nabla_{\theta_u} [\lambda_h \, \mathrm{d}(h, f) + \lambda_{h_u} \, \mathrm{d}(h_u, 0)]$$

  end
  $\lambda_h = \tau_2 \lambda_h$; $\lambda_{h_k} = \frac{1}{\tau_2} \lambda_{h_k}$; $\lambda_{h_u} = \frac{1}{\tau_2} \lambda_{h_u}$
end

D.6.1 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 $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$. We select the ones with the lowest prediction errors, i.e with lowest $\mathrm{d}_\phi(h, f)$.

For the ablation study of Table 9.1, we set to 0 the hyper-parameters associated to the non-considered loss.

The training time for this dataset is 1 hour.

D.6.2 Lotka-Volterra

Architecture Details The physical parameters to be learned is a vector of dimension 2 accounting for (α, β) in eq. (9.15), 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 $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$. We select the ones with the lowest prediction errors (i.e lowest $\mathrm{d}(h, f)$).

For the ablation study of Table 9.1, we set to 0 the hyper-parameters associated to the non-considered loss.

The training time for this dataset is 2 hours.

D.6.3 Adv+S

Architectures Details The physical parameters to be estimated are the velocity fields U, of dimension $(2, 64, 64)$. As U varies over time, we follow data assimilation principles to map a sequence of 4 consecutive measurements of the tracer field T to the associated velocity field (Gaultier et al. 2013). To do so, we parameterize a recognition network G_ψ 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: $(T_{t_0}, .., T_{t_0+3\Delta t})$. The residual dynamics h_u is learned by a convolutional ResNet, with 1 residual block taking as entry the same sequence of T. We implement h_k via a semi-lagrangian scheme, taking as input T_t and the estimated U_t to predict T_{t+1}.

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 d_ϕ on $(T_{t_0+\Delta t}, ..., T_{t_0+6\Delta t})$. Overall the number of sequences for training is 36800. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 4-batches on h_k then 6-batches on h_u.

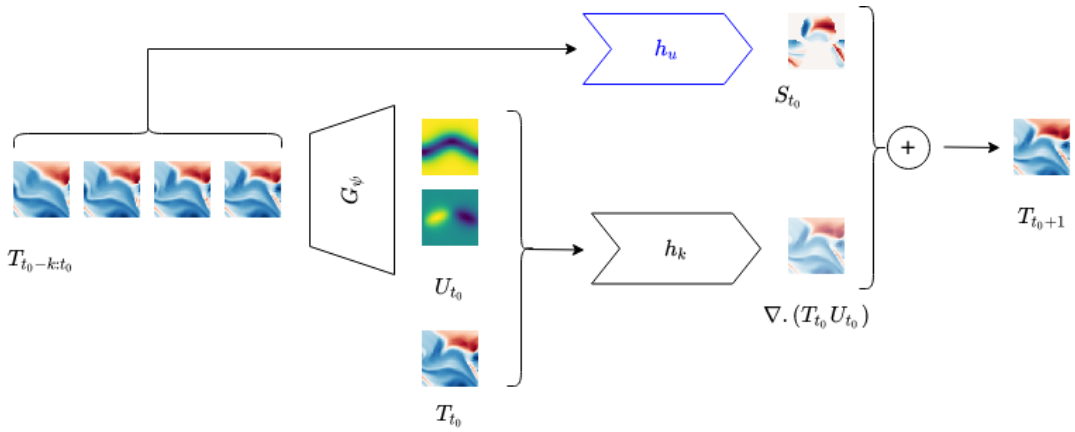


Figure D.1. – *Best viewed in color*. Schematic view of our model in the context of section 5.2, applied on the Adv+S dataset.

Hyperparameters, setting of eq. (9.3) We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.00001$. We initialize $\lambda_h = 0.01$ and increase it geometrically every epoch up to $\lambda_{h,f} = 1000$. λ_{h_u} is fixed through training at 0.1. We select the hyperparameters with the lowest prediction errors (i.e lowest $\mathrm{d}(h, f)$). For the ablation study of Table 9.1, we set to 0 the hyper-parameters associated to the non-considered loss.

The training time for this dataset is 8 hours.

D.6.4 Natl

Architecture Details The architectures in this setting are identical to the ones described in appendix D.6.3.

Optimization We use Adam optimizer with learning rate 0.00001 for 50 epochs with batch size 32. Overall the number of sequences for training is 67000. We enforce d_ϕ over 6 time-steps, i.e we supervise the predictions on timesteps: $(t_0 + \Delta t, .., t_0 + 6\Delta t)$. We use dropout in both G_ψ and h_u.

Hyperparameters, setting of eq. (9.3) For this setting, λ_h geometrically increases from 0.01 up to 100. We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.00001$. λ_{h_u} is fixed through training at 0.1. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 10-batches on both h_k and h_u.

The selected model is the one with lowest prediction errors on validation set (i.e lowest $\mathrm{d}(h, f)$), sampling uniformly the hyperparameters: $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$.

Hyperparameters, setting of eq. (9.4) Because the dynamics of Natl is highly non linear and chaotic, we follow (X. Jia et al. 2019) and first warm-up the parameters recognition network G_ψ on the velocity fields proxies for 10 epochs. For this setting, λ_h geometrically increase from 0.01 up to 1. λ_{h_k} is set equal to λ_h. λ_{h_u} is fixed through training at 0.01.

After warm-up, we alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 100-batches on h_k and 300 on h_u. In this setting of eq. (9.4), the selected model is the one with lowest $\mathrm{d}(h, f) + \mathrm{d}(h_k, f_k^{pr})$ error, sampling uniformly the hyperparameters: $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$.

The training time for this dataset is 12 hours.

Baselines We train NODE (R. T. Q. Chen et al. 2018) and Aphynity (Yin et al. 2021b) on both the Adv+S and Natl dataset. For the training of Aphinity, we set the learning rate at 0.0001 and train on 30 epochs. We initialize $\lambda_h = 0.01$ and increase it geometrically every epoch up to $\lambda_h = 100$. λ_{h_u} is fixed through training at 0.1. For the training of NODE, 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 network h. h is a 3-layer convolutional networks, with 64 hidden channels. It is integrated using RK4 scheme available from `https://github.com/rtqichen/torchdiffeq`.

D.7 Additional Results and Samples

D.7.1 Results for Pendulum and Lotka-Volterra Datasets

We provide respectively in figs. D.2 and D.3 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. D.3), and a converging spiral for the damped pendulum (fig. D.2).

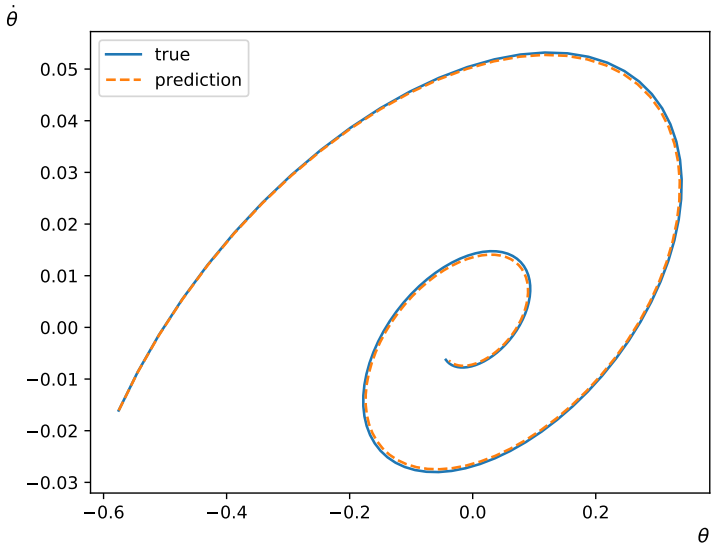


Figure D.2. – Damped Pendulum Phase Diagram. The true phase diagram (blue) and learned (orange dashed) are close, indicating consistency in the prediction

D.7.2 Results for Adv+S and Natl

In this section, we provide additionial results on both Adv+S and Natl datasets. A thorough ablation study (table D.1) gives results with constant hyperparameters λ_h and λ_{h_k} (row Vanilla Optim), which validates our hyper-parameters interpretation. Indeed, the results are better when respectively increasing and decreasing λ_h and λ_{h_k}. Besides, the row Ours eq. (9.4) refers to a training performed as introduced in appendix D.3.4 with $f^{pr} = h^\star$ trained on eq. (9.3). Figure D.5 shows predictions up to 4 days on the Adv+S data. Finally, figs. D.7 and D.9 provide results on Natl dataset associated to training relying on both eq. (9.3) and eq. (9.4) and with NODE (R. T. Q. Chen et al. 2018).

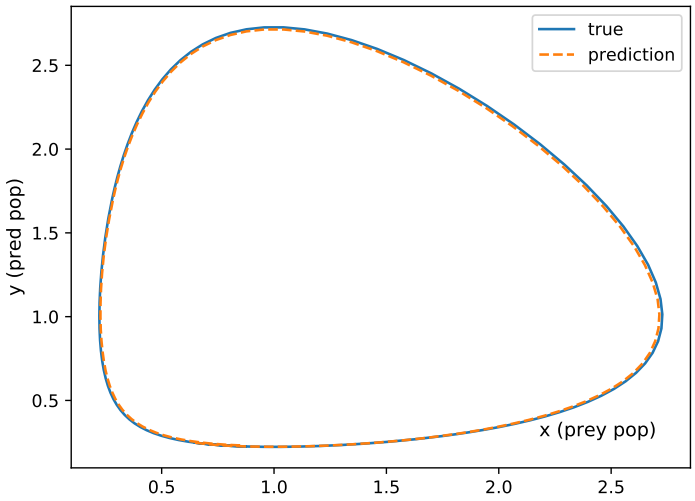


Figure D.3. – Lotka-Volterra Phase Diagram. The true phase diagram (blue) and learned (orange dashed) are close, indicating consistency in the prediction

Table D.1. – Ablation Study on Adv+S. We report the MSE ($\times$ 100) on the predicted observations T, the estimated velocity fields U and the residual source term S over 6 and 20 time steps from an initial datum t_0. Unlike alternate training, i.e. Algorithm 9.1, "Joint" rows refer to the simultaneous optimization of h_k and h_u.

Training	Models	$t_0 + 6$			$t_0 + 20$		
		T	U	S	T	U	S
	Ours (U known)	0.52	n/a	0.19	2.0	n/a	0.32
Alternate	Ours eq. (9.3)	**0.74**	**1.99**	0.17	8.49	**2.26**	0.31
	only $\mathrm{d}(h, f)$	1.02	4.08	0.19	10.59	4.19	0.32
	$\mathrm{d}(h, f) + \mathrm{d}(h_k, f)$	1.02	3.66	0.19	11.42	3.84	0.34
	$\mathrm{d}(h, f) + \mathrm{d}(h, h_k)$	0.77	2.38	0.19	9.5	2.45	0.34
	Ours eq. (9.4)	0.75	2.77	**0.17**	**8.36**	2.84	**0.29**
	Vanilla optim.	1.51	3.77	0.3	13.33	4.1	5.15
Joint	Ours eq. (9.3)	1.44	3.3	0.3	12.82	3.5	0.5
	only $\mathrm{d}(h, f)$	1.38	6.96	0.39	11.9	7.09	0.54

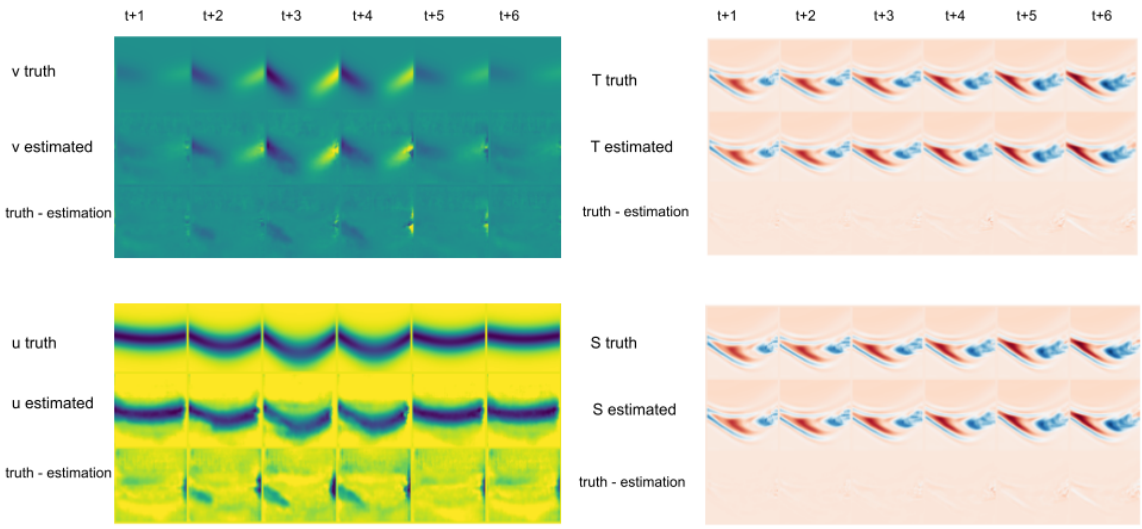


Figure D.4. – *Best viewed in color*. Estimations, targets and differences between estimations and targets on T, $U = (u, v)$ and S for Adv+S. Each column refers to a time step, ranging from 1 to 6 half-days. On the left, true and estimated $U = (u, v)$ over 6 time steps, and differences between targets and estimations. On the right, prediction of T and S over 6 time steps, and differences between targets and estimations.

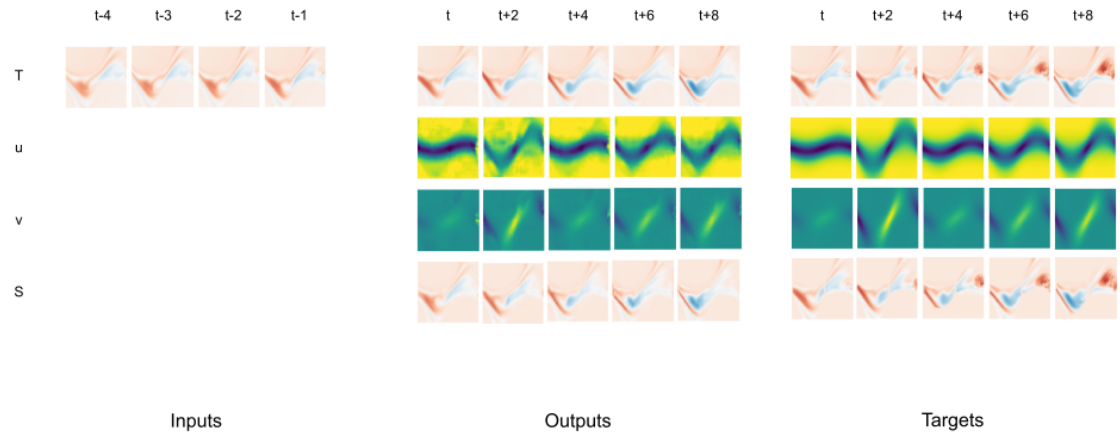


Figure D.5. – *Best viewed in color*. Estimations and targets on T, $U = (u, v)$ and S for Adv+S. Each column refers to a time step, ranging from 1 to 8 half-days. On the left, sequence of T inputs (4 time steps). In the middle, prediction of T, $U = (u, v)$ and S over 8 time steps. On the right, true T, U and S over 8 time steps.

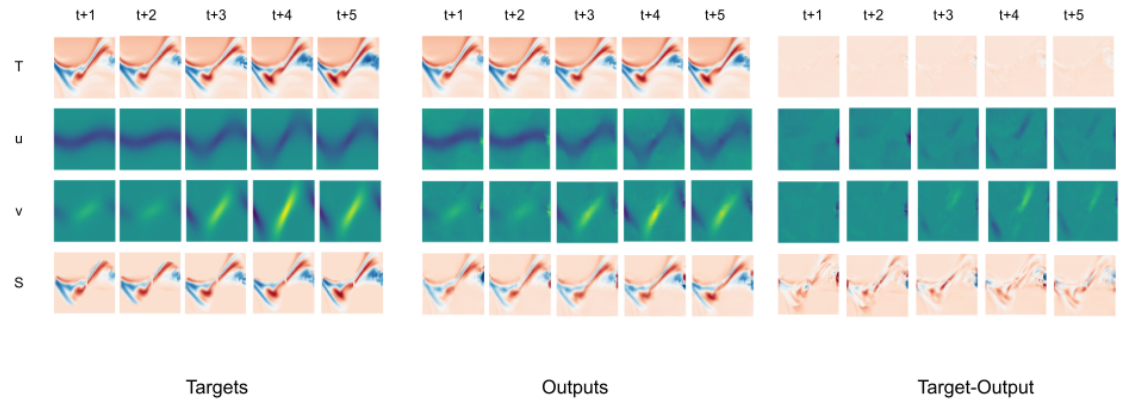


Figure D.6. – *Best viewed in color*. Estimations, targets and differences between estimations and targets on T, $U = (u, v)$ and S for Adv+S. Each column refers to a time step, ranging from 1 to 5 half-days. On the left, true T, U and S over 5 time steps.. In the middle, prediction of T, $U = (u, v)$ and S over 8 time steps. On the right, differences between targets and estimations.

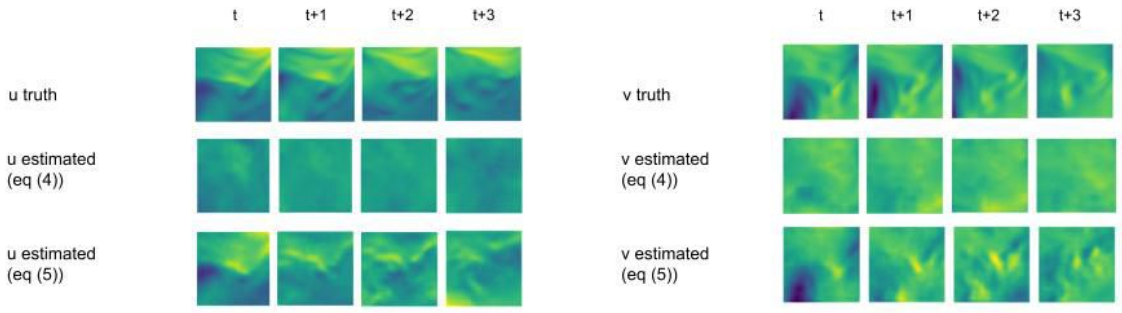


Figure D.7. – *Best viewed in color*. Sequence of estimations on $U = (u, v)$ for the Natl data. The second and third row respectively refer to training according to eq. (9.3) and eq. (9.4). The loss term $\mathrm{d}(h_k, f_k^{pr})$ in eq. (9.4) enables our model to learn more accurate velocity fields than when only trained following eq. (9.3).

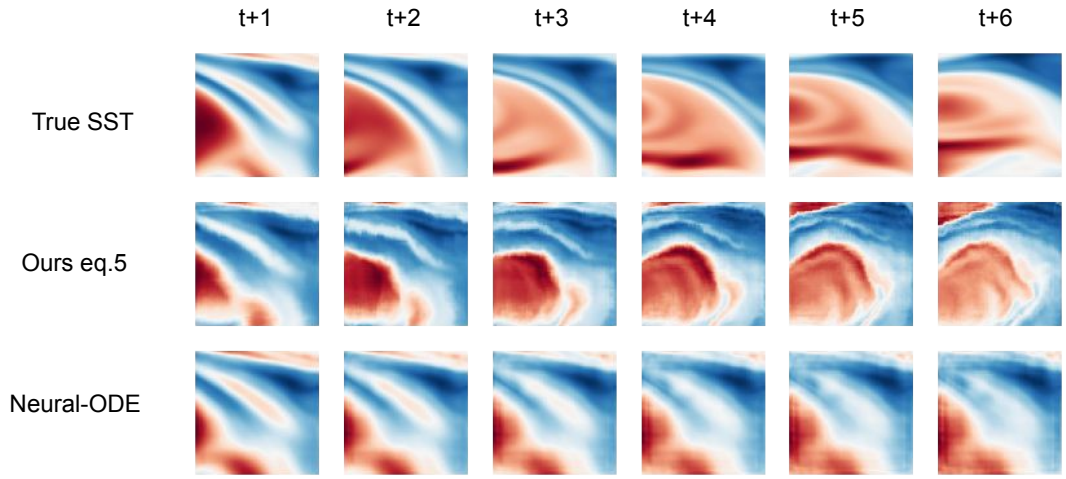


Figure D.8. – *Best viewed in color*. Sequence of prediction on T for the Natl data. Contrary to our model (row eq. (9.4)), NODE (row Neural-ODE) struggles to predict any motion in T.

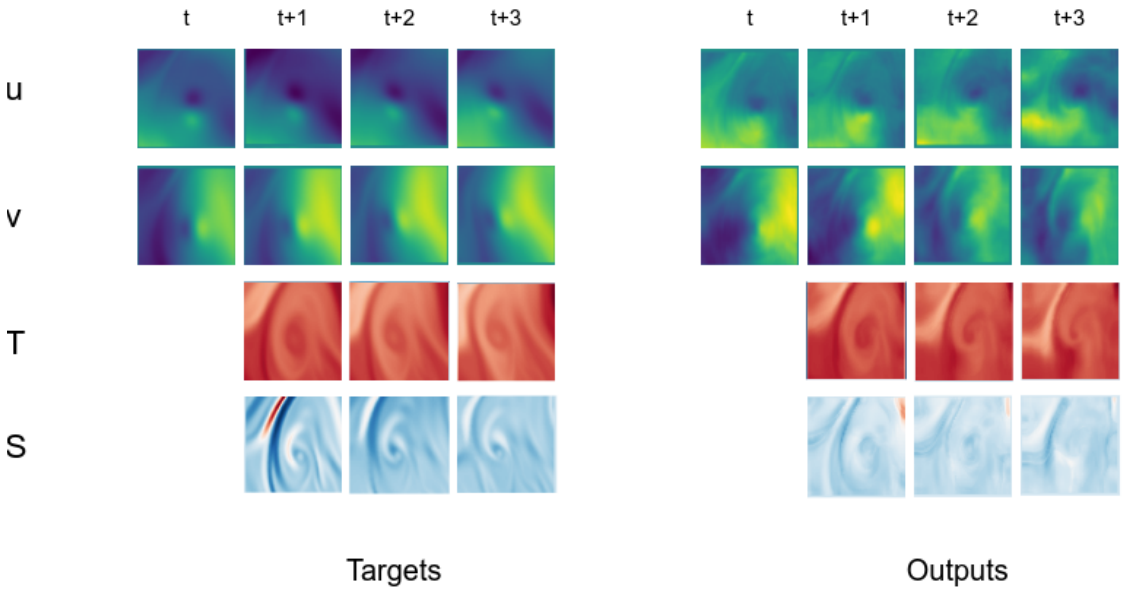


Figure D.9. – *Best viewed in color*. Sequence of prediction on T, u, v, S for the Natl data across 3 days trained using proxy data according to eq. (9.4)

APPENDIX E

APPENDICES TO CODA

E.1 Discussion

We discuss in more details the originality and differences of CoDA w.r.t. several Multi-Task Learning (MTL) and gradient-based or contextual meta-learning methods illustrated in Figure E.1. We consider CAVIA (Zintgraf et al. 2019), MAML (Finn et al. 2017), ANIL (Raghu et al. 2020), hard-parameter sharing MTL (Caruana 1997; Ruder 2017), LEADS (Yin et al. 2021a).

E.1.1 Adaptation Rule

We compare the adaptation rule in Equation (10.4) w.r.t. these work.

GBML Given k gradient steps, MAML defines

$$\theta^e = \theta^c + (-\eta \sum_{i=0}^{k} \nabla_\theta \mathcal{L}(\theta_i^e, \mathcal{D}^e)) \tag{E.1}$$

$$\text{where } \begin{cases} \theta_{i+1}^e = \theta_i^e - \eta \nabla_\theta \mathcal{L}(\theta_i^e, \mathcal{D}^e) & i > 0 \\ \theta_0^e = \theta^c & i = 0 \end{cases}$$

With $\delta\theta^e \triangleq -\eta \sum_{i=0}^{k} \nabla_\theta \mathcal{L}(\theta_i^e, \mathcal{D}^e)$, Equation (10.4) thus includes MAML. ANIL and related GBML methods (Kwonjoon Lee et al. 2019; Bertinetto et al. 2019) restrict Equation (E.1) to parameters of the final layer, while remaining parameters are shared.

MTL MTL models can be identified to Equation (E.1). They fix $\theta^c \triangleq \mathbf{0}$, removing the ability of performing fast adaptation as parameters are retrained from scratch instead of being initialized to θ^c. Hard-parameter sharing MTL restricts the sum in Equation (E.1) to the final layer, as ANIL. LEADS sums the outputs of a shared

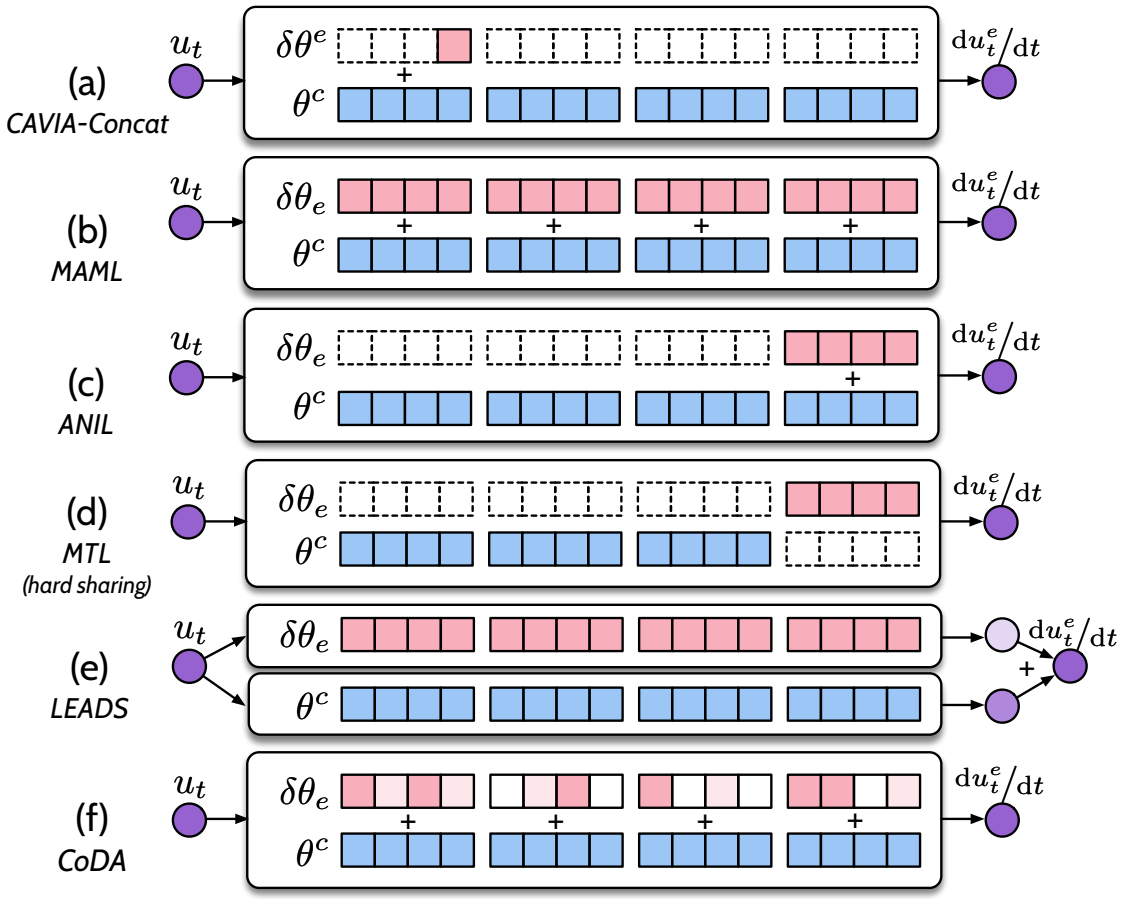


Figure E.1. – Illustration of representative baselines for multi-environment learning. Shared parameters are blue, environment-specific parameters are red. (a) CAVIA-Concat acts upon the bias of the first layer with conditioning via concatenation. (b) MAML acts upon all parameters without penalization nor prior structure information. (c) ANIL restricts meta-learning to the final layer. (d)Hard-sharing MTL train the final layer from scratch, while the remaining is a *hard-shared*. (e) LEADS sums the output of a common and a environment-specific network. (f) CoDA acts upon a subspace of the parameter space with a locality constraint.

and an environment specific network, thus splits parameters into two independent blocks that do not share connections.

E.1.2 Decoding for Context-Informed Adaptation

We show that conditioning strategies in contextual meta-learning for decoding context vectors ξ^e into $\delta\theta^e$ are a special case of hypernetwork-decoding. The two

main approaches are conditioning via concatenation and conditioning via feature modulation a.k.a. FiLM (Perez et al. 2018).

Conditioning via Concatenation

We show that conditioning via concatenation is equivalent to a linear hypernetwork $A_\phi : \xi^e \mapsto W\xi^e + \theta^c$ with $\phi = \{\theta^c, W\}$ that only predicts the bias of the first layer of g_θ.

We assume that g_θ has N layers and analyze the output of the first layer of g_θ, omitting the nonlinearity, when the input $x \in \mathbb{R}^{d_x}$ in an environment $e \in \mathcal{E}$ is concatenated to a context vector $\xi^e \in \mathbb{R}^{d_\xi}$. We denote $x \| \xi^e$ the concatenated vector, n_h the number of hidden units of the first layer, $W^1 \in \mathbb{R}^{n_h \times (d_x + d_\xi)}$ and $b^1 \in \mathbb{R}^{n_h}$ the weight matrix and bias term of the first layer, $W^2, \cdots, W^N$ and $b^2, \cdots, b^N$ those of the following layers. The output of the first layer is

$$y^1 = W^1 \cdot x \| \xi^e + b^1$$

We split W^1 along rows into two weight matrices, $W^1_x \in \mathbb{R}^{n_h \times d_x}$ and $W^1_\xi \in \mathbb{R}^{n_h \times d_\xi}$ s.t.

$$y^1 = W^1_x \cdot x + W^1_\xi \cdot \xi^e + b^1$$

$b^1_\xi \triangleq W^1_\xi \cdot \xi^e + b^1$ does not depend on x and corresponds to an environment-specific bias. Thus, concatenation is included in Equation (10.4) when

$$\theta^c \triangleq \{W^1_x, b^1, W^2, b^2, \cdots, W^N, b^N\}$$
$$\delta\theta^e \triangleq \{\ 0\ , b^1_\xi,\ 0\ , 0\ , \cdots,\ 0\ , 0\}$$

where $\delta\theta^e$ is decoded via a hypernetwork with parameters $\{\theta^c, W \triangleq (0, W^1_\xi, 0, \cdots, 0)\}$.

Conditioning via Feature Modulation

We show that conditioning via FiLM is equivalent to a linear hypernetwork $A_\phi : \xi^e \mapsto W\xi^e + \theta^c$ with $\phi = \{\theta^c, W\}$ that only predicts the batch norm (BN) statistics of g_θ.

For simplicity, we focus on a single BN layer and denote $\{h_i\}_{i=1}^M$, M feature maps output by preceding convolutional layers. These feature maps are first normalized then rescaled with an affine transformation. Rescaling is similar to a FiLM layer that transforms linearly $\{h_i\}_{i=1}^M$ with:

$$\forall i \in \{1, \cdots, M\}, \text{FiLM}(h_i) = \gamma_i \odot h_i + \beta$$

where $\gamma, \beta \in \mathbb{R}^M$ are output by a NN f_ψ conditioned on the context vectors ξ^e i.e. $[\gamma, \beta] = f_\psi(\xi^e)$. In general, f_ψ is linear s.t. $f_\psi(\xi^e) \triangleq W_\xi \xi^e + b_\xi$, with $\psi = \{W_\xi, b_\xi\}$. Then $\gamma = W_\xi^\gamma \xi^e + b_\xi^\gamma, \beta = W_\xi^\beta \xi^e + b_\xi^\beta$.

Thus, for this layer, modulation is included in Equation (10.4) when

$$\begin{aligned} \delta\theta^e &\triangleq W\xi^e = \{W_\xi^\gamma \xi^e, W_\xi^\beta \xi^e\} \\ \theta^c &\triangleq b_\xi = \{b_\xi^\gamma, b_\xi^\beta\} \end{aligned}$$

where $\delta\theta^e$ is decoded via hypernetwork $f_\psi \triangleq A_\phi$ with parameters $\phi = \{\theta^c \triangleq b_\xi, W \triangleq W_\xi\}$.

E.2 Proofs

Proposition E.1. *10.3 Given a class of linearly parametrized dynamics $\mathcal{F}$ with d_p varying parameters, $\forall \theta^c \in \mathbb{R}^{d_\theta}$, subspace $\mathcal{G}_{\theta^c}$ in Theorem 10.2 is low-dimensional and satisfies* $\dim(\mathcal{G}_{\theta^c}) \leq d_p \ll d_\theta$.

Proof. We define the linear mapping $\psi : p \in \mathbb{R}^{d_p} \to f \in \mathcal{F}$ from parameters to dynamics s.t. $\psi(\mathbb{R}^{d_p}) = \mathcal{F}$. Given this linear mapping, we first prove the following lemma: $\dim(\mathcal{F}) \leq d_p$. The proof is based on surjectivity of ψ onto $\mathcal{F}$, given by definition. We define $\{b_i\}_{i=1}^{d_p}$ a basis of $\mathbb{R}^{d_p}$. Given $f \in \mathcal{F}$, $\exists p \in \mathbb{R}^{d_p}, \psi(p) = f$. We note $p = \sum_{i=1}^{d_p} \lambda_i b_i$ where $\forall i, \lambda_i \in \mathbb{R}$. Then $\psi(p) = \sum_{i=1}^{d_p} \lambda_i \psi(b_i)$. We extract a basis from $\{\psi(b_i)\}_{i=1}^{d_p}$ and denote $d_f \leq d_p$ the number of elements in this basis. This basis forms a basis of $\mathcal{F}$ i.e. $d_f = \dim(\mathcal{F}) \leq d_p$.

Now, given $f^e \in \mathcal{F}$ and $\theta \in \mathbb{R}^{d_\theta}$,

$$\mathcal{L}(\theta, \mathcal{D}^e) \triangleq \mathbb{E}_{x \in \mathcal{D}^e} \|(f^e - g_\theta)(x)\|_2^2 = \|f^e - g_\theta\|_2^2$$

The gradient of $\mathcal{L}(\theta, \mathcal{D}^e)$ is then

$$\nabla_\theta \mathcal{L}(\theta, \mathcal{D}^e) = -2 \left(\frac{\mathrm{d}g_\theta}{\mathrm{d}\theta}\right)^\top (f^e - g_\theta)$$

where the adjoint of a linear map h is denoted $h^\top$. $f \mapsto -2 \left(\frac{\mathrm{d}g_\theta}{\mathrm{d}\theta}\right)^\top f$ is a linear map as $\theta \mapsto \frac{\mathrm{d}g_\theta}{\mathrm{d}\theta}$ is linear (differential of g_θ) and the adjoint preserves linearity, s.t. $\dim \mathrm{Span}(\{\nabla_\theta \mathcal{L}(\theta, \mathcal{D}^e)\}_{e \in \mathcal{E}}) \leq \dim(\mathcal{F}) \leq d_p$. □

Proposition E.2. *10.1 Given $\{\theta^c, W\}$ fixed, if $\|\cdot\| = \ell_2$, then Equation (10.8) is quadratic. If $\lambda' W^\top W$ or $\bar{H}^e(\theta^c) = W^\top \nabla_\theta^2 \mathcal{L}(\theta^c, \mathcal{D}^e) W$ are invertible then $\bar{H}^e(\theta^c) + \lambda' W^\top W$ is*

invertible except for a finite number of λ' values. The problem in Equation (10.8) is then also convex and admits an unique solution, $\{\xi^{e\star}\}_{e\in\mathcal{E}_{ad}}$. With $\lambda' \triangleq 2\lambda$,

$$\xi^{e*} = -\Big(\bar{H}^e(\theta^c) + \lambda' W^\top W\Big)^{-1} W^\top \nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e)$$

$\bar{H}^e(\theta^c) + \lambda' W^\top W$ is invertible $\forall \lambda'$ except a finite number of values if $\bar{H}^e(\theta^c)$ or $\lambda' W^\top W$ is invertible.

Proof. When $\|\cdot\| = \ell_2$, we consider the following second order Taylor expansion of $\mathcal{L}_{\text{reg}}(\theta, \mathcal{D}^e) = \mathcal{L}(\theta, \mathcal{D}^e) + \lambda\|\theta - \theta^c\|_2^2$ at θ^c, where $\delta\theta^e = \theta - \theta^c = W\xi^e$.

$$\mathcal{L}_{\text{reg}}(\theta^c + \delta\theta^e, \mathcal{D}^e) = \mathcal{L}(\theta^c, \mathcal{D}^e) + \nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e)^\top \delta\theta^e + \frac{1}{2}\delta\theta^{e\top}\Big(\nabla_\theta^2 \mathcal{L}(\theta^c, \mathcal{D}^e) + 2\lambda \mathrm{Id}\Big)\delta\theta^e + o(\|\delta\theta^e\|_2^3) \quad \text{(E.2)}$$

With $\delta\theta^e = W\xi^e$, we expand Equation (E.2) into

$$\mathcal{L}_{\text{reg}}(\theta^c + W\xi^e, \mathcal{D}^e) = \mathcal{L}(\theta^c, \mathcal{D}^e) + \Big(W^\top \nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e)\Big)^\top \xi^e + \frac{1}{2}\xi^{e\top}\Big(W^\top \nabla_\theta^2 \mathcal{L}(\theta^c, \mathcal{D}^e) W + 2\lambda W^\top W\Big)\xi^e + o(\|\delta\theta^e\|_2^3)$$

i.e. with $\bar{H}^e(\theta^c) = W^\top \nabla_\theta^2 \mathcal{L}(\theta^c, \mathcal{D}^e) W$ and $\lambda' = 2\lambda$

$$\mathcal{L}_{\text{reg}}(\theta^c + W\xi^e, \mathcal{D}^e) = \mathcal{L}(\theta^c, \mathcal{D}^e) + \Big(W^\top \nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e)\Big)^\top \xi^e + \frac{1}{2}\xi^{e\top}\Big(\bar{H}^e(\theta^c) + \lambda' W^\top W\Big)\xi^e + o(\|\delta\theta^e\|_2^3) \quad \text{(E.3)}$$

Equation (E.3) is quadratic. If $\bar{H}^e(\theta^c) + \lambda' W^\top W$ is invertible, then the problem is also convex with unique solution

$$\xi^{e*} = -\Big(\bar{H}^e(\theta^c) + \lambda' W^\top W\Big)^{-1} W^\top \nabla_\theta \mathcal{L}(\theta^c, \mathcal{D}^e)$$

$\bar{H}^e(\theta^c)$ and $\lambda' W^\top W$ are two square matrices. The application $p : \lambda' \mapsto \det(\bar{H}^e(\theta^c) + \lambda' W^\top W)$ is well-defined and forms a continuous polynomial. Thus either it equals zero or it has a finite number of roots. If $\bar{H}^e(\theta^c)$ or $\lambda' W^\top W$ is invertible, then $p(0) = \det(\bar{H}^e(\theta^c)) \neq 0$ or $p(\infty) \sim \det(\lambda' W^\top W) \neq 0$. Thus $p \neq 0$ has a finite number of roots i.e. $\bar{H}^e(\theta^c) + \lambda' W^\top W$ is invertible $\forall \lambda'$ except a finite number of values corresponding to the roots of p. □

E.3 System Parameter Estimation

We show in Theorem 10.5 that parameters of new systems can be recovered under Theorem 10.4.

Proposition E.3. *10.5 Under Theorem 10.4, system parameters are perfectly identified on new environments if the model g and hypernetwork A satisfy $\forall f_i \in \mathcal{B}, g_{A(p_i)} = f_i$.*

Proof. We define the linear mapping $\psi : p \in \mathbb{R}^{d_p} \to f \in \mathcal{F}$ from parameters to dynamics s.t. $\psi(\mathbb{R}^{d_p}) = \mathcal{F}$ (Theorem 10.4 (a)). Unicity of parameters (Theorem 10.4 (c)) implies that ψ is bijective with inverse ψ^{-1}, thus $\dim(\mathcal{F}) = \dim(\mathbb{R}^{d_p}) = d_p$. Given a basis $\mathcal{B} = \{f_i\}_{i=1}^{d_p}$ of $\mathcal{F}$, we denote $p_i = \psi^{-1}(f_i)$. We fix g, A s.t. $\forall i \in \{1, ..., d_p\}, g_{A(p_i)} = f_i = \psi(p_i)$. This is possible as f_i and g are linear w.r.t. inputs (Theorem 10.4 (a) and (b)) and p_i are known (Theorem 10.4 (e)).

$\forall i \in \{1, ..., d_p\}, f_i \in \mathrm{Im}(g_{A(\cdot)})$, thus $\mathcal{F} \subset \mathrm{Im}(g_{A(\cdot)})$ i.e. $d_p \leq \dim(\mathrm{Im}(g_{A(\cdot)}))$. g, A are linear (Theorem 10.4 (b)), thus $g_{A(\cdot)}$ is linear with inputs in $\mathbb{R}^{d_\xi}$. Then, $\dim(\mathrm{Im}(g_{A(\cdot)})) \leq d_\xi$ and $d_p \leq \dim(\mathrm{Im}(g_{A(\cdot)})) \leq d_\xi$. $d_\xi = d_p$ (Theorem 10.4 (d)) i.e. $\dim(\mathrm{Im}(g_{A(\cdot)})) = d_p$. As $\mathcal{F} \subset \mathrm{Im}(g_{A(\cdot)})$, this implies that $\mathcal{F} = \mathrm{Im}(g_{A(\cdot)})$ i.e. $g_{A(\cdot)}$ is surjective onto $\mathcal{F}$. As $\dim(\mathcal{F}) = d_\xi$, $g_{A(\cdot)}$ is bijective.

By bijectivity of ψ, $\{p_i\}_{i=1}^{d_p}$ forms a basis of $\mathbb{R}^{d_p}$. $g_{A(\cdot)}$ and ψ map this basis to the same basis $\{f_i\}_{i=1}^{d_p}$ of $\mathcal{F}$. As both mappings are bijective, this implies that $g_{A(\cdot)} = \psi(\cdot)$. This means that $\forall e \in \mathcal{E}, {g_A}^{-1}(f^e) = \psi^{-1}(f^e)$ i.e. system parameters p^e are recovered. □

Proposition E.4 (Extension to Nonlinear Dynamics)**.** *For linearly parametrized systems, non-linear w.r.t. inputs and nonlinear dynamics model g_θ where θ is output by a linear hypernetwork A, $\exists \alpha > 0$ s.t. system parameters are perfectly identified on all environments $e \in \mathcal{E}$ that satisfy $\|\xi^e\| \leq \alpha$, if $\forall i \in [\![1, d_p]\!], g_{A(\alpha \frac{p_i}{\|p_i\|})} = f_i$.*

Proof. On environment $e \in \mathcal{E}$, g_{θ^e} is differentiable w.r.t. $\theta^e = A(\xi^e) = \theta_c + W\xi^e \in \mathbb{R}^{d_\theta}$. We perform a first order Taylor expansion of $g_{A(\cdot)}$ around $\mathbf{0}$. We note $\alpha > 0$, s.t. $\forall \xi^e \in \mathbb{R}^{d_\xi}$ that satisfy $\|\xi^e\| < \alpha$, we have $g_{\theta^e} = g_{\theta^c} + \nabla_\theta g_{\theta^c} W \xi^e$. $g_{A(\cdot)}$ is then linear in the neighborhood of $\mathbf{0}$ defined by α. $\forall i, \alpha \frac{p_i}{\|p_i\|}$ belongs to this neighborhood s.t. the proof of Theorem 10.5 applies to this neighborhood if $\forall i \in [\![1, d_p]\!], g_{A(\alpha \frac{p_i}{\|p_i\|})} = f_i$. □

We now show the validity of the unicity condition (Theorem 10.4c) for two linearly parametrized systems.

Lemma E.1. *There is an unique set of parameters in $\mathbb{R}^4$ for a Lotka-Volterra (LV) system.*

Proof.

$$\text{With } \psi : c \triangleq (\alpha, \beta, \delta, \gamma) \mapsto \left[\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} \alpha x - \beta xy \\ \delta xy - \gamma y \end{pmatrix} \right]$$

a surjective linear mapping from $\mathbb{R}^4$ to $\mathcal{F}$ (all LV systems are parametrized). Injectivity of ψ i.e. $\psi(c_1) = \psi(c_2) \iff c_1 = c_2$ will imply bijectivity i.e. unicity of parameters for a LV system. As ψ is linear, injectivity is equivalent to $\psi(c) = 0 \iff c = 0$, shown below:

$$\begin{aligned}
\psi(c) = 0 &\iff \forall \begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} \alpha x - \beta xy \\ \delta xy - \gamma y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
&\iff \forall \begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} x(\alpha - \beta y) \\ (\delta x - \gamma) y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
&\iff \forall \begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} \alpha - \beta y \\ \delta x - \gamma \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
&\iff c = (\alpha, \beta, \delta, \gamma) = (0, 0, 0, 0)
\end{aligned}$$

□

Lemma E.2. *There is an unique set of parameters in $\mathbb{R}^{d+1}$, where d is the grid size, for a Navier-Stokes (NS) system.*

Proof. With $\psi : c \triangleq (\nu, f) \mapsto \Big[w \mapsto -v\nabla w + \nu \Delta w + f\Big]$, a surjective linear mapping from $\mathbb{R}^{d+1}$ to $\mathcal{F}$ (all NS systems are parametrized), bijectivity of ψ is induced by injectivity i.e. $\psi(c_1) = \psi(c_2) \iff c_1 = c_2$, shown below:

$$\begin{aligned}
&\psi(c_1) = \psi(c_2) \\
\iff &\forall w, -v\nabla w + \nu_1 \Delta w + f_1 = -v\nabla w + \nu_2 \Delta w + f_2 \\
\iff &\forall w, (\nu_1 - \nu_2)\Delta w = -(f_1 - f_2) \\
\iff &(\nu_1, f_1) = (\nu_2, f_2) \iff c_1 = c_2
\end{aligned}$$

□

E.4 Low-rank Assumption

When the systems are nonlinearly parametrized, we show empirically with Figure E.2 that the low-rank assumption is still reasonable for two different systems.

Glycolitic-Oscillator (GO) We consider the Glycolitic-Oscillator system (GO), described in Appendix E.6.1, which is nonlinear w.r.t. K_1. We vary parameters k_1, K_1 in Equation (E.5) across environments. We observe in Figure E.2 (Left, Middle) that there are three main gradient directions with SVD. The first is the

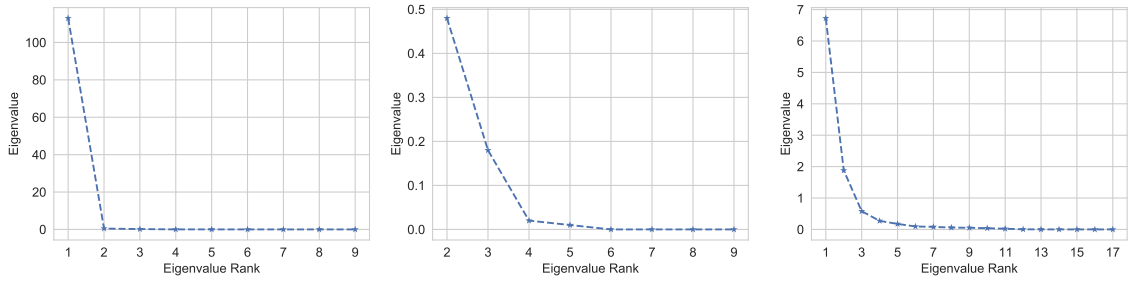


Figure E.2. – Ranked singular values of the gradients across environments $\mathcal{E}_{\text{tr}}$, $\mathcal{G}_{\theta^c}$ for CoDA-ℓ_1. On the Left and Middle, we consider `GO` where k_1 and K_1 vary across $\mathcal{E}$. On the right, we consider `Sin`.

most significant one while the second and third ones are orders of magnitude smaller.

Sinusoidal (`Sin`) We consider a sinusoidal family of functions $S(n) = \{f : \mathbb{R} \to \mathbb{R} | f(x) = \sum_{i=1}^{N} \lambda_i \sin(\omega_i x + \phi_i)\}$ (`Sin`). We sample 20 environments that correspond each to different amplitudes (uniformly sampled in $[0, 1]$), frequencies (uniformly sampled in $[0, 10]$)) and phases (uniformly sampled in $[0, 3.14]$). We depict in Figure E.2 (Right) the evaluation of the singular values at initialization. Figure E.2 (Right) shows that the number of directions to consider for convergence is small and that a single direction accounts for a significant amount of the variance in the gradients. This corroborates the low-rank assumption.

E.5 Locality Constraint

We derive the upper-bounds to $\|\cdot\|$ for two variations.

$\|\cdot\| = \ell_2$: we apply triangle inequality to obtain $\Omega = \ell_2^2$

$$\|W\xi^e\|_2^2 \leq \|W\|_2^2 \|\xi^e\|_2^2$$

$\|\cdot\| = \ell_1$: we apply Cauchy-Schwartz inequality to obtain $\Omega(W) = \ell_{1,2}(W) \triangleq \sum_{i=1}^{d_\theta} \|W_{i,:}\|_2$

$$\|W\xi^e\|_1 = \sum_{i=1}^{d_\theta} |W_{i,:}\xi^e| \leq \|\xi^e\|_2 \sum_{i=1}^{d_\theta} \|W_{i,:}\|_2$$

Equation (10.11) minimizes the $\log$ of the above upper-bounds.

E.6 Experimental Settings

We present in Appendix E.6.1 the equations and the data generation specificities for all considered dynamical systems.

E.6.1 Dynamical Systems

Lotka-Volterra (LV, (Lotka 1925)) The system describes the interaction between a prey-predator pair in an ecosystem, formalized into the following ODE:

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= \alpha x - \beta xy \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= \delta xy - \gamma y \end{aligned} \tag{E.4}$$

where x, y are respectively the quantity of the prey and the predator, $\alpha, \beta, \delta, \gamma$ define how two species interact.

We generate trajectories on a temporal grid with $\Delta t = 0.5$ and temporal horizon $T = 10$. We sample on each environment $N = 4$ initial conditions for training from a uniform distribution $p(X_0) = \mathrm{Unif}([1, 3]^2)$. We sample for evaluation 32 initial conditions from $p(X_0)$. Across environments, $\alpha = 0.5, \gamma = 0.5$. For training, we consider $\#\mathcal{E}_{\text{tr}} = 9$ environments with parameters $\beta, \delta \in \{0.5, 0.75, 1.0\}^2$. For adaptation, we consider $\#\mathcal{E}_{\text{ad}} = 4$ environments with parameters $\beta, \delta \in \{0.625, 1.125\}^2$.

Glycolytic-Oscillator (GO, (Daniels and Nemenman 2015)) GO describes yeast glycolysis dynamics with the ODE:

$$\begin{aligned} \frac{\mathrm{d}S_1}{\mathrm{d}t} &= J_0 - \frac{k_1 S_1 S_6}{1 + (1/K_1^q) S_6^q} \\ \frac{\mathrm{d}S_2}{\mathrm{d}t} &= 2\frac{k_1 S_1 S_6}{1 + (1/K_1^q) S_6^q} - k_2 S_2 (N - S_5) - k_6 S_2 S_5 \\ \frac{\mathrm{d}S_3}{\mathrm{d}t} &= k_2 S_2 (N - S_5) - k_3 S_3 (A - S_6) \\ \frac{\mathrm{d}S_4}{\mathrm{d}t} &= k_3 S_3 (A - S_6) - k_4 S_4 S_5 - \kappa (S_4 - S_7) \\ \frac{\mathrm{d}S_5}{\mathrm{d}t} &= k_2 S_2 (N - S_5) - k_4 S_4 S_5 - k_6 S_2 S_5 \\ \frac{\mathrm{d}S_6}{\mathrm{d}t} &= -2\frac{k_1 S_1 S_6}{1 + (1/K_1^q) S_6^q} + 2k_3 S_3 (A - S_6) - k_5 S_6 \\ \frac{\mathrm{d}S_7}{\mathrm{d}t} &= \psi \kappa (S_4 - S_7) - k S_7 \end{aligned} \tag{E.5}$$

where $S_1, S_2, S_3, S_4, S_5, S_6, S_7$ represent the concentrations of 7 biochemical species. We generate trajectories on a temporal grid with $\Delta t = 0.05$ and temporal horizon $T = 1$. We sample on each environment $N = 32$ initial conditions for training from a uniform distribution $p(X_0)$ defined in Table 2 in (Daniels and Nemenman 2015). We sample for adaptation 1 initial condition from $p(X_0)$. Across environments, $J_0 = 2.5, k_2 = 6, k_3 = 16, k_4 = 100, k_5 = 1.28, k_6 = 12, q = 4, N = 1, A = 4, \kappa = 13, \psi = 0.1, k = 1.8$. For training, we consider $\#\mathcal{E}_{\text{tr}} = 9$ environments with parameters $k_1 \in \{100, 90, 80\}, K_1 \in \{1, 0.75, 0.5\}$. For adaptation, we consider $\#\mathcal{E}_{\text{ad}} = 4$ environments with parameters $k_1 \in \{85, 95\}, K_1 \in \{0.625, 0.875\}$.

Gray-Scott (GS, (Pearson 1993)) The PDE descibes a reaction-diffusion system with complex spatiotemporal patterns through the following 2D PDE:

$$\begin{aligned} \frac{\partial u}{\partial t} &= D_u \Delta u - uv^2 + F(1-u) \\ \frac{\partial v}{\partial t} &= D_v \Delta v + uv^2 - (F+k)v \end{aligned} \tag{E.6}$$

where u, v represent the concentrations of two chemical components in the spatial domain S with periodic boundary conditions. D_u, D_v denote the diffusion coefficients respectively for u, v and F, k are the reaction parameters.

We generate trajectories on a temporal grid with $\Delta t = 40$ and temporal horizon $T = 400$. S is a 2D space of dimension 32×32 with spatial resolution of $\Delta s = 2$. We define initial conditions $(u_0, v_0) \sim p(X_0)$ by uniformly sampling three two-by-two squares in S. These squares trigger the reactions. $(u_0, v_0) = (1 - \epsilon, \epsilon)$ with $\epsilon = 0.05$ inside the squares and $(u_0, v_0) = (0, 1)$ outside the squares. We sample on each environment $N = 1$ initial conditions for training. We sample for adaptation 1 initial condition. Across environments, $D_u = 0.2097, D_v = 0.105$. For training, we consider $\#\mathcal{E}_{\text{tr}} = 4$ environments with parameters $F \in \{0.30, 0.39\}, k \in \{0.058, 0.062\}$. For adaptation, we consider $\#\mathcal{E}_{\text{ad}} = 4$ environments with parameters $F \in \{0.33, 0.36\}, k \in \{0.59, 0.61\}$.

Navier-Stokes (NS, (Stokes 1851)) NS describes the dynamics of incompressible flows with the 2D PDE:

$$\begin{aligned} \frac{\partial w}{\partial t} &= -v \nabla w + \nu \Delta w + f \\ \nabla v &= 0 \\ w &= \nabla \times v \end{aligned} \tag{E.7}$$

where v is the velocity field, $w = \nabla \times v$ is the vorticity. Both v, w lie in a spatial domain S with periodic boundary conditions, ν is the viscosity and f is the

constant forcing term in the domain S. We generate trajectories on a temporal grid with $\Delta t = 1$ and temporal horizon $T = 10$. S is a 2D space of dimension 32×32 with spatial resolution of $\Delta s = 1$. We sample on each environment $N = 16$ initial conditions for training from $p(X_0)$ as in Zongyi Li et al. (2021). We sample for adaptation 1 initial condition from $p(X_0)$. Across environments, $f(X, Y) = 0.1(\sin(2\pi(X+Y)) + \cos(2\pi(X+Y)))$. For training, we consider $\#\mathcal{E}_{\text{tr}} = 5$ environments with parameters $\nu \in \{8 \cdot 10^{-4}, 9 \cdot 10^{-4}, 1.0 \cdot 10^{-3}, 1.1 \cdot 10^{-3}, 1.2 \cdot 10^{-3}\}$. For adaptation, we consider $\#\mathcal{E}_{\text{ad}} = 4$ environments with parameters $\nu \in \{8.5 \cdot 10^{-4}, 9.5 \cdot 10^{-4}, 1.05 \cdot 10^{-3}, 1.15 \cdot 10^{-3}\}$.

E.6.2 Implementation and Hyperparameters

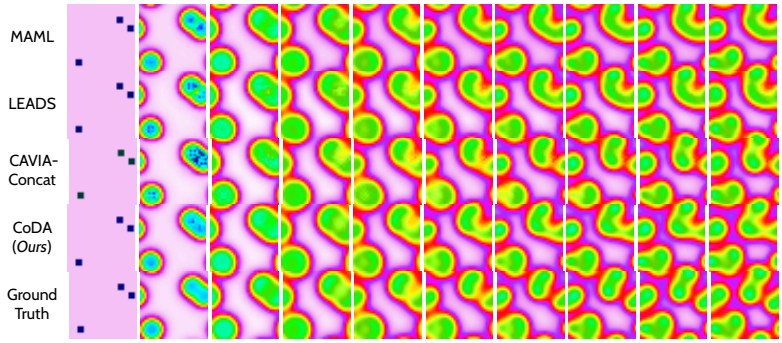
Architecture and Solver We implement the dynamics model g_θ with the following architectures:

- for `LV` and `GO`, 4-layer MLPs with 64-dimension hidden layers
- for `GS`, 4-layer ConvNet with 64-channel hidden layers, and 3×3 convolution kernels
- for `GS` and a Fourier Neural Operator Zongyi Li et al. 2021 with 4 spectral convolution layers for `NS`. The number of frequency modes is 12 and the hidden layers have 10 dimensions.

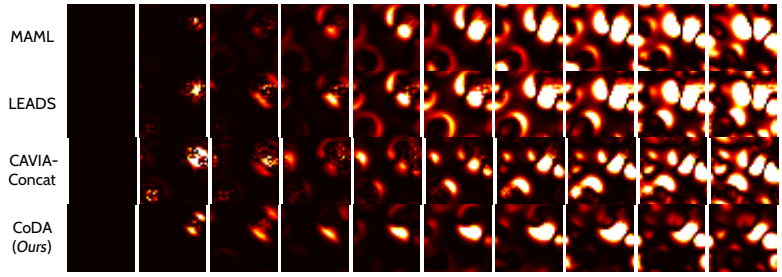
We apply Swish activation (Ramachandran et al. 2018) on all architectures and RK4 solver for `LV`, `GS`, `GO` and Euler solver for `NS`. The hypernet A is a single affine layer NN.

Optimizer We use the Adam optimizer (Diederik P. Kingma and Ba 2015) with learning rate 10^{-3} and $(\beta_1, \beta_2) = (0.9, 0.999)$. We apply early stopping. All experiments are performed with a single NVIDIA Titan Xp GPU on an internal cluster. For GBML methods, we choose a single inner-loop / outer-loop step to maintain low running times. We distribute training by batching together predictions across trajectories to reduce running time. States across batch elements are concatenated.

Hyperparameters We define hyperparameters for the following models: (a) CoDA: • `LV`: $\lambda_\xi = 10^{-4}$, $\lambda_{\ell_1} = 10^{-6}$, $\lambda_{\ell_2} = 10^{-5}$ • `GO`: $\lambda_\xi = 10^{-3}$, $\lambda_{\ell_1} = 10^{-7}$, $\lambda_{\ell_2} = 10^{-7}$ • `GS`: $\lambda_\xi = 10^{-2}$, $\lambda_{\ell_1} = 10^{-5}$, $\lambda_{\ell_2} = 10^{-5}$ • `NS`: $\lambda_\xi = 10^{-3}$, $\lambda_{\ell_1} = 2 \cdot 10^{-3}$, $\lambda_{\ell_2} = 2 \cdot 10^{-3}$ (b) LEADS: we use the same parameters as Yin et al. (2021a). (c) GBML: the outer-loop learning rate is 10^{-3}, we apply 1-step inner-loop update for training and adaptation, and the inner-loop learning rate for each system is:



(a) Predicted trajectory



(b) Prediction MSE

Figure E.3. – Adaptation to new GS system - $(F, k) = (0.033, 0.061)$

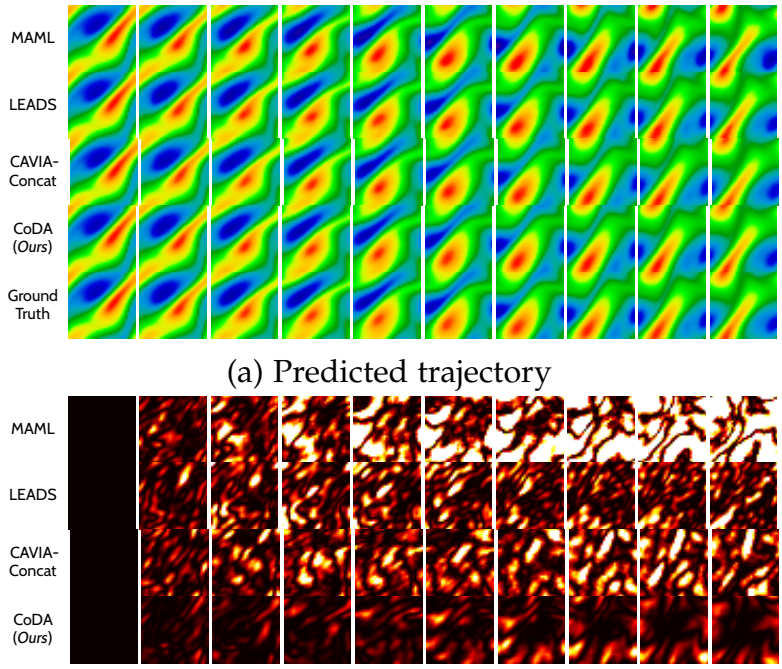
• LV: 0.1 • GO: 0.01 • GS: 10^{-3} • NS: 10^{-3}. These values are also used to initialize the per-parameter inner-loop learning rate in Meta-SGD.

E.7 Trajectory Prediction Visualization

We visualize in Figures E.3 and E.4 predicted trajectories by MAML, LEADS, CAVIA-Concat and CoDA-ℓ_1 along ground truth trajectories on the PDE systems NS and GS. We consider a new test trajectory on an *Adaptation* environment $e \in \mathcal{E}_{\text{ad}}$ with parameters defined in the caption.

E.8 Loss Landscape Visualization

We visualize loss landscapes of CoDA, ERM in Figure E.5.



(a) Predicted trajectory

(b) Prediction MSE

Figure E.4. – Adaptation to new NS system - $\nu^e = 1.15 \cdot 10^{-3}$

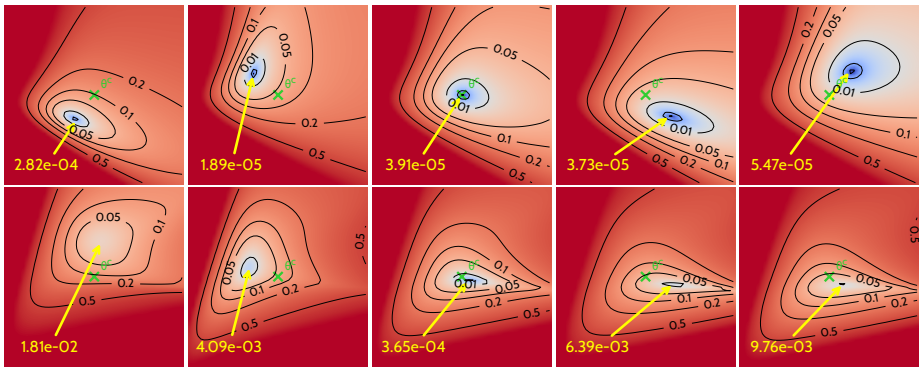


Figure E.5. – Loss landscapes around θ^e (CoDA's loss (Row 1) is projected onto subspace $\mathcal{W}$, with $d_\xi = 2$. ERM's loss (Row 2) is projected onto the two principal directions of the gradients computed with Singular Value Decomposition.

E.9 System Parameter Estimation

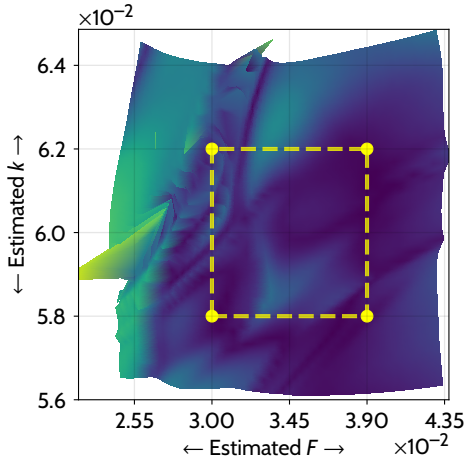


Figure E.6. – Parameter estimation MAPE (↓) and estimated parameters on GS over environments defined by $(F, k) \in [0.0225, 0.0435] \times [0.056, 0.064]$

APPENDIX F

SUPPLEMENTARY MATERIAL TO FEATURE SELECTION

F.1 The logitNormal Distribution:

The logitNormal distribution defines a probability distribution over the simplex, see Aitchison and Shen 1980 Initially introduced to describe compositional data Aitchison 1982, it is defined has:

Definition F.1. LogitNotmal Let X be random variable defined over $\mathbb{R}^n$ such that $X \sim \mathcal{N}(\mu, \sigma)$. Then, consider the following transformation:

$$Y_{-n} = e^X/(1 + \sum_{j=1}^{n} e^{X_j}), \text{ and } Y_{n+1} = 1 - \sum_{j=1}^{n} Y_j$$

Then the vector $Y = (Y_1, ...Y_{n+1})$ follows a logitNormal distribution denoted $\mathcal{LN}(\mu, \sigma)$ and is defined over the $\mathbb{R}^{n+1}$ simplex. Moreover, Y admits a density and can be found in Aitchison and Shen 1980.

If $Y \sim \mathcal{LN}(\mu, \sigma)$, it defines a probability distribution over the simplex which makes it practical to model compositional data, i.e data where the involved data forms some sort of proportion of a whole Aitchison 1982.

F.2 Reparametrizing the logitNormal Distribution:

Using Theorem F.1 of the logitNormal distribution, we can use the reparameterization trick in order to learn the parameters of a logitNormal law from samples of Normal law.

Theorem F.2. *Reparameterization: Let $X = (X_i)_{i \leq n}$ such that $X_i \sim \mathcal{N}(0,1)$ and all X_i are iid ($X \in \mathbb{R}^n$), $W \in \mathcal{M}_{m \times n}(\mathbb{R})$, and $b \in \mathbb{R}^m$, then :*

$$Y_{-n} = \exp(WX + b)/(1 + \sum_i \exp(W_i.X + b_i))$$

$$Y = (Y_{-n}, 1 - \sum_{j=1}^{n} Y_j) \tag{F.1}$$

$$Y \sim \mathcal{LN}(b, \Sigma) \tag{F.2}$$

This comes from the simple fact that an affine transformation of i.i.d. $\mathcal{N}(0,1)$ follows also a Normal law, which co-variance matrix can be expressed through the matrix of linear weights. Moreover, this advantageously correspond to a neural network layer with an extended sigmoidal function.

F.3 Proof For 0-Temperature

Here we prove the convergence of the reparameterization of the logitNormal law for the zero temperature.

Proof. Let $(\lambda_n)_{n \geq 0}$ be a positive sequence decreasing towards 0. We prove the o-temperature convergence for $z \sim \mathcal{N}(\mu, \sigma)$. Let $Y_n = \text{sigmoid}_{\lambda_n}(z)$. We investigate the convergence in distribution of Y_n towards a Bernoulli distribution. Let f be a continuous bounded function. We have:

$$\begin{aligned}\mathbb{E}(f(Y_n)) &= \int_0^1 f(Y_n) dP_{Y_n} = \int_{\mathbb{R}} f(\text{sigmoid}_{\lambda_n}(z)) dP_z \\ &= \int_{\mathbb{R}} f(\text{sigmoid}_{\lambda_n}(z)) \frac{1}{\sqrt{2\pi}\,\sigma} \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2} dz\end{aligned}$$

We first have point-wise convergence of the sequence of function inside the integral. Indeed,

If $z > 0, \lim_{n \to \infty} \text{sigmoid}_{\lambda_n}(z) = 1$.

If $z < 0, \lim_{n \to \infty} \text{sigmoid}_{\lambda_n}(z) = 0$. We have:

$$\lim_{n \to \infty} f(\text{sigmoid}_{\lambda_n}(z)) \frac{1}{\sqrt{2\pi}\,\sigma} \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2} = \frac{1}{\sqrt{2\pi}\,\sigma} f(\delta_{z>0}) \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2}$$

The domination is verified using the function:

$$g(z) = \frac{1}{\sqrt{2\pi}\,\sigma} ||f||_\infty \times \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2}$$

We can finally apply the theorem of dominated convergence:

$$\begin{aligned}
\lim_{n\to\infty} \mathbb{E}(f(Y_n)) &= \mathbb{E}(\lim_{n\to\infty} f(Y_n)) \\
&= \int \frac{1}{\sqrt{2\pi}\,\sigma} f(\delta_{z>0}) \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2} dz \\
&= \frac{1}{\sqrt{2\pi}\,\sigma} f(0) \int_{-\infty}^{0} \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2} dz \; + \frac{1}{\sqrt{2\pi}\,\sigma} f(1) \int_{0}^{+\infty} \exp^{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2} dz \\
&= f(0)\Phi(-\frac{\mu}{\sigma}) + f(1)(1 - \Phi(-\frac{\mu}{\sigma})) \\
&= \mathbb{E}_{b\sim\mathcal{B}(1-\Phi(\frac{-\mu}{\sigma}))} f(b),
\end{aligned}$$

where $\mathcal{B}$ denotes Bernoulli distribution. Finally, we can conclude that Y_n converges in law towards a Bernoulli distribution such that: $Y_n \to \mathcal{B}(1 - \Phi(\frac{-\mu}{\sigma}))$

□

F.4 Proof L_0-logitNormal:

$$\begin{aligned}
L_0(S_\theta(z)) &= \sum_i 1 - \mathbb{P}(\bar{z} \leq 0) \\
&= \sum_i 1 - \mathbb{P}(\text{sigmoid}(Wz + b) \leq -\gamma/(\eta - \gamma)) \\
&= \sum_i 1 - \mathbb{P}(W_i.z \leq \log(\frac{-\gamma}{\eta}) - b) \\
&\quad \text{as } W_i.z \text{ has a normal law } \mathcal{N}(0, \sqrt{\sum_j w_{j,i}^2}) \\
&= \sum_i 1 - \Phi(\frac{\log(\frac{-\gamma}{\eta}) - b}{\sqrt{\sum_j W_{j,i}^2}})
\end{aligned}$$

F.5 Ill poseness of the ℓ_1-formulation:

Consider the auto encoding setting with a ℓ_1-norm instead of the derived L_0. The optimization problem is:

$$\mathcal{L}_{\ell_2} = \lambda_{\ell_2} \mathbb{E}_{x\sim p_x} ||x - G_\phi(S_\theta(z) \odot x)||_2 + \lambda_s . L_1(S_\theta) \tag{F.3}$$

Let (G^*_ϕ, S^*_θ) be an optimal solution, i.e that realizes the minimum of the above optimization cost function. Then, consider: $S_2 = S^*_\theta/2$ and G_2 defined as $G_2(x) = G^*_\theta(2 * x)$. Then the MSE term of eq. (F.3) for the couple (G_2, S_2) is equivalent as the one with $(G*_\phi, S*_\theta)$, however the ℓ_1-norm of (S_2) is lower. Therefore (G^*_ϕ, S^*_θ) is not optimal and the problem of eq. (F.3) is ill-posed. However, note that, in the case of binary vectors, ℓ_0-norm and ℓ_1-norm are equals.

F.6 On the Stretching Scheme:

We initially start from a distribution p that lives in $[0, 1]$ and need to transform it in order to obtain a non zero probability of sampling 0 while maintaining both tractability and differentiability. We denote this function f. We need $f^{-1}(0)$ to be a non-zero measure set of the original support. In other words, we need f to be a surjection, and $f^{-1}(0)$ to be Lebesgue measurable with a non zero mass. Instead of the HT function we could have used a stretched $relu$ function. One significant advantage of the chosen function is that it also creates a non-zero probability of sampling 1 therefore enforcing the binary behaviour of our masks. Unbalanced binary scheme can also be investigated in future works. Indeed one can think of creating a higher portion of the stretched distribution above one, enforcing the binary behaviour of the mask.

F.7 Algorithm

We present here the algorithm for the proposed logitNormal based feature selection algorithm.

F.8 Practical Consideration on the Temperature:

As duely noted by Maddison et al. 2016, the temperature in sigmoid activation plays a crucial role in the training. This remark holds for our work. Indeed, in our work decreasing the temperature in the sigmoid, amounts to increase the variance and the absolute value of the average of the initial Gaussian distribution.

Also aiming at approximating binary distribution, we don't want any interior mode as in the green curve depicted in fig. 11.1: $\mathcal{LN}(0, 1)$ has an interior maximum point. This case is not acceptable for the approximation of Bernoulli random variable as, it could allow a leakage of information, i.e the distribution is not approximating a binary distribution anymore. Therefore, during training one

Algorithm F.1 Differentiable Feature Selection

Result: Converged S G_ϕ
Initialize $\theta = (W, b)$ and G_ϕ
while *Convergence not reached* **do**
  sample batch $x = (x_1, ...x_n)$ and $z = (z_1, ..., z_n)$, such that $z_i \sim \mathcal{N}(0, I_d)$
  Compute $S_\theta(z) = \text{sigmoid}_\lambda(Wz + b)$ and the observations $x^{obs} = \bar{S}_\theta(z) \odot x$
  Estimate reconstruction $\hat{x} = G_\phi(x^{obs})$
  $L = ||x - \hat{x}||_2 + \lambda_{sparse} L_0(\bar{S}_\theta(z))$
  Update ϕ and θ:

$$\phi \leftarrow \phi - \frac{\partial L}{\partial \phi}$$

$$\theta \leftarrow \theta - \frac{\partial L}{\partial \theta}$$

end

should ensure that the learned distribution has no interior maxima. Fortunately, it suffices to sufficiently decrease the temperature λ of the sigmoid in order to recover two modes at 0 and 1. Indeed, decreasing sufficiently the temperature in the sigmoid pushes the interior maximum towards the edges. In practice, we observe that initializing our W so that $W.z$ with a variance higher than 0.5 with a temperature of $\lambda = 0.3$ suffices.

F.9 Removing the Randomness

Both our propositions of eq. (11.6) or eq. (11.7) estimates distribution in the spaces of binary variables. To collapse the distribution, one can take advantage of theorem 11.1 and select the K desired number of features. One can also, empirically select the K features the mask with the highest probability to be selected. Both approaches lead to similar results in practice. Note that in both cases, if K is far from the observed number of pixel, the selected features may not be the best subset of the learned distribution.

In practice, we chose to collapse the distribution using Theorem 11.1: We first estimate the expected ℓ_0-norm of the distribution, which equals to $\sum(1 - \phi(-\frac{\mu_i}{\sigma_i}))$. Let L_0 be the value of the expected ℓ_0-norm of our learned distribution. We then select two masks made of the most likely features to be selected: the first one has L_0 rounded *down* to the nearest ten pixels. The other one has L_0 rounded *up* to the nearest ten selected pixels. Note that, for SCT baseline, we use a property similar to Theorem 11.1 for the concrete distribution, available in Maddison et al. 2016.

F.10 Concrete Law

Introduced by Maddison et al. 2016 to approximate discrete variables, binary concrete random variable is defined as follows:

$$\begin{aligned} u &\sim \mathcal{U}([0,1]) \\ G &= \log(u) - \log(1-u) \\ X &= \text{sigmoid}\Big(\frac{\log(\alpha) + G}{\lambda}\Big), \end{aligned}$$

And X follows a relaxed binary concrete law.

F.11 Experimental Details

All experiments were trained on Titan XP GPU via using Pytorch framework and mixed precision training. For all experiments the expected ℓ_0-norm is normalized by the number of pixels in the signal. Also for all algorithms trained using correlated logitNormal approach, the dimension of z is 16, i.e. $z \sim \mathcal{N}(0, I_{16})$.

For all mask based methods, G_ϕ is a resent following the implementation of Isola et al. 2016 with 2 residual blocks and 16 filters.

F.11.1 Mnist

All masked based algorithms were trained using ADAM optimizer with $\beta = (0.9, 0.99)$ and a learning rate of 2.10^{-4} for 550 epochs with batch size 256. CAE method was trained for 1400 epochs with a temperature decreasing form 10 to 0.01 following recommendation of the authors.

F.11.2 Climate Data

All masked based algorithms were trained using ADAM optimizer with $\beta = (0.9, 0.99)$ and a learning rate of 2.10^{-4} for 550 epochs with batch size 128. CAE method was trained for 1400 epochs with a temperature decreasing form 10 to 0.01 following recommendation of the authors.

F.11.3 CelebA

All masked based algorithms were trained using ADAM optimizer with $\beta = (0.9, 0.99)$ and a learning rate of 2.10^{-4} for 140 epochs with batch size 128. CAE method was trained for 400 epochs with a temperature decreasing form 10 to 0.01 following recommendation of the authors.

F.11.4 Hyperparameters Search

Except for CAE where the number of selected features is a structural constraint, we search the hyperparameter space by sampling from the interval $[10^{-2}; 1]$ discretized by steps of 3.10^{-2}. For all dataset, the CAE method was trained with a decreasing temperature from 10 to 0.01 following the guidelines of the authors Abid et al. 2019. For the mask method based on the concrete distribution the temperature of the sigmoid was set to $\lambda = 2/3$ following the recommendation of Maddison et al. 2016. For logitNormal based algorithm, the temperature was fixed to $\lambda = 0.3$.

F.11.5 Initialization

For all mask based methods, we chose the initialization parameters so that the resulting distribution of the each variable in the mask is symmetrical, with as many chances to be sampled than to be rejected, i.e. for all variable i in the masks: $\mathbb{P}(S_\theta(z)_i < \epsilon) \approx \mathbb{P}(S_\theta(z)_i > 1 - \epsilon) \approx 0.2$. That way, all distribution can explore the space of binary masks. Also, in order to verify whether a covariance matrix is learned during training for the logitNormal sampling method of eq. (11.6), W is initialized with using an uniform law.

F.12 Additional Samples:

F.13 cGAN Details and Samples

Simply speaking, a cGAN has two main learnable functions: a discriminator network with parameters ψ named D_ψ trained to differentiate "true" data labeled as 1 from data generated by G_ϕ labeled as 0. A generative network with parameter ϕ denoted G_ϕ. $G_\phi : \mathbb{R}^p \times \mathbb{R}^{n \times n} \to \mathbb{R}^n$ takes as input a random variable $\gamma \in \mathbb{R}^p$ and our conditional information $x^{obs} = \bar{S}_\theta(z) \odot x \in \mathbb{R}^{n \times n}$, and aims at fooling D_ψ,

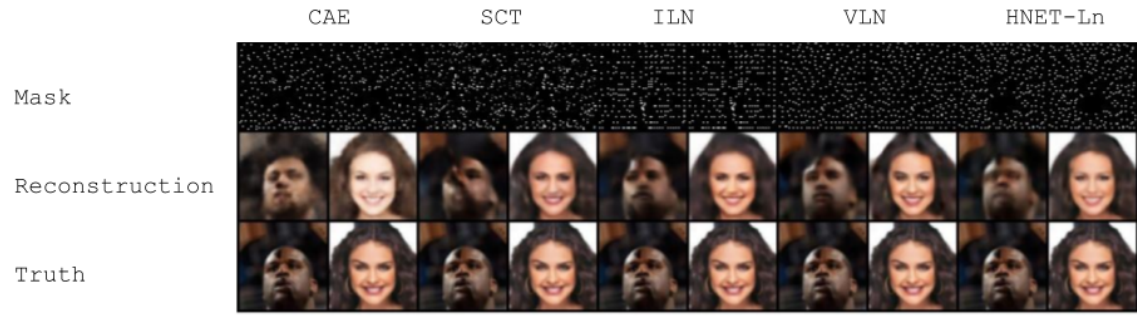


Figure F.1. – Sample of masks (first row), Reconstruction (second row) and True Data (Last row) for CelebA dataset on all considered algorithms for 200 features with ℓ_2-encoding

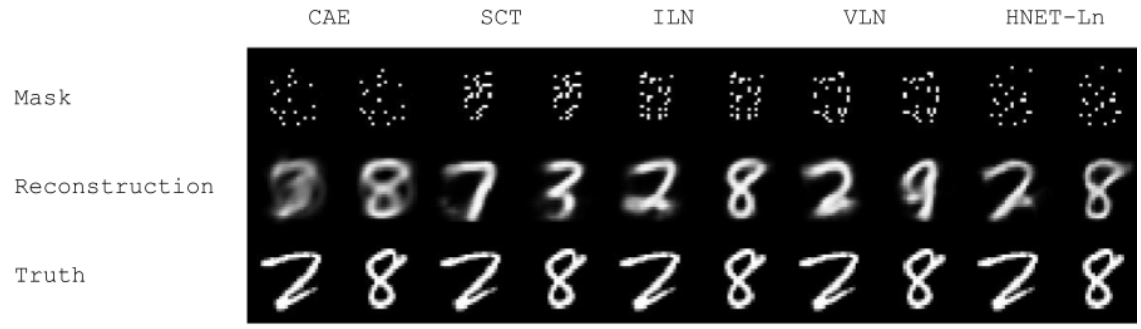


Figure F.2. – Sample of masks (first row), Reconstruction (second row) and True Data (Last row) for Mnist dataset on all considered algorithms for 20 selected features with ℓ_2-encoding

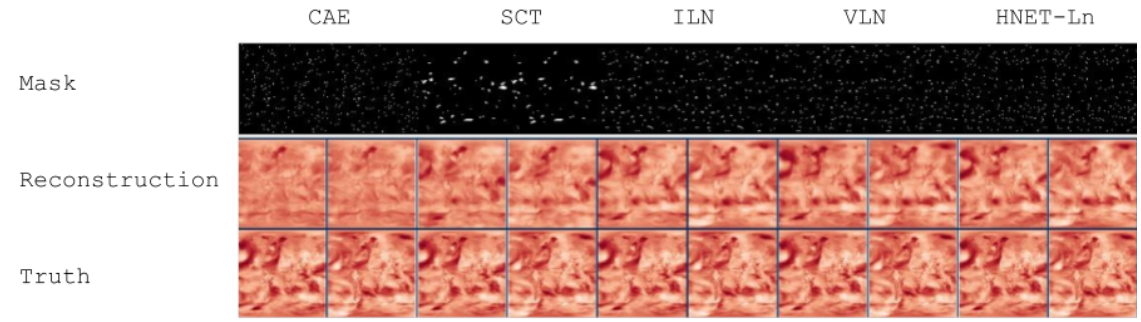


Figure F.3. – Sample of masks (first row), Reconstruction (second row) and True Data (Last row) for the Geophysical Dataset on all considered algorithms for 200 features with ℓ_2-encoding

making it classify the conditionally generated images as true. For our experiments

Figure F.4. – Samples of masks (first row), reconstruction (second row) and true data (last row) for Mnist dataset obtained using a cGAN approach following Isola et al. 2016, i.e including a ℓ_1+Gan loss as reconstruction objective for approximately 15 sampled pixels ($\lambda_s = 100$)

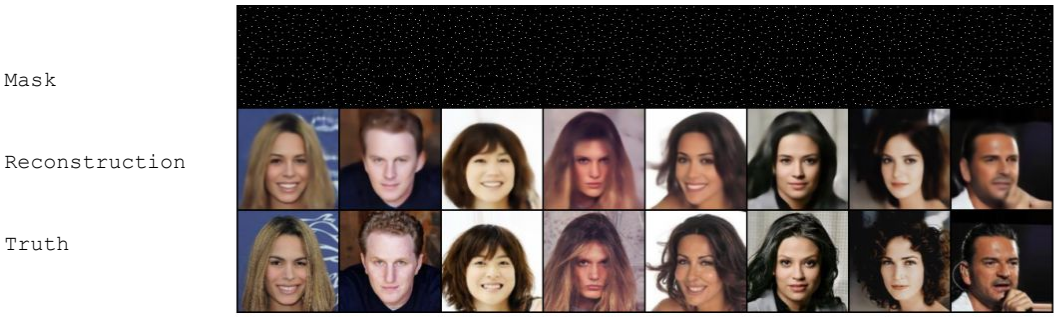


Figure F.5. – Samples of masks (first row), reconstruction (second row) and true data (last row) for CelebA dataset obtained using a cGAN approach following Isola et al. 2016, i.e including a ℓ_1+Gan loss as reconstruction objective for approximately 1.7% sampled pixels ($\lambda_s = 100$)

we used the cGAN implementation of Isola et al. 2016 optimizing the following loss, with $x^{obs} = x \odot \bar{S}_\theta(z)$:

$$\min_{\phi,\theta} \max_{\psi} \mathbb{E}_{z,x} \log D_\psi(x, x^{obs}) + \mathbb{E}_{z,x} \log \{1 - D_\psi(G_\phi(x^{obs}), x^{obs})\} + \lambda_{sparse} \times \ell_0(\bar{S}_\theta(z)) + \lambda_{rec} \times \ell_1(x - G_\phi(x^{obs})), \tag{F.4}$$

Consider S_θ fixed, one interesting advantage about the cGAN approach is that we can prove that the optimal distribution p_{G_ϕ} for G_ϕ is given x^{obs}: $p_{G_\phi}(x, x^{obs}) = p_{x \sim data}(x|x^{obs})$ which means that G_ϕ will sample according to the observed data distribution.

Proof. To lighten notation, we will use the notation $y = x \odot \bar{S}_\theta(z)$ as conditioning variable, giving the following game value function:

$$V(G, D) = \mathbb{E}_{x,y} \log D(x, y) + \mathbb{E}_{z,y} \log\{1 - D(G(z, y), y)\}$$

Following I. J. Goodfellow et al. 2014, we can write:

$$V = \int_{x,y} \log D(x, y) p_x(x, y) dx dy + \int_{z,y} \log\{1 - D(G(z, y), y)\} p_z(z) p_y(y) dz dy$$

if G induce a distribution p_g,

$$V = \int_{x,y} [\log D(x, y) p_x(x|y) p_y(y) + \log\{1 - D(x, y)\} p_g(x|y) p_y(y) dx dy]$$

$$= \int_y \Big(\int_x \log D(x, y) p_x(x|y) + \log\{1 - D(x, y)\} p_g(x|y) \Big) p_y(y) dy$$

Then classically the maximal value of $x \to a \log(x) + b \log(1 - x)$ is reached in $\frac{a}{a+b}$ Thus, given y, the optimal distribution followed by D:

$$p_D(x, y) = \frac{p_x(x|y)}{p_x(x|y) + p_G(x|y)}$$

The optimal distribution of G is completely doable at y fixed following the original reasoning of I. J. Goodfellow et al. 2014 □