Deep learning for spatio-temporal forecasting - application to solar energy

Vincent Le Guen

HESAM UNIVERSITÉ

le cnam

ÉCOLE DOCTORALE Sciences des Métiers de l'Ingénieur
Centre d'études et de recherche en informatique et communications

THÈSE

présentée par : **Vincent LE GUEN**
soutenue le : **30 novembre 2021**

pour obtenir le grade de : **Docteur d'HESAM Université**

préparée au : **Conservatoire national des arts et métiers**

Discipline : **Informatique**

Spécialité : **Informatique**

Deep learning for spatio-temporal forecasting - application to solar energy

THÈSE dirigée par :
M. Nicolas THOME , Professeur, Conservatoire national des arts et métiers

Jury

M. Greg MORI	
Professeur, Simon Fraser University et directeur scientifique, Borealis AI	Rapporteur
M. Patrick PEREZ	
Directeur scientifique Valeo AI	Rapporteur
M. Patrick GALLINARI	
Professeur, Sorbonne Université et chercheur senior, Criteo AI	Président du jury
M. Philippe BLANC	
Professeur, Mines ParisTech	Examinateur
Mme Stéphanie DUBOST	
Docteure, EDF R&D	Examinatrice
Mme Elisa FROMONT	
Professeur, Université Rennes 1, IRISA	Examinatrice
M. Etienne MEMIN	
Directeur de recherche INRIA	Examinateur
M. Nicolas THOME	
Professeur, Conservatoire national des arts et métiers	Directeur de thèse

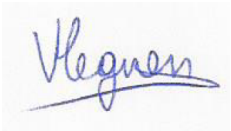
THÈSE

le cnam

Affidavit

Je soussigné, Vincent Le Guen, déclare par la présente que le travail présenté dans ce manuscrit est mon propre travail, réalisé sous la direction scientifique de Nicolas Thome (directeur de thèse), dans le respect des principes d'honnêteté, d'intégrité et de responsabilité inhérents à la mission de recherche. Les travaux de recherche et la rédaction de ce manuscrit ont été réalisés dans le respect de la charte nationale de déontologie des métiers de la recherche. Ce travail n'a pas été précédemment soumis en France ou à l'étranger dans une version identique ou similaire à un organisme examinateur.

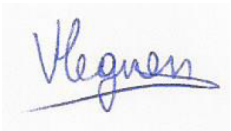
Fait à Paris, novembre 2021



Affidavit

I, undersigned, Vincent Le Guen, hereby declare that the work presented in this manuscript is my own work, carried out under the scientific direction of Nicolas Thome (thesis director), in accordance with the principles of honesty, integrity and responsibility inherent to the research mission. The research work and the writing of this manuscript have been carried out in compliance with the French charter for Research Integrity.

Paris, November 2021



Remerciements

Je tiens à remercier ici toutes les personnes qui ont concouru à l'achèvement du travail présenté dans ce manuscrit.

Pour revenir chronologiquement sur le montage de ce projet, commencer une thèse de doctorat 5 ans après la sortie d'école d'ingénieur a été en soi un premier défi. Je remercie chaleureusement mes collègues à EDF pour leur soutien, en particulier Nicolas Paul, Bruno Charbonnier, Loic Vallance. J'ai aussi une grande reconnaissance pour Stéphanie Dubost pour toutes nos discussions utiles sur la prévision d'énergies et pour avoir défendu dès le début cette idée. Stéphanie a grandement oeuvré pour assurer le financement de ma thèse, réparti sur 4 projets et 4 programmes de recherche différents, ce qui était assez inédit ! J'ai aussi pu compter sur l'appui décisif de ma hiérarchie, Nicolas Roche, Julien Berland, dans cette démarche. Je remercie aussi Dominique Demengel pour son travail crucial depuis de nombreuses années sur l'instrumentation des caméras au sol et pyranomètres et la bonne qualité des données, sans quoi ce travail n'aurait pas été possible.

Ensuite, mes remerciements les plus sincères s'adressent à Nicolas Thome qui a dirigé ma thèse au conservatoire national des Arts et Métiers. Quand je suis venu la première fois dans son bureau à l'été 2018 avec mon sujet de thèse déjà écrit, Nicolas a accueilli l'idée avec grand intérêt. Ces 3 années de travail commun se sont révélées être intenses et très stimulantes. Nicolas a toujours été très actif pour guider nos réflexions et discuter de nouvelles pistes quand nous arrivions dans une impasse. Nicolas m'a beacoup appris sur comment mener un projet de recherche, et notamment sur la difficile tâche de rédaction d'articles scientifiques. Un grand merci pour tout le temps passé (soir et week-end compris) pour guider, relire, (ré)écrire, ce qui a été d'une importance cruciale pour l'acceptation des soumissions. Je remercie aussi Clément Rambour au CNAM, pour son co-encadrement très important sur mes travaux de dernière année et ses conseils judicieux.

Au cours de ces 3 années de thèse, j'ai pu passer de très bons moments avec les autres doctorants de l'équipe au CNAM: Olivier Petit, Thuy Le, Laura Calem, Charles Corbière, Rémy Sun, Elias Ramzi, Loïc Thémyr, Marc Lafon, Perla Doubinsky, Yannis Karmim. Malgré le confinement et le télétravail résultant de la crise sanitaire, nous avons réussi à maintenir des contacts techniques et conviviaux réguliers à distance. J'en retiendrai l'amitié, la solidarité et l'entraide nées de ces périodes de dur labeur.

Je remercie également mes collègues à EDF avec qui les discussions ont toujours été fructueuses, en particulier Charlotte Gauchet, Christophe Chaussin, Lorenzo Audibert, Louis Apffel, Georges Hebrail, Nicolas Bousquet, Benoît Braisaz, Eric Lajoie-Mazenc, Matthieu Chiodetti, Gerald Kwiatkowski.

Du côté de Sorbonne Université, je remercie Matthieu Cord et tous ses doctorants pour l'organisation

des réunions hebdomadaires "cordettes", à la fois studieuses et conviviales. J'ai également particulièrement apprécié les travaux avec Patrick Gallinari et ses doctorants Yuan Yin, Jérémie Dona, Ibrahim Ayed, Emmanuel de Bézenac, qui ont permis d'aboutir à une publication jointe très approfondie. Je remercie aussi Edouard Oyallon, ancien camarade au master MVA et maintenant chercheur CNRS, dont le regard sur nos travaux a été très pertinent. Sur ses conseils, nous avons collaboré de manière fructueuse avec Edouard Leurent, dont je salue la gentillesse et la disponibilité.

J'exprime ma gratitude pour tous les membres de mon jury de thèse pour avoir accepté d'évaluer mes travaux et pour leurs retours très pertinents: Greg Mori, Patrick Pérez, Patrick Gallinari, Philippe Blanc, Stéphanie Dubost, Elisa Fromont, Etienne Mémin.

Pour finir, je tiens à remercier mes parents, mon épouse pour leur soutien et patience durant ces 3 années très chargées, avec une pensée pour le petit Louis qui est venu au monde 1 mois avant ma soutenance de thèse.

Abstract

This thesis tackles the subject of spatio-temporal forecasting with deep learning, which is the task of forecasting complex phenomena represented by time series or videos, involving both complex temporal dynamics and strong spatial correlations. This is of crucial importance for many industrial applications, such as climate, healthcare or finance. The motivating application at Electricity de France (EDF) is short-term solar energy forecasting with fisheye images. Despite the great successes of deep learning in computer vision and natural language processing, pure data-driven methods still struggle in the task of physical process extrapolation, especially in data-scarce contexts and for non-stationary time series that can present sharp variations. We explore two main research directions for improving deep forecasting methods by injecting external physical knowledge. The first direction concerns the role of the training loss function. Instead of using the largely dominant mean squared error (MSE), we show that differentiable shape and temporal criteria, typically used as evaluation metrics in applications, can be leveraged to improve the performances of existing models. We address both the deterministic context with the proposed DILATE loss function and the probabilistic context, for which we aim at describing the predictive distribution with a small set of diverse and accurate scenarios, with our proposed STRIPE model. Our second direction is to augment incomplete physical models with deep data-driven networks for accurate forecasting. For video prediction, we introduce the PhyDNet model that disentangles PDE (partial differential equations) dynamics from residual information necessary for prediction, such as texture or details. We further propose a learning framework (APHYNITY) that ensures a principled and unique linear decomposition between physical and data-driven components under mild assumptions, leading to better forecasting performances and parameter identification. We validate our contributions on many synthetic and real-world datasets, and on the solar energy dataset at EDF.

Keywords : deep learning, machine learning, spatio-temporal forecasting, solar energy forecasting.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Content

1.1 Spatio-temporal forecasting

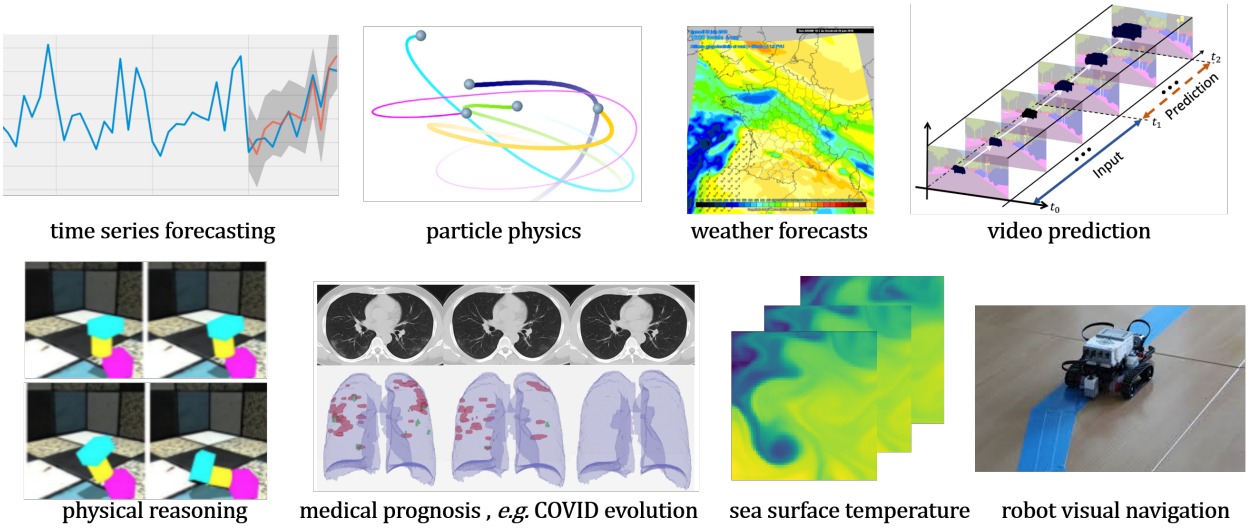


Figure 1.1: Spatio-temporal forecasting applications include time series forecasting, physical systems extrapolation, forecasting phenomena with visual data, generic video prediction, *etc.*

1.1.1 General context: perception vs extrapolation

In this thesis, we tackle the problem of spatio-temporal forecasting, which is the task of forecasting complex phenomena represented by time series or videos, involving both complex temporal dynamics and strong spatial correlations. Advances in this field could lead to immediate and possibly large impacts in the society. A wide range of sensitive applications heavily rely on accurate forecasts of uncertain events with potentially sharp variations for making decisions (see Figure 1.1). In weather and climate science, better anticipating floods, hurricanes, earthquakes or other extreme events could help taking emergency measures on time and save lives. In medicine, predicting the evolution of a disease is a particularly actual topic. In retail and business, accurately predicting the demand for a product is fundamental for stock management and profit maximization. For industrial applications, failure prediction is an important issue for maintenance.

We address spatio-temporal forecasting from a machine learning point of view, i.e. by leveraging training data for solving the task. Machine Learning (ML) is a subfield of Artificial Intelligence (AI) that is appealing for solving complex problems. Bolstered by the recent advances in computer hardware and the exponential growth of available data, ML has witnessed a renewed interest in the last decade from both academic and industrial actors. At the ImageNet competition in 2012, which consists in classifying images between 1000 categories, the deep neural network of Krizhevsky et al. [127] has for the first time outperformed traditional methods by a large margin. Given enough training data, Deep Learning (DL) can automatically learn meaningful representations useful for downstream tasks, replacing the manual feature extraction necessary in traditional ML algorithms. Since, Deep Learning has shown impressive results in many practical applications (see Figure 1.2), such as object detection [27], image segmentation [172], natural language understanding [61], or human speech recognition [2]. Combined with reinforcement learning, DL has led to super-human performance on many board games,

e.g. at the game of Go with alphaGo [230].

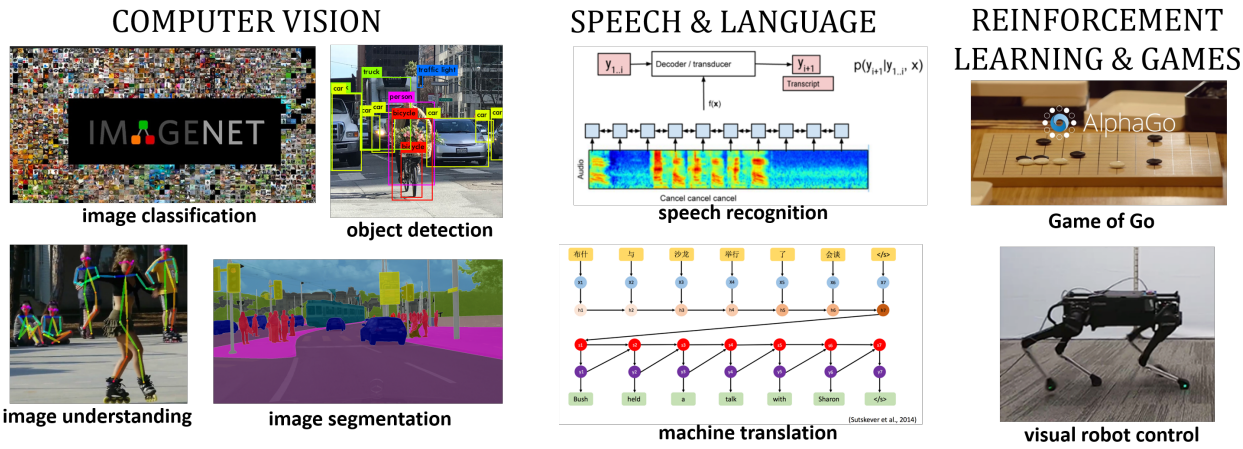


Figure 1.2: The main Artificial Intelligence and Deep learning successful applications include tasks linked to perception, such as computer vision, speech, language, reinforcement learning and games.

However, the successes of AI in these tasks are essentially linked to perception and not directly transferable to spatio-temporal forecasting. Modelling and extrapolating complex physical dynamics, such those arising in climate sciences, seems still beyond the scope of pure ML methods. The extrapolation task we address is quite different by nature from perception: future is inherently stochastic and multimodal, i.e. multiple outcomes may happen from the same context situation. Moreover, the volume of available data for learning complex dynamical systems such as in climate is by several orders of magnitude not sufficient still nowadays [224]. Many extreme events appear very scarcely in datasets and are thus highly challenging to learn from data.

1.1.2 Incorporating prior knowledge in machine learning models

To overcome these issues, injecting prior physical knowledge about the system is a key aspect for accurate extrapolation. This is an old question in machine learning that yet remains widely open. We illustrate in Figure 1.3 the main classes of methods for spatio-temporal forecasting.

On the right side of Figure 1.3, the traditional Model-Based (MB) approaches require a deep mathematical or physical understanding of the underlying phenomena. For time series, classical state space models (SSMs) [106, 21] explicitly exploit the trend and seasonality patterns. For physical processes, physicists attempt to model the dynamics with first principles, conservation laws or other empirical behaviours. This physical knowledge can often be formulated through ordinary or partial differential equations (ODE/PDE) with known coefficients. With data available for the initial and boundary conditions, forecasting is performed with numerical simulation solvers. This is the classical setting in many engineering fields, such as in mechanics (where systems are described by Newtonian mechanics) or in computational fluid dynamics (with the Navier-Stockes equations), and the numerical analysis solvers are well theoretically grounded.

However, this class of methods is limited in the case of *incomplete* physical models. Models can be considered incomplete in two situations. In the first case, the complexity of the phenomenon prevents

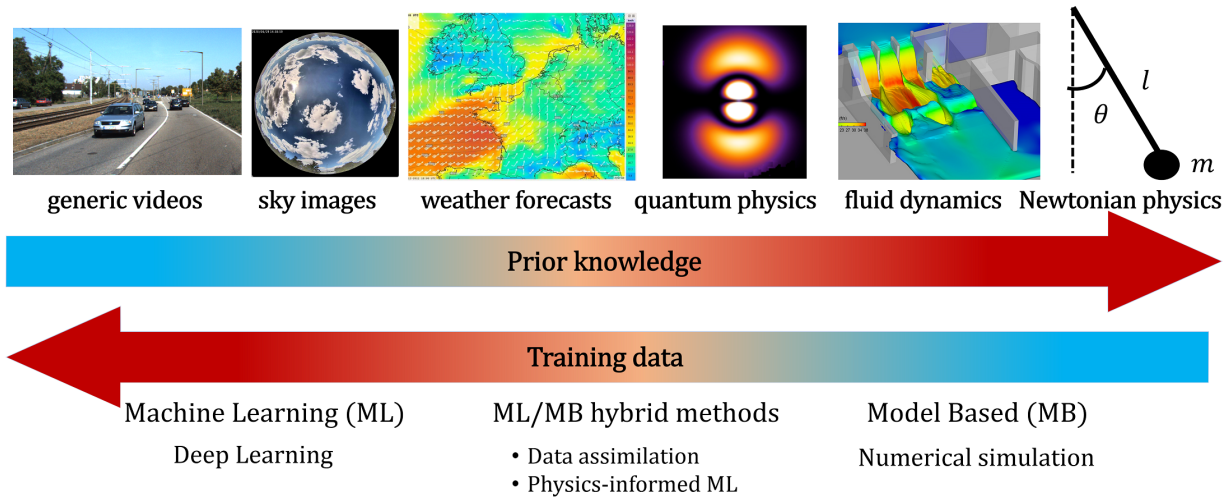


Figure 1.3: **Data *vs.* prior knowledge contexts.** On the left, Machine Learning (ML) and particularly Deep Learning can extrapolate dynamical systems with no prior information after training on a large dataset. On the right, traditional Model-Based (MB) approaches assume a full physical knowledge of the system and predict the future with numerical simulation from a set of initial and boundary conditions. In-between, with some data and a possibly incomplete knowledge, the ML/MB coupling is a very active and promising research direction that we explore in this thesis.

from deriving an exhaustive analytical description of the system. For example when modelling climate change, many complex interactions governing the state of the atmosphere are not modelled. The complete set of input variables of the system may also be unknown, e.g. when forecasting financial markets or human interactions. In the second case, certain approximations are made to make the complete equations numerically tractable. For example, the Schrödinger equation that governs the wave function of a quantum-mechanical system is not exactly solvable in many non-trivial situations. Solutions are typically computed by approximate numerical schemes and with several simplifying assumptions, e.g. the Born-Oppenheimer approximation. For computational time issues, the equations can also be solved on rather coarse meshes, which can prevent from capturing certain phenomena, e.g. the turbulence behaviour in computational fluid dynamics.

On the other side of the spectrum, Machine Learning (ML) represents a more prior-agnostic approach. Given a large amount of training data, deep learning has encountered impressive successes in automatically learning complex relationships without any prior knowledge, and has become state-of-the-art for many forecasting tasks, such as generic video prediction [277]. However, as discussed above, deep learning is still limited for modelling highly complex dynamics of natural phenomena such as climate; although more and more data is collected about the atmosphere with in-situ or remote sensing, it is still largely largely insufficient for matching the complexity of the task. Moreover, deep neural networks lack the physical plausibility required in several domains and cannot properly extrapolate to new conditions.

In-between, there exists a category of hybrid methods that combine MB approaches and data. Historically, data assimilation techniques [50, 18] leverage data to correct the predictions of physical models in presence of noisy observations. This includes the popular Kalman filter [116], particle filter [192] or 4D-Var [52] that have achieved great successes for many smoothing/filtering/forecasting

applications, for example for tracking objects in videos [191]. Data assimilation still constitutes the state-of-the-art paradigm for weather forecasting.

Revisiting the ML/MB cooperation with modern deep learning is an emerging research topic motivating a great interest in many communities, attested by the soaring number of publications and workshops in top ML conferences[1]. Physics can be leveraged in the training process of ML models, either as a soft constraint in the loss function [201, 231] or as hard constraints in the neural network topology [57, 174]. From the ML point of view, these physical constraints lead to more interpretable ML models compliant to physical laws that remain robust in case of noisy data. This typically results in an increased data efficiency and better extrapolation performances beyond the training domain. Another particularly appealing direction concerns identifying and discovering physical systems: data-driven models can learn the unknown coefficients or parts in parameterized PDEs [214, 155], and discover new physical connections from data [54].

In this thesis, we explore this category of hybrid methods and our contributions are targetted towards the following question:

How to properly exploit prior physical knowledge to improve Machine Learning forecasting models?

We focus on two particular directions: injecting prior knowledge in the training objective (part I) and designing augmented MB/ML neural architectures in the case of incomplete physical models (part II).

1.1.3 Industrial application at EDF: solar energy forecasting with fisheye images

At Electricité de France (EDF), the industrial use-case motivating this thesis is solar irradiance forecasting. With the increasing share of intermittent renewable energy sources such as solar or wind, accurately forecasting the electricity production and its possibly sharp variations is of great importance since the the consumption-production balance must be satisfied at every timestep. The possible data sources for this task are illustrated in Figure 1.4. Numerical weather forecasts are commonly used for predicting solar energy for long-term horizons up to a few days, with a typical temporal scale of 1 hour and a spatial scale of approximately 10 km. For shorter term horizons, satellite images offer forecasts up to a few hours, at a 15 min temporal granularity and a 1 to 5km spatial scale. However the spatial and temporal granularity of these two techniques are too coarse to precisely forecast the photovoltaic (PV) energy production of a given plant for very short horizons (< 20min).

To this end, images of the sky from ground-based fisheye cameras have been increasingly investigated in recent years [87, 43, 42, 164, 223]. Coupled with ground truth solar irradiance measurements from pyranometers, fisheye images offer an hemispheric view of the sky enabling to anticipate the evolution of the cloud cover responsible for the electric production variations. A database of several million annotated fisheye images has been collected by EDF R&D. Estimating the irradiance corresponding to a given fisheye image is a favorable perception task for the application of deep learning. We have confirmed at the beginning of this thesis [136] that deep learning indeed provides a large improvement gap over traditional machine learning methods for this estimation task.

[1]For example, the two workshops "Machine learning and the physical sciences" and "Tackling climate change with machine learning" at NeurIPS 2019 gathered together more than 200 papers, and even more at NeurIPS 2020.

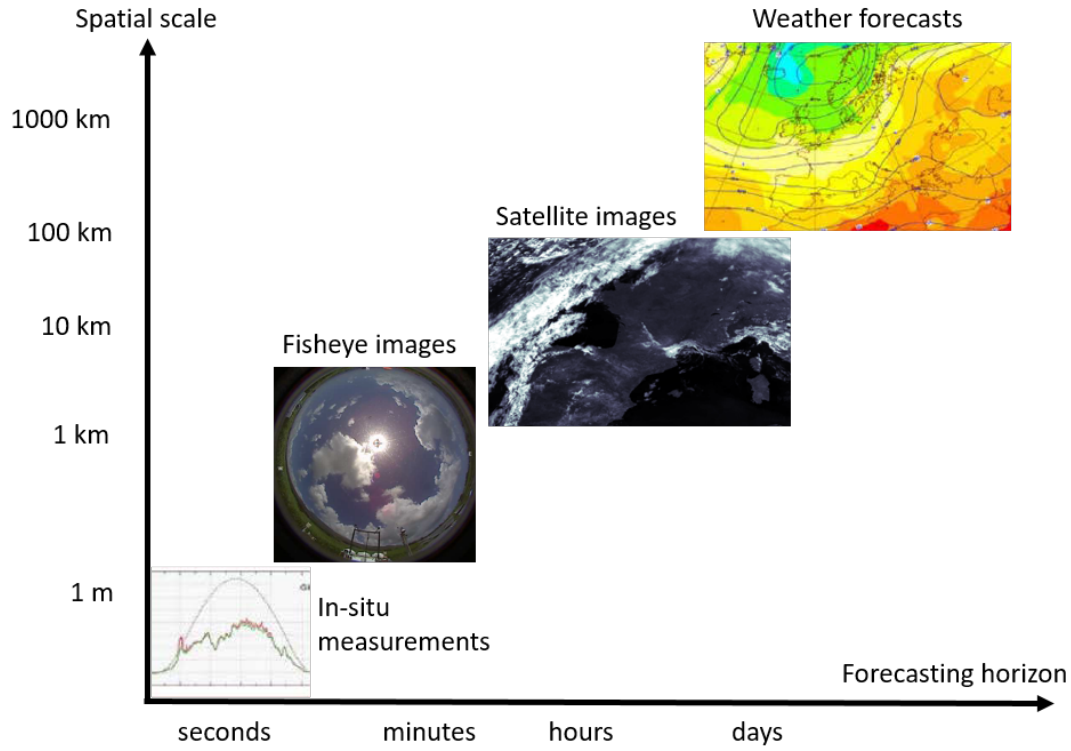


Figure 1.4: The different data sources for forecasting solar energy.

On the contrary, predicting future fisheye images for anticipating the PV production is a much more challenging extrapolation task: clouds are deformable objects with complex stochastic behaviour (that can appear or evaporate), several layers with different speeds and directions may be simultaneously present, and the fisheye camera distortion exacerbates the difficulty. In this context, even recent state-of-the-art deep learning algorihms struggle to properly extrapolate the cloud motion. We describe this use-case with more details in Chapter 8.

1.2 Scientific challenges

We present here the main scientific challenges, highlighted by our industrial application, that we address in this thesis.

1.2.1 Multistep forecasting of non-stationary dynamics

We address the problem of forecasting complex dynamical systems with non-stationary dynamics, i.e. with possible sharp variations. We are interested in describing the distribution of possible futures with a small set of predicted trajectories. In this context, pure data-driven methods are still limited. Paletta *et al.* [184] compared the performances of mainstream convolutional and recurrent neural networks for solar irradiance forecasting at a 10 minutes horizon. They show (see Figure 1.5) that Deep Learning (DL) predictions struggle to match the ground truth (black curve). Two main drawbacks can

be observed: (1) DL predictions smoothen the shape of the sharp drop of solar irradiance in B, and (2) the predictions are late, for example do not anticipate the drop in B[2].

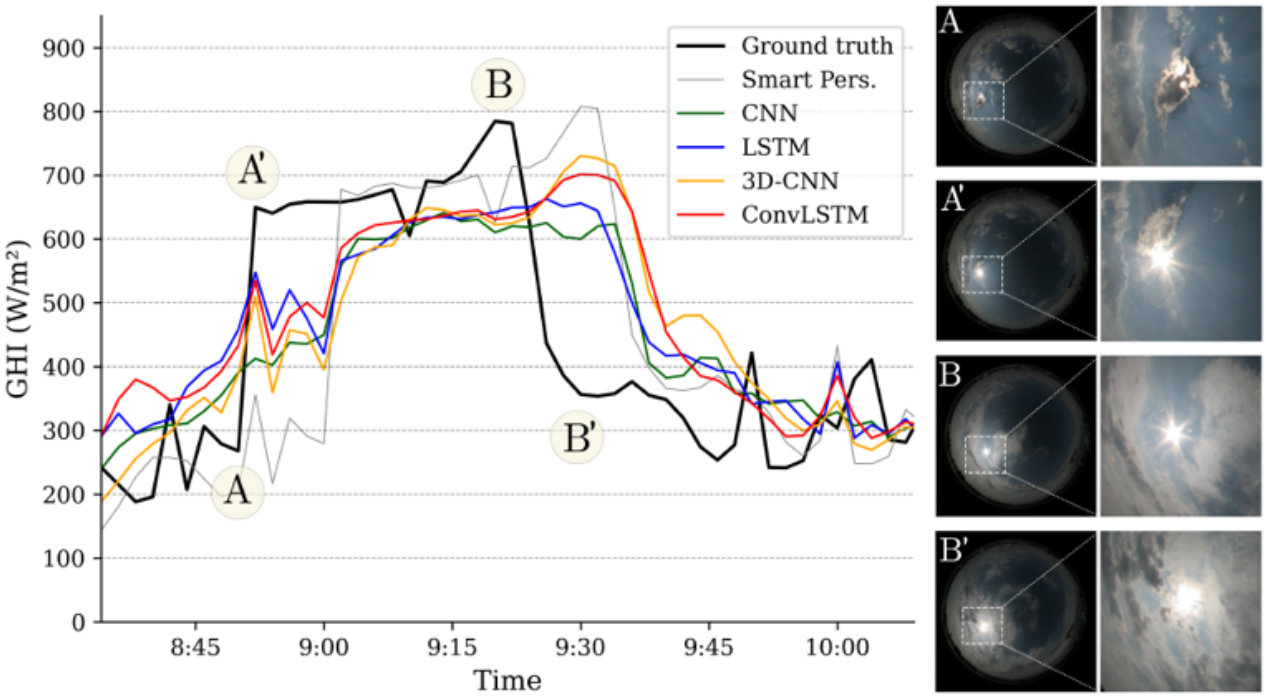


Figure 1.5: Limitations of standard Deep Learning models for 10-min ahead solar irradiance forecasting with fisheye images. Prior-agnostic Deep Learning models trained with the mean squared error do not capture the correct shape of the ground truth nor its exact temporal localization (they are temporally aligned with the smart persistence). Figure taken from Paletta *et al.* [184].

This solar energy forecasting problem illustrates a non-stationary forecasting context, with possible abrupt variations that need to be anticipated on time. This also occurs in many other important applications, e.g. predicting future traffic flows, stocks markets, *etc.* Traditional time series forecasting methods, often relying on stationarity assumptions, are not adapted for this context, and pure data-driven models struggle as well. One of the reasons is the mismatch existing between the evaluation metrics typically used to assess predictions in practice (that take into account shape and temporal errors) and the dominantly used training loss for deep models (the mean squared error).

The main scientific challenges raised by this use-case are the following:

- How to design differentiable metrics for assessing the correctness of shape and the temporal localization of future trajectories?
- How to efficiently describe the uncertainty by providing to the decision makers a small set of possible scenarios reflecting the shape and temporal diversity of future trajectories? In particular, how to structure the diversity of future trajectories according to shape and temporal criteria?

[2]Predictions are temporally aligned with the smart persistence, which corresponds to copying the current value for the future time horizon.

1.2.2 Exploiting incomplete prior physical knowledge in machine learning models

The majority of existing works for combining machine learning and physics assume a *complete* physical knowledge about the system in the training process [58, 201]. In contrast to this mainstream direction, we investigate in this thesis how to leverage *incomplete* physical models, i.e. models that are insufficient for totally describing the dynamics. We have seen that physical models are coarse representations of the reality in many situations, e.g. in physics, climate, robotics, finance, *etc.*

In the solar forecasting energy example, the dynamics of clouds can be described from fluid mechanics principles. However, an exhaustive physical description is mainly out of reach since the dynamics of atmosphere is governed by many complex and interacting physical phenomena (e.g. formation, evaporation of clouds, turbulence). Moreover, even a complete physical model becomes insufficient in case of missing input information, i.e. when the true state of the system (appearing in the dynamical equations) is not fully observed. In our case, we do not have a full observation about the state of the atmosphere above the PV station: we only dispose of fisheye images and we do not use information about the wind speed, the altitude of clouds and we cannot resolve if there exists several cloud layers that mask one another.

Another exacerbating difficulty is the *non-observability of the prior dynamical model*, i.e. when the physical model does not apply directly in the input space. For example common laws of motion for tracking clouds in fisheye images, e.g. a simple advection model, suppose that the clouds have been correctly identified and segmented and that a linear translation of clouds translates in a linear translation in the image, which is not the case because of the circular distortion of the fisheye objective.

So far, exploiting incomplete physical models has been explored by very few works [153, 216, 168]. This problem poses many technical challenges from several points of view:

- Neural network architecture: how to design deep architectures with hard or soft physical constraints?
- Training: how to efficiently train these models? From a theoretical point of view, can we provide guarantees on the quality of the ML/MB decomposition (existence, uniqueness)?

1.3 Contributions and outline

In this thesis, we address the two aforementioned scientific challenges for spatio-temporal forecasting. For multistep and non-stationary time series forecasting in deterministic and probabilistic contexts, we propose to incorporate differentiable shape and temporal features in the training scheme of deep forecasting models (part I of the thesis). For exploiting physical knowledge in deep architectures in incomplete-knowledge settings, we introduce a disentangling architecture and explore the theoretical properties of the resulting ML/MB decomposition (thesis part II). Finally, we apply our proposed ideas to the solar irradiance forecasting problem (thesis part III).

Part I: Differentiable shape and time criteria for deterministic and probabilistic forecasting

In non-stationary contexts occurring in many industrial applications, current deep learning forecasting methods are often inadequate to properly predict sharp variations. The literature is mainly

focused on new neural network architectures to improve forecasts. In contrast, the choice of the training loss function is rarely questioned. The large majority of methods are trained with the proxy Mean Squared Error (MSE) or variants that lead to non-sharp predictions. Besides, current state-of-the-art probabilistic forecasting methods are also ill-adapted for representing the shape and temporal variability of future scenarios. In this part, we propose to design training objectives that account for the shape and temporal localization of predictions.

Our contributions to tackle the first scientific challenge are the following:

- For training deep forecasting models, we introduce in Chapter 3 differentiable shape and temporal criteria inspired by evaluation metrics commonly used in applications. We propose an unifying view of these criteria both in terms of dissimilarities (loss functions) and similarities (positive semi-definite kernels). We insist on their efficient computation and differentiability, which allows to use them in deep learning pipelines.
- For deterministic forecasting, we introduce in Chapter 4 the DILATE training loss function that combines a shape and a temporal dissimilarity to accurately predict sharp events with precise temporal localization. We show that training with DILATE loss instead of the MSE leads to better results at test time on several non-stationary benchmarks for generic and state-of-the-art architectures.
- For probabilistic forecasting, we present in Chapter 5 the STRIPE model that provides a set of diverse and accurate possible future trajectories. The diversity is structured with shape and temporal positive semi-definite kernels embedded in a determinantal point process (DPP) mechanism. We show that our method leads to predictions with a better quality/diversity tradeoff than competing diversifying mechanisms.

Part II: Physically-informed forecasting with incomplete knowledge

To advance towards the exploitation of incomplete physical knowledge in deep forecasting models, we first introduce in this part a new ML/MB deep architecture dedicated to video prediction, for which the physical laws are often not directly applicable at the pixel level. We further delve deeper into the ML/MB decomposition and we propose a new learning framework with uniqueness guarantees.

Our contributions to tackle the second scientific challenge are the following:

- In Chapter 6, we propose a new deep architecture called PhyDNet dedicated to video prediction in non-observable prior contexts. PhyDNet learns physical dynamics parameterized by a general class of PDEs. Since the physical laws may not directly apply at the pixel level in videos, we complement the physical model with a data-driven model in charge of learning the residual information necessary for accurate prediction, such as appearance, texture, details. We show that PhyDNet reaches very good performances on several video prediction benchmarks, from a strong (linear translation for the Moving MNist dataset) to a weak prior physical knowledge (modelling general human motion for Human 3.6 dataset).
- In Chapter 7, we concentrate on the ML/MB decomposition problem and the optimal cooperation between physical and data-driven models. We introduce a principled learning framework, called APHYNITY, for forecasting complex physical systems with incomplete knowledge. Inspired

by the least-action principle, APHYNITY minimizes the norm of the data-driven complement under the constraint of perfect prediction of the augmented model, which leads to a unique decomposition under mild assumptions (Chebychev set). We show on several challenging physical dynamics that APHYNITY ensures better forecasting and parameter identification performances than MB or ML models alone, and that competing ML/MB hybrid methods.

Part III: Application to solar irradiance forecasting

Finally, we apply the methodological contributions of this thesis to the solar irradiance forecasting problem at EDF.

- In Chapter 8, we present the industrial solar irradiance forecasting problem in more details and review the existing literature for solving it. We also propose a first deep learning model for estimating and forecasting solar irradiance.
- In Chapter 9, we apply the methodological contributions of this thesis to this problem. We propose an adaptation of the introduced PhyDNet architecture to perform physically-constrained prediction. We also evaluate the DILATE loss and the APHYNITY framework on this problem and discuss future improvement directions.

Before delving in the core of the thesis, we present in Chapter 2 on overview of the basics of machine learning and the related works on spatio-temporal forecasting and physically-constrained machine learning. Finally, in Chapter 10, we summarize our work and propose appealing perspectives for future works.

This thesis is based on the following list of publications:

Publication	Chapter
Vincent Le Guen and Nicolas Thome. "Deep Time Series Forecasting with Shape and Temporal Criteria". IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.	3
Vincent Le Guen and Nicolas Thome. "Shape and Time Distortion Loss for Training Deep Time Series Forecasting Models". In Advances in Neural Information Processing Systems (NeurIPS 2019).	4
Vincent Le Guen and Nicolas Thome. "Probabilistic Time Series Forecasting with Shape and Temporal Diversity". In Advances in Neural Information Processing Systems (NeurIPS 2020).	5
Vincent Le Guen and Nicolas Thome. "Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020).	6
Yuan Yin*, Vincent Le Guen*, Jeremie Dona*, Ibrahim Ayed*, Emmanuel de Bézenac*, Nicolas Thome and Patrick Gallinari. "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting", In International Conference on Learning Representations (ICLR 2021, oral presentation), Journal of Statistical Mechanics: Theory and Experiments (JSTAT 2021).	7
Vincent Le Guen and Nicolas Thome. "Prévision de l'irradiance solaire par réseaux de neurones profonds à l'aide de caméras au sol". In: GRETSI 2019.	8
Vincent Le Guen and Nicolas Thome. "A Deep Physical Model for Solar Irradiance Forecasting With Fisheye Images". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops 2020 (OmniCV 2020 workshop)	9

Chapter 2

State-of-the-art on spatio-temporal forecasting

Content

Chapter abstract

In this Chapter, we first present the basic concepts of machine learning and deep learning targeted to the problem of time series forecasting (Section 2.1). Then we present an historical view of spatio-temporal forecasting, from the traditional to the more recent deep approaches for deterministic and probabilistic forecasting (Section 2.2). We make a focus on the training and testing metrics, on the specific challenges of video prediction and on the question of diversity in probabilistic forecasting. Finally, we introduce the concepts of physics-based machine learning (Section 2.3). We comment the existing strategies for regularizing machine learning with physical knowledge, at the training loss and at the architectural level. We also review the question of physical system identification with machine learning and discuss the few recent works for augmenting incomplete physical models.

2.1 Machine Learning

2.1.1 Background

Deep Learning belongs to the broader category of statistical machine learning. In the *supervised learning* context, the goal is to estimate the optimal mapping $Y = f(X)$ between inputs X and outputs Y, given a training dataset of N labelled examples $\{(X_i, Y_i)\}_{i=1}^{N} \in (\mathcal{X} \times \mathcal{Y})^N$. The inputs are represented by the attribute (or feature) vectors $X_i \in \mathbb{R}^d$, and the target Y_i can be a categorical variable $Y_i \in \{0, 1, ..., K\}$ for classification tasks or a real variable $Y_i \in \mathbb{R}^k$ for regression tasks. We illustrate in Figure 2.1 the supervised machine learning framework in the case of time series forecasting.

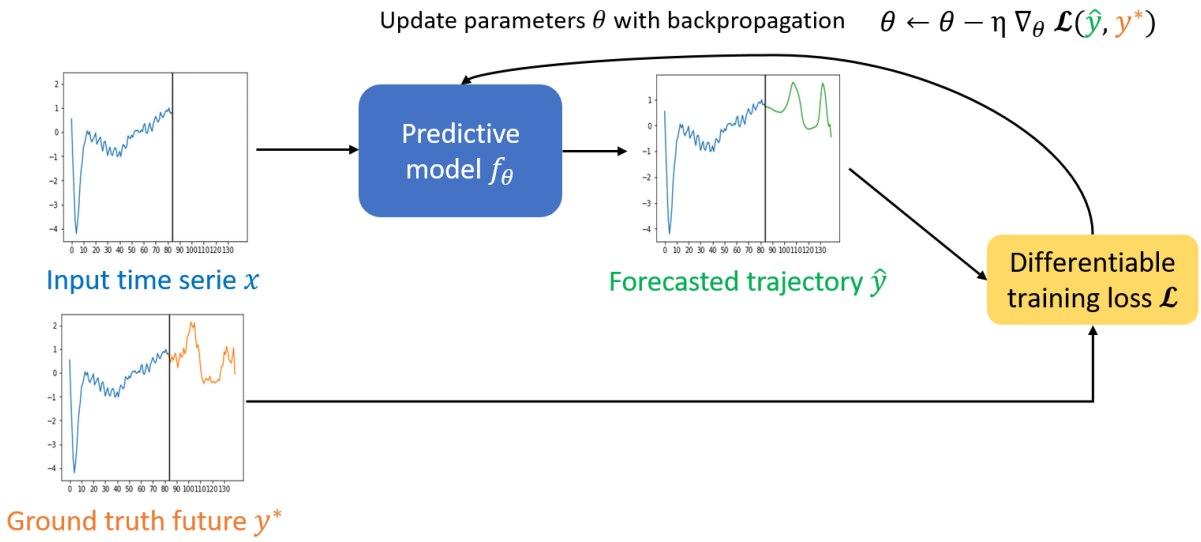


Figure 2.1: Supervised machine learning framework for time series forecasting.

Learning framework The classifier or regressor function f is optimized over an hypothesis class $\mathcal{H}$ of functions. Examples of classes include the linear models, the kernel methods, or the neural networks. This class should be carefully chosen for the task, guided by the bias-variance tradeoff [16]. The class $\mathcal{H}$ should be sufficiently expressive for modelling the solution of the problem; on the contrary, a too large model capacity reduces the bias but favors the overfitting phenomenon on the training set.

Once the class $\mathcal{H}$ is defined, we want to select the function f that best fits the training data, while generalizing correctly to unseen input data coming from the same distribution. Training the model consists in minimizing the risk $R(f)$ that measures the disagreement between the predictions and the ground truth labels with a loss function $\ell : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$:

$$R(f) := \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ \ell(f(X), Y) \tag{2.1}$$

$$f^* = \text{argmin}_{f\in\mathcal{H}}\ R(f). \tag{2.2}$$

In practice, the joint distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ is unknown, therefore we minimize the empirical risk

defined with the training samples:

$$R_n(f) := \frac{1}{n}\sum_{i=1}^{N} \ell(f(X_i), Y_i). \tag{2.3}$$

Training loss functions In the context of binary classification ($\mathcal{Y} = \{0, 1\}$), a common loss function is the binary cross-entropy:

$$\ell(f(X), Y) = -[Y \log f(X) + (1 - Y) \log(1 - f(X))]. \tag{2.4}$$

For regression problems such that found in time series or video prediction, the most common loss function is the mean squared error (MSE), corresponding to the L2 loss averaged over input-output pairs:

$$\ell(f(X), Y) = \|f(X) - Y\|_2^2. \tag{2.5}$$

Monostep vs. multistep forecasts For time series forecasting, the loss function ℓ can either applied to compare monostep or multistep forecasts. Monostep forecasting methods compute a one-step ahead prediction $\hat{\mathbf{y}}_{T+1}$ given past values $(\mathbf{y}_1, \cdots, \mathbf{y}_T)$, which is compared to the ground truth future $\mathbf{y}^*_{T+1}$: $\ell(\hat{\mathbf{y}}_{T+1}, \mathbf{y}^*_{T+1})$. In contrast, multistep forecasts compute the loss on multiple predicted timesteps: $\ell((\hat{\mathbf{y}}_t)_{T+1:T+H}, (\mathbf{y}^*)_{T+1:T+H})$. The mean squared error (MSE), dominantly used in applications, is *separable*, i.e. the multistep loss is the sum of the loss for all individual timesteps. In this thesis, we study dedicated loss functions for multistep forecasting, that are non separable, for explicitly imposing a desired behaviour based on the whole predicted dynamic's trajectory.

Regularization Machine learning models are optimized to predict the labels of the training set. However, a model that perfectly predicts those labels does not necessarily generalize well to unseen data. With high capacity models such as deep neural networks, the risk is to learn the training set by heart and represent a too complex function; this phenomemon is called *overfitting*.

To overcome this issue, a common strategy is to add a *regularization* term Ω to the training objective for penalizing the complexity of the model:

$$\min_{f \in \mathcal{H}} \; R_n(f) + \Omega(f). \tag{2.6}$$

From a Bayesian point of view, many regularizers correspond to certain prior distributions over the model parameters. The most popular choices include the L2 and L1 weight normalization. As we will dicuss in Section 2.3, normalization is a possible way to leverage physical priors in a model.

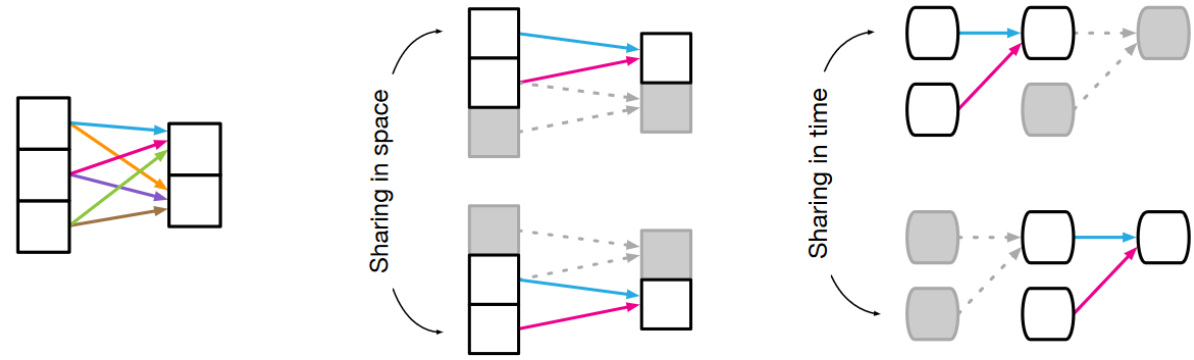
2.1.2 Deep neural networks

Neural networks are based on the simple artificial neuron modelling proposed by MCulloch and Pitts [167] and have been explored from the 1980's [142]. Standard feedforward neural networks are composed of a succession of mathematical functions called *layers* that progressively transform the inputs X to the outputs Y through a sequence of intermediate representations $\mathbf{h}_l$ called *hidden states*. A typical *dense* (or *fully-connected*) layer consists in a linear combination of the inputs followed

by a nonlinear activation ϕ: $\mathbf{h}_{l+1} = \phi(\mathbf{W}_l \mathbf{h}_l + \mathbf{b}_l)$ for the l^{th} layer. The typical nonlinearities are traditionally the sigmoid, hyperbolic tangent or the Rectified Linear Unit (ReLU) $x \mapsto \max(0, x)$.

Neural networks are trained using gradient descent algorithms, such as the basic Stochastic Gradient Descent (SGD) [20] or variants with momentum like AdaDelta [68] or Adam [118]. The gradient of the loss with respect to the model's parameters is computed by the backpropagation method [142]. Thus all applied operations in the model should be differentiable, in particular the loss function. We will see in this thesis that the choice of a differentiable loss function is a key aspect for imposing a desired behaviour.

Deep Learning has become popular since the victory of the AlexNet model [127] at the ImageNet competition in 2012. The main revolution of Deep Learning relies in the depth of the neural networks. By stacking many layers, the network progressively learns more and more complex feature representations of the input, from the low-level concepts (such as color or contours) to the most semantics concepts (such as the recognition of a particular object) necessary for image classification.



(a) Fully connected (b) Convolutional (c) Recurrent

Figure 2.2: Common layers used in deep learning models. Shared parameters are shown with the same color. Figure taken from Battaglia *et al.* [10].

The choice of the neural network architecture is a critical aspect for solving a task. We illustrate in Figure 2.2 the three main kinds of layers. The Multi-Layer Perceptron (MLP) [212], only composed of fully-connected layers, is the most generic architecture but at the expense of a number of parameters exponentially growing with the number of layers, making it not amenable for many applications. Other architectures encode specific inductive biases on data. For example, convolutional neural networks [142] encodes spatial equivariance, i.e. the response of a classifier should be independent to the particular location of objects in the image, by sharing a convolutional filter for all spatial positions. Likewise, recurrent neural networks encode translation equivariance for processing sequential data by reusing the same weight in time. More recent architecture also encode other kinds of inductive biases: graph neural networks [9] encode permutation invariance among a set of items, and the recent Transformer architecture [255] implements an attention mechanism over neighbouring positions.

When investigating deeper and deeper architectures, researchers have been faced with training issues like the vanishing gradient problem, i.e. the gradient of the loss can become very small after backpropagating through a large number of layers. To overcome this problem, He *et al.* [101] has proposed the *residual neural networks* (ResNets) by adding skip connections between a block of standard

layers:

$$\mathbf{x}_{l+1} = \mathcal{F}(\mathbf{x}_l) + \mathbf{x}_l, \tag{2.7}$$

where $\mathbf{x}_l$ is the hidden state after the l^{th} block and $\mathbf{F}$ denotes a nonlinear function (e.g. a series of convolutions and nonlinear activations). These "identity shortcuts" allow a direct flow of the gradient and have significantly improved the training of very deep networks, leading to new state-of-the-art performances on ImageNet. Pursuing this idea, the *densely connected networks* (DenseNets) of Huang *et al.* [105], connecting all layers together within a block with skip connections, have further improved the performances.

Difference between traditional ML and DL The main differences between traditional Machine Learning (ML) and Deep Learning (DL) are illustrated in Figure 2.3 for the case of solar irradiance forecasting with fisheye images. The traditional ML pipeline (from the existing method at EDF [87]) is composed of several steps with manual intervention: camera calibration for compensating the fisheye distortion, projection of the input images on a plane at a given altitude, optical flow estimation, image warping for computing the future frame, future image segmentation with handcrafted features and thresholds, and finally prediction of the future irradiance with a traditional regressor (e.g. linear regression). Many of these steps require expert manual intervention. On the other side, the Deep Learning approach directly learns the image to irradiance mapping on raw fisheye images and automatically derives the appropriate intermediate concepts.

In fact, the difficulty of the task has shifted from the handcrafted feature engineering of traditional ML methods to the manual neural network architecture design of DL that encodes appropriate inductive biases or behaviours.

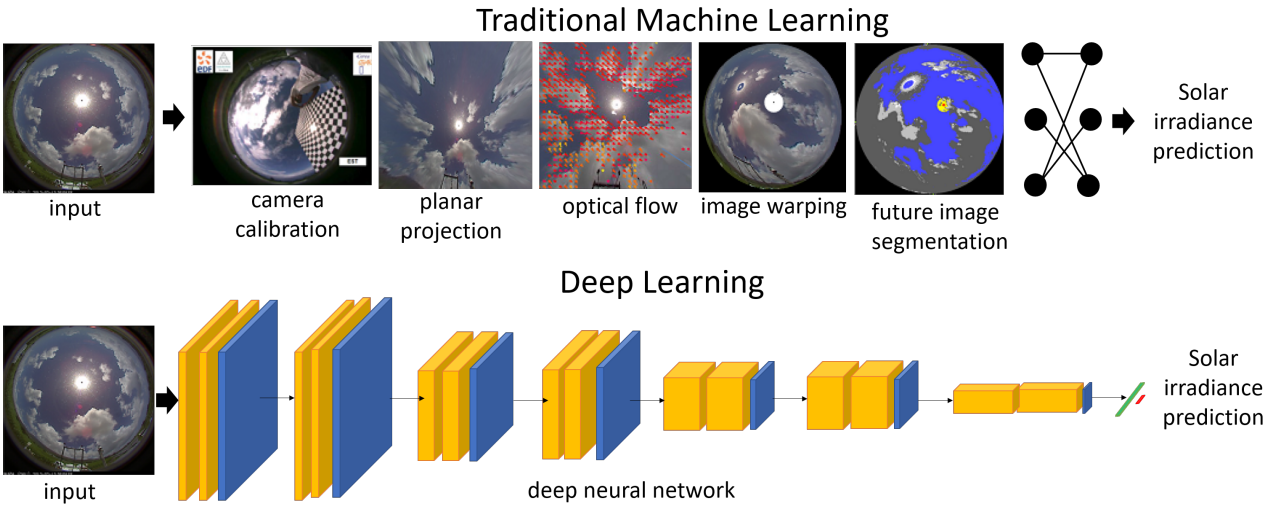


Figure 2.3: Traditional Machine Learning vs. Deep Learning for forecasting solar irradiance with fisheye images.

2.2 Spatio-temporal forecasting

In this Section, we review the main existing machine learning approaches for spatio-temporal forecasting, from the traditional statistical time series forecasting to the most recent deep learning methods.

2.2.1 Context and notations

As discussed in Introduction (Chapter 1), we are interested in forecasting spatio-temporal processes driven by some underlying physical phenomenon. We consider dynamical systems formalized through a differential equation of the form:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = F(X_t). \tag{2.8}$$

The *state* of the system X_t represent the variables whose knowledge at time t_0 is sufficient, in combination with the evolution function F, for describing the phenomenon for each time $t > t_0$. The state X_t can be either be parameterized by:

- a d-dimensional vector, i.e. we have $X_t \in \mathbb{R}^d$ for every t. In that case, equation 2.8 is an *ordinary differential equation* (ODE);
- a d-dimensional vector field over a spatial domain $\Omega \subset \mathbb{R}^k$, with $k \in \{2, 3\}$, i.e. $X_t(x) \in \mathbb{R}^d$ for every $(t, x) \in [0, T] \times \Omega$. If the description in Eq 2.8 involves spatial derivatives of the state, it corresponds to a *partial differential equation* (PDE).

Many phenomena occurring in physics, biology, computer vision, finance follow a general equation of the form 2.8.

To solve the differential equation 2.8 numerically, the most common approach is to discretize the phenomenon into a sequence $(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T)$ and approximate the time derivative with finite differences. The simplest numerical scheme is the forward Euler method:

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \; F(\mathbf{x}_n), \tag{2.9}$$

where Δt is a fixed step size. We will see that this approximation scheme has strong connections with residual neural networks (Section 2.3.1). More complex numerical schemes exist with lower truncation errors, e.g. Runge-Kutta [24].

For predicting a dynamical system of the form 2.8, two main modelling approaches exist:

- parameterize the relationship between future time steps and context time steps: $(\hat{\mathbf{y}}_{T+1}, \ldots, \hat{\mathbf{y}}_{T+H}) = g_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_T)$ with parameters θ. The function g_θ can represent a traditional time series forecasting model like an autoregressive model [21] or a deep neural network.
- parameterize the derivative function F_θ and integrate the ODE/PDE with a numerical solver. This is the typical case of numerical simulation with a physical model F_θ. The function F_θ can also be a deep neural network approximating the dynamics, as done by the Neural ODEs [33] presented in Section 2.3.1.

2.2.2 Model-Based forecasting methods

As discussed in Chapter 1, the traditional modelling paradigm in physics is to derive analytical laws of motion from first principles and integrate the equations with numerical simulation. These models are often expressed as ordinary or partial differential equations (ODEs/PDEs). This arises in a multitude of scientific fields, such as Newtonian mechanics, fluid dynamics or quantum mechanics. For example, we will consider in this thesis the wave equations:

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

where k is the damping coefficient and c the celerity of the wave.

For time series forecasting, traditional Model-Based methods rely on linear state space models (SSMs) [70, 106], which provide a principled framework for modelling known temporal patterns. SSMs include the popular integrated autoregressive moving-average model (ARIMA) and Exponential Smoothing. SSMs assume linear dynamics with structural components (e.g. level, trend, seasonality), which makes forecasting robust and interpretable. However, the model selection procedure can be tedious and these methods often exploit strong statistical (e.g. i.i.d. additive Gaussian noise) and structural assumptions on data (e.g. stationarity or stationarity after differentiation) that are not satisfied for many real-world time series that can present abrupt changes of distribution. Moreover, SSMs are fitted independently on each time series, and thus cannot learn patterns between sets of similar series.

Regarding video prediction, traditional methods focus on predicting the motion field with optical flow, rather than predicting future frames at the pixels level. The seminal works of Lucas-Kanade [160] and Horn-Schunk [103] rely on the brightness consistency constraint, which assumes that the intensity value of a pixel remains constant between two frames. In its linearized form, this constraint can be expressed as a PDE:

$$\frac{\partial I}{\partial t}(t, \mathbf{x}) = -w(t, \mathbf{x}) \cdot \nabla I(t, \mathbf{x}). \tag{2.10}$$

Again, this PDE corresponds to an incomplete model, since the brightness constancy assumption is violated in many situations, e.g. in presence of occlusions, illumination changes, specular reflexions.

2.2.3 Deep learning forecasting methods

Artificial neural networks were first explored in the 1990's for time series forecasting with Multi-Layer Perceptrons (MLPs) [29, 143, 242] and Recurrent Neural Networks (RNNs) [48, 128]. At that time, most of these architectures were limited to a single hidden layer and trained with one-step targets, restricting their applicability to simple problems.

With the advances in computer hardware and modern training techniques of the deep learning era, neural networks have become appealing for time series forecasting due to their automatic feature extraction, the ability to capture complex nonlinear temporal patterns and the ease to incorporate exogenous variables.

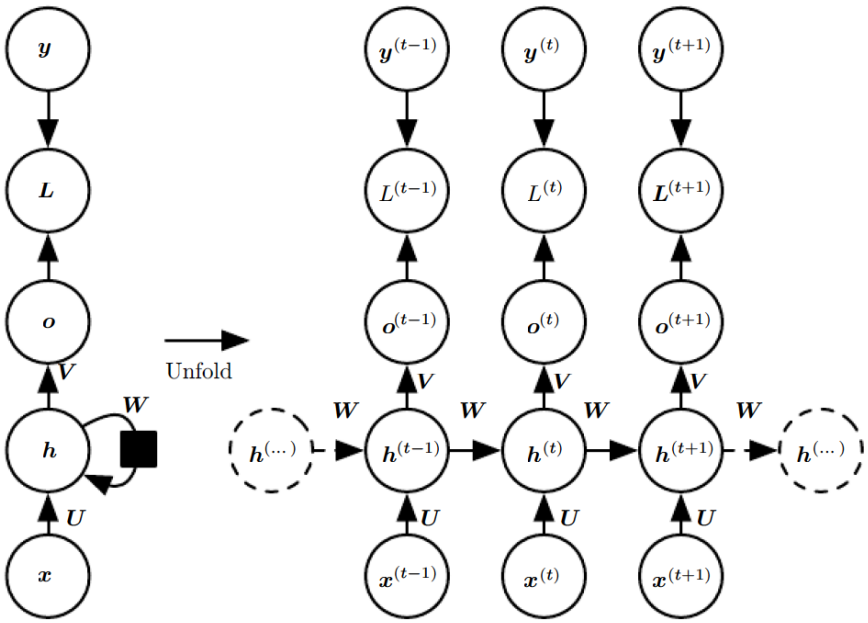


Figure 2.4: A recurrent neural network. Figure taken from Goodfellow [94].

2.2.3.1 Recurrent Neural Networks (RNNs)

RNNs denote a family of architectures dedicated to handling sequential data such as text, speech or time series. Illustrated in Figure 2.4, RNNs implement a discrete time dynamical system, where a hidden variable $\mathbf{h}_t \in \mathbb{R}^d$, serving as a memory of the system, is recurrently updated across time. A basic RNN formulation can be written as:

$$\mathbf{h}_t = F(\mathbf{W}\ \mathbf{h}_{t-1} + \mathbf{U}\ \mathbf{x}_t + \mathbf{b}) \tag{2.11}$$

$$\mathbf{o}_t = \mathbf{V}\ \mathbf{h}_t, \tag{2.12}$$

where $\mathbf{U}$ and $\mathbf{W}$ are weight matrices, $\mathbf{b}$ is a bias and F an activation function (e.g. tanh). The output $\mathbf{o}_t$ at time t, obtained by a projection of the latent state with a weight matrix $\mathbf{V}$, is compared to the ground truth target $\mathbf{y}_t$ with a loss function L. Crucially, the weights of the RNN are identical for all timesteps (as shown in Figure 2.4). Contrary to more general MLPs, weight sharing in RNNs enables to encode time equivariance and to process sequences of arbitrary lengths. Deep recurrent neural networks can be build by stacking RNN cells.

RNNs are trained by backpropagation through time [175], i.e. by propagating the gradient of the loss function in the unfolded computational graph (see Figure 2.4). A major drawback of the vanilla formulation in Eq 2.11 is that the vanshing / exploding gradients when processing long sequences [188]. It prevents the network from memorizing long-term information in the current latent state. To address this limitation and model long-term dependencies, Hochreiter *et al.* [102] introduced the Long-Short Term Memory (LSTM) networks which have an additional memory cell $\mathbf{c}_t$ controlled by a learned input

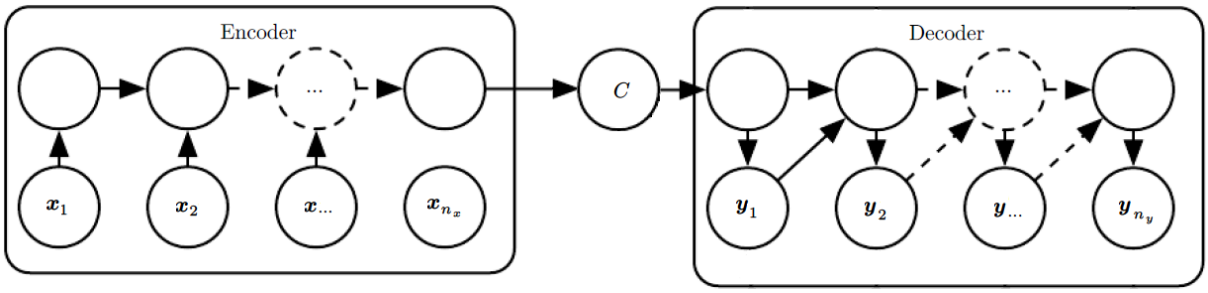


Figure 2.5: Sequence To Sequence model. Figure adapted from Goodfellow [94].

gate $\mathbf{i}_t$ and forget gate $\mathbf{f}_t$:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{ih}\ \mathbf{h}_{t-1} + \mathbf{W}_{ix}\ \mathbf{x}_t + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_{fh}\ \mathbf{h}_{t-1} + \mathbf{W}_{fx}\ \mathbf{x}_t + \mathbf{b}_f) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_{gh}\ \mathbf{h}_{t-1} + \mathbf{W}_{gx}\ \mathbf{x}_t + \mathbf{b}_g) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_{oh}\ \mathbf{h}_{t-1} + \mathbf{W}_{ox}\ \mathbf{x}_t + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t).
\end{aligned}$$

LSTM networks and their variants such as the Gated Recurrent Unit (GRU) [39], have become a reference for many sequential tasks. Shi *et al.* [281] proposed the ConvLSTM adaptation for video prediction, by replacing all the full-connected operations of the LSTM by convolutions. The ConvLSTM was adopted in many subsequent studies [77, 112, 282] and is at the basis of the most recent video prediction algorithms such as PredRNN [268, 266], Memory in Memory [269] or MotionRNN [277].

2.2.3.2 Sequence To Sequence models

For mapping a variable-length sequence to another variable-length sequence, Cho *et al.* [39] and Sutskever *et al.* [240] proposed the Sequence To sequence (Seq2Seq) architecture. The input sequence $(\mathbf{x}_1, \cdots, \mathbf{x}_{n_x})$ is processed by an encoder RNN that provides a fixed-size context vector C summarizing the sequence, typically defined as the last hidden state of the RNN. This context vector is used for initializing the decoder which is another RNN producing the predictions $(\mathbf{y}_1, \cdots, \mathbf{y}_{n_y})$ one step at a time. In a Seq2Seq model, both RNNs are trained jointly to maximize the likelihood $p(\mathbf{y}_1, \cdots, \mathbf{y}_{n_y} | \mathbf{x}_1, \cdots, \mathbf{x}_{n_x})$ averaged over all the input/output sequences of the training set.

When generating predictions, the RNN decoder is rolled forwards by recursively feeding back its own predictions as inputs for the next timesteps. Seq2Seq models can be trained with *teacher forcing*, consisting in feeding the true targets as inputs to the RNN (that are known at training time) instead of the prediction from the last timestep. A popular curriculum often used in practice to mitigate the train/test discrepancy is *scheduled sampling* [14] that randomly chooses to use true values or model predictions as inputs, with a sampling probability to use model predictions increasing over time to gradually converge towards test-time conditions.

Seq2Seq architectures with RNNs are at the basis of many successful models [80, 202, 131]. Salinas *et al.* [219] proposed DeepAR, a Seq2Seq model which estimates the parameters of a Gaussian distribution for the next timestep. Rangapuram *et al.* [202] revisit the traditional state space models (SSMs) by

parameterizing them with deep recurrent networks. To limit error accumulation due to autoregressive predictions, some models directly predict all future values at once, often with a MLP decoder [275].

RNN forecasting can be improved with the attention mechanism, introduced by Bahdanau *et al.* [7] for machine translation [198, 133, 76]. Attention consists in learning which part of the input sequence is the most relevant for predicting a given timestep. More precisely, the context vector C is replaced with a combination of the hidden states from past timesteps weighted by their learned attention weights.

2.2.3.3 Beyond recurrent architectures

Training RNNs with backpropagation through time is expensive since it requires sequential operations that cannot be parallelized. Researchers have explored alternative architectures than RNNs. Following the success of the Wavenet model for audio processing [253], temporal convolution networks (TCNs) [19, 36] use causal dilated 1D-convolutions, that exponentially increase the receptive field with additional layers and respect the temporal causality. In addition, TCNs can be easily trained in parallel.

Recently, a line of works has questioned the convolutional or recurrent layers used in most architectures, showing that fully-connected layers arranged in a careful way can outperform other methods. For example, pure attention-based models have revealed better than LSTMs for capturing long-range relationships. The Transformer architecture of Vaswani *et al.* [255], only composed of self-attention and fully-connected layers, avoids the recurrent structure and provides a direct access to any previous timestep. Several works have proposed adaptations of the Transformer for time series forecasting [**?** 300]. In particular, the Informer model of Zhou *et al.* [300] is able to extend the predictions to a long-term horizon with less degradation than competing methods.

Another example is the NBeats forecasting architecture [183] shown in Figure 2.6 that has recently shown state-of-the-art performances for deterministic forecasting. NBeats is composed of stacks of fully-connected layers, each block outputting a forecast for the following block and a backcast that removes the part of the signal that is well-explained by the current block. Partial forecasts from each block are finally combined into the global forecast.

2.2.4 Training and evaluation metrics for time series forecasting

Current research on time series forecasting mainly focuses on new architecture design (the predictive model f_θ in the blue box in Figure 2.1) and the question of the training loss (yellow box in Figure 2.1) is often overlooked. The Mean Squared Error (MSE) in Eq 2.5, Mean Absolute Error (MAE) and its variants (SMAPE, *etc*) are predominantly used as proxies for training models. In practice, forecasts are evaluated with application-specific metrics, often reflecting the shape and temporal localization of future trajectories. However, their non-differentiability makes them unsuitable for training deep models. For characterizing shape, the Dynamic Time Warping (DTW) algorithm [217, 110, 298], originally introduced for speech recognition, computes the similarity between time series after temporal alignment. DTW is particularly popular for time series classification [110] or clustering [31] and has been recently explored for time series forecasting [55]. Another shape metric is the ramp score [79, 252] that assesses the detection of ramping events in wind and solar energy forecasting. Timing errors can be characterized among other ways by the Temporal Distortion Index (TDI) [83, 252], or by computing detection scores (precision, recall, Hausdorff distance) after change point detection [249].

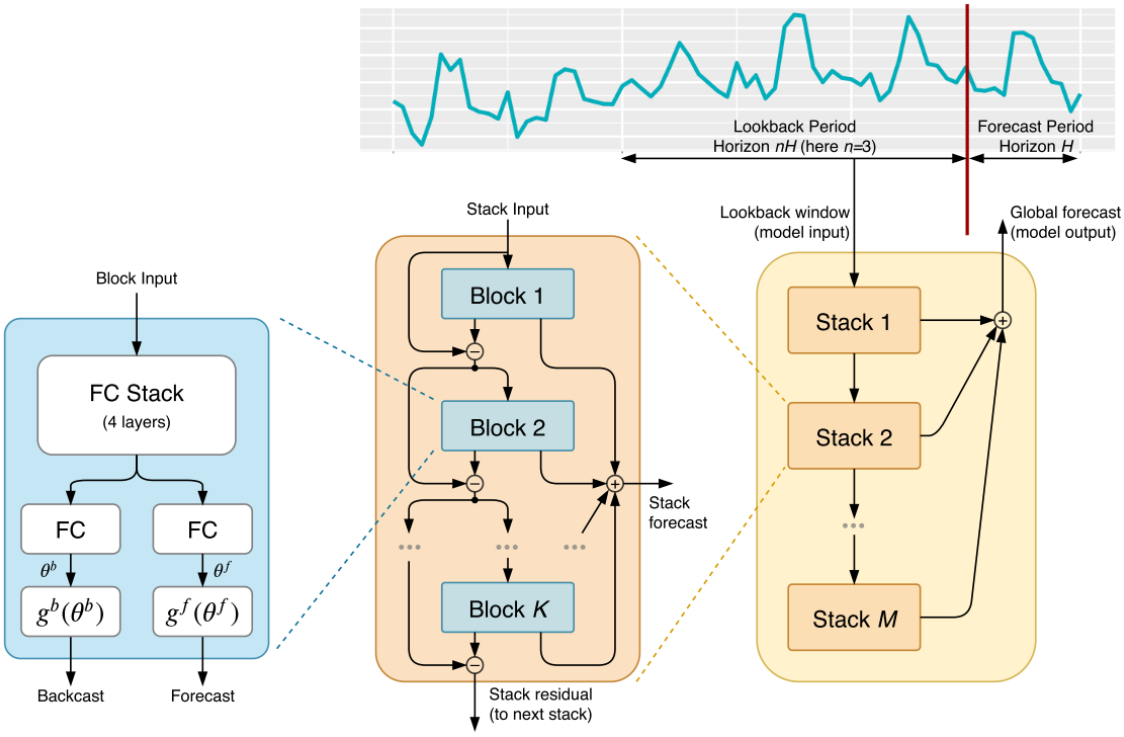


Figure 2.6: The NBeats model for deterministic forecasting [183].

Recently, some attempts have been made to train deep neural networks based on alternatives to MSE, especially based on smooth approximations of DTW [55, 171, 1, 256, 17], in particular the soft DTW [55] that we will detail in Chapter 4.

In this thesis, we intend to bridge the gap between these common evaluation metrics and the training losses used in practice. We explore how to efficiently combine explicit shape and temporal differentiable criteria at training time, regardless of the training architecture. We will review the most related works in more details in Chapter 3.

2.2.5 Particular challenges in video prediction

Videos are a particular form of multivariate time series, and all the time series forecasting methods presented above could in principle be directly applied to videos by forecasting the dynamics of individual pixels. However this approach neglects the keys properties of images: the spatial coherence between neighboring pixels and the semantics of the scene. Specific architectures dedicated to video prediction were explored [268, 266, 269, 267, 277], often based on variants of the seminal ConvLSTM [229].

Moreover, extrapolating high-dimensional signals such as images at the pixel level is extremely challenging. To constrain this generation problem, several methods rather use domain-specific knowledge such as predicting geometric transformations between frames [77, 112, 283], estimating the optical flow [190, 161, 152**?** , 148] or exploiting the semantics of the scene [12]. This is very effective for short-term prediction, but degrades quickly when the video content evolves, where more complex models and memory about dynamics are required.

Disentanglement Another line of work consists in disentangling independent factors of variations in order to apply the prediction model on lower-dimensional representations. The typical decomposition criteria are as content/motion [258, 144] or deterministic/stochastic [60]. We illustrate in Figure 2.7 an example of decomposition from the DPPAE model [104]: the moving objects are extracted and their individual motion estimated separately to provide the final prediction. In specific contexts, the prediction space can be structured using additional information, e.g. with human pose [259, 262] or key points [173].

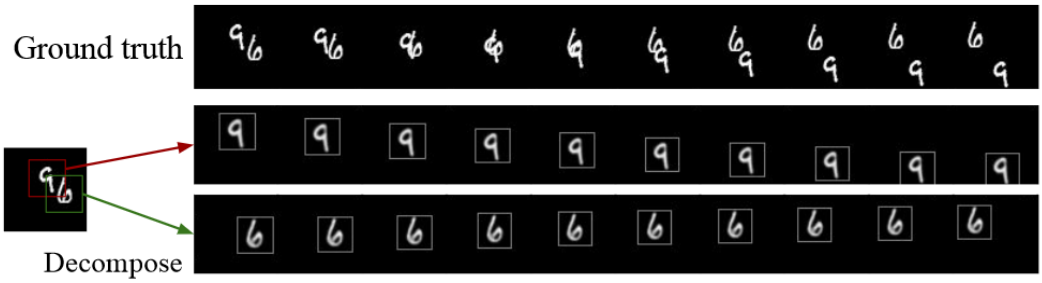


Figure 2.7: Disentanglement approach for video prediction. In this Moving MNIST example, the DPPAE model [104] decomposes the two digits and predicts their dynamics separately.

We provide a more detailed review of existing deep video prediction methods in Chapter 6.

2.2.6 Diversity in probabilistic forecasting

Many critical applications require forecasts associated with uncertainty to make relevant decisions. Probabilistic forecasting consists in estimating the predictive distribution of future values given an input sequence. Two main categories of methods exist for probabilistic forecasting. The first class of methods directly characterizes the predictive distribution. This includes estimating the variance of predictions (e.g. with Monte Carlo dropout [84]), estimating the quantiles [275, 86, 274] or modelling this distribution by a parametric distribution, e.g. a Gaussian for the DeepAR algorithm [219]).

In this thesis, we focus on a second class of probabilistic methods that propose to describe the predictive distribution with a set of plausible scenarios reflecting the uncertainty of future behaviour. This class includes ensemble methods [233] and generative models, which produce diverse forecasts by sampling multiple latent variables from a prior distribution. The most popular generative models are conditional variational autoencoders (cVAEs) [292], conditional generative adversarial networks (cGANs) [124], and normalizing flows [203, 59]). For further diversifying forecasts, several repulsive schemes were studied such as the variety loss [97, 244] that consists in optimizing the best sample, or entropy regularization [63, 263] that encourages a uniform distribution.

However the aforementioned methods are limited for representing the diversity of future behaviour with a limited number of scenarios, as discussed in Chapter 1. Standard generative models sample points belonging to the dominant mode, e.g. by sampling multiple forecasts at test time from a standard Gaussian prior, and do not provide control over the diversity of predictions.

Determinantal Point Processes (DPP) To improve this unstructured mechanism, prior works [292, 293] introduced proposal neural networks for generating the latent variables that are trained with

a diversity objective based on Determinantal Point Processes (DPPs).

DPPs are appealing probabilistic models for describing the diversity of a set of items $\mathcal{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_N\}$. A DPP is a probability distribution over all subsets of $\mathcal{Y}$ that assigns the following probability to a random subset $\mathbf{Y}$:

$$\mathcal{P}_{\mathbf{K}}(\mathbf{Y} = Y) = \frac{\det(\mathbf{K}_Y)}{\sum_{Y' \subseteq \mathcal{Y}} \det(\mathbf{K}'_Y)} = \frac{\det(\mathbf{K}_Y)}{\det(\mathbf{K} + \mathbf{I})}, \tag{2.13}$$

where $\mathbf{K}$ is a positive semi-definite (PSD) kernel and $\mathbf{K}_A$ denotes its restriction to the elements indexed by A.

We illustrate the behaviour of DPPs in Figure 2.8 for sampling random points in the plane. When we draw points randomly according to a uniform distribution, some regions may become more densely populated than other. In contrast, when sampling from a DPP distribution with a Gaussian kernel, points are farther from one another and better spread on to the plane.

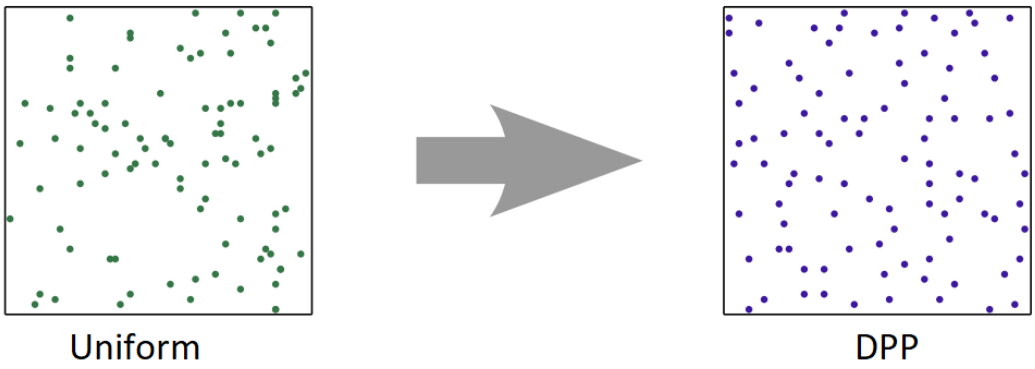


Figure 2.8: Random points sampled in the plane from a uniform distribution vs a determinantal point process (DPP) distribution. Figure taken from Kulesza and Taskar [130].

DPPs offer efficient algorithms for sequentially sampling diverse items or maximizing the diversity of a set with a given sampling budget. Importantly, the choice of the kernel enables to incorporate prior structural knowledge on the targeted diversity. As such, DPPs have been successfully applied in various contexts, e.g. document summarization [92], recommendation systems [90], image generation [71] and diverse trajectory forecasting [292].

In this thesis, we design specific shape and temporal PSD kernels for imposing our structured diversity. We further describe the most related works for probabilistic forecasting in Chapter 5.

2.3 Physics-informed machine learning

As discussed in Chapter 1, pure data-driven machine learning methods struggle to extrapolate complex dynamical systems, and often overfit on the training set. Incorporating prior knowledge about the system is an appealing way to regularize the training process. In this Section, we review the main existing approaches for combining machine learning with physical knowledge (called *ML/MB*, *gray-box*, or *hybrid* modelling in the literature).

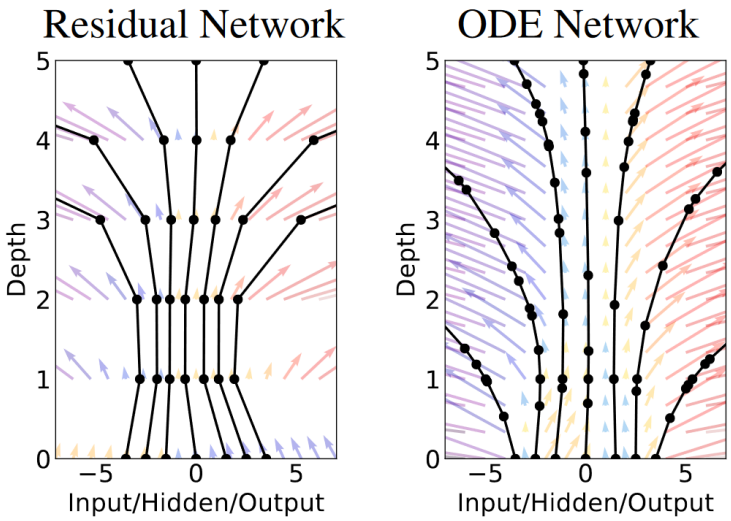


Figure 2.9: Left: a residual neural network [101] defines a discrete sequence of layers from the input to the output. Right: a Neural ODE [33] solves an ODE starting from the input for evolving the hidden state. Figure taken from [33].

2.3.1 Continuous time models

Continuous-time models, consisting in modelling the rate of change F of an ODE with a neural network, were first explored from the 1980's [45, 93, 294]. More recently, researchers have drawn tight connections between dynamical systems and deep (residual) neural networks [273, 158, 302, 33]. The residual bloc of a ResNet [101]

$$\mathbf{h}_{t+1} = \mathbf{h}_t + \Delta t \; F(\mathbf{h}_t, \theta) \tag{2.14}$$

can be interpreted as the forward Euler discretization of the dynamical system

$$\frac{\mathrm{d}\mathbf{h}(t)}{\mathrm{d}t} = F(\mathbf{h}(t), \theta). \tag{2.15}$$

Mainstream recurrent neural networks also have a continuous-time ODE counterpart. The vanilla RNN $\mathbf{h}_t = F(\mathbf{W}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x}_t + \mathbf{b})$ in Eq 2.11 is the Euler discretization of the following ODE:

$$\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) = F(\mathbf{W}\mathbf{h}(t) + \mathbf{U}\mathbf{x}(t) + \mathbf{b}) - \mathbf{h}(t). \tag{2.16}$$

We derive the associated ODE formulation for the LSTM [102] and the Gated Recurrent Unit (GRU) [39] in Appendix II, which makes our ODE assumptions for forecasting (Eq 2.8) quite general.

Since, many other successful deep architectures have been linked to numerical schemes for ODEs [159, 75] and new architectures were proposed and analyzed with the rich dynamical system theory [99, 215, 197, 30, 8], e.g. with the notions of stability or reversibility.

The Neural ODEs (or ODE networks) of Chen *et al.* [33] consider the continuous-time limit in residual networks. Instead of a discrete sequence of layers (or timesteps in a RNN), the evolution of the hidden state in the network is supposed to follow an ODE. This leads to a continuous transformation

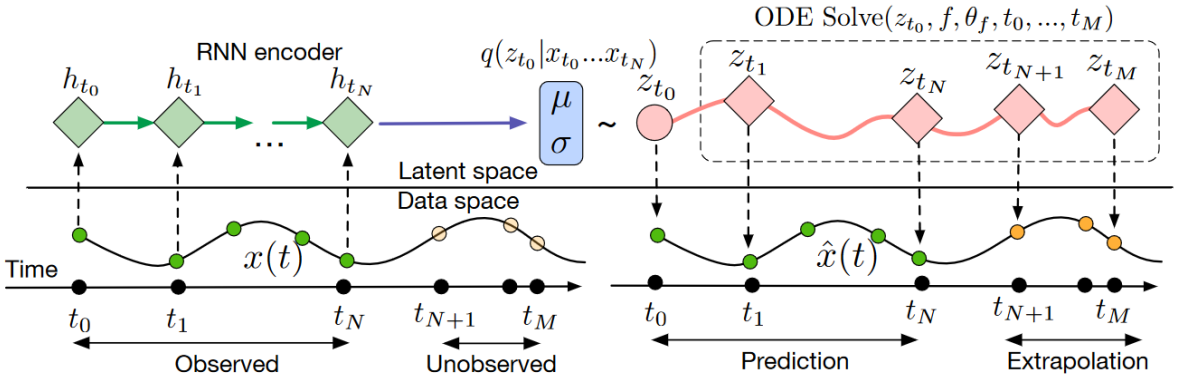


Figure 2.10: Modelling dynamical systems with Neural ODEs. From an initial condition z_{t_0} inferred by an encoder network, the latent trajectory is computed by solving the dynamical model F_θ (parameterized by a neural network) by a differentiable ODE solver. Figure taken from [33].

of the hidden state as shown in Figure 2.9. Neural ODEs are trained with the adjoint sensitivity method [194], which consists in solving a backward ODE instead of backpropagating through the operations of the solver[1]. Many extensions and analyses of Neural ODEs were subsequently proposed [69, 5, 165, 111, 297, 286] and have shown great successes in several tasks such as generative models with normalizing flows [95] or modelling continuous-time data [213, 100].

For predicting dynamical systems, the advantages of the continuous-time modelling of Neural ODEs are twofold. First, Neural ODEs can accommodate any ODE solver, in particular adaptive solvers that automatically adapt the number of iterations in function of the complexity of the dynamics to reach a given accuracy. Second, Neural ODEs can seamlessly handle irregularly-sampled temporal data, which arises in many applications (e.g. medical records) or in case of missing data.

Neural ODEs provide a generative approach for modelling dynamical systems. As illustrated in Figure 2.10, time series are represented by a latent trajectory $z(t)$ governed by a dynamical function F_θ parameterized by a neural network: $\frac{\partial z(t)}{\partial t} = F_\theta(z(t))$. The latent trajectory is computed by solving the ODE with a differentiable ODE solver from an initial condition z_{t_0} (which is known or estimated via an encoder network on an input trajectory). The solution can be evaluated for any time point in the observation range $[t_0, t_N]$ (interpolation) or in the future $[t_N; \infty[$ (extrapolation). The dynamical model F_θ is trained by reconstructing the trajectories of a training dataset.

Although Neural ODEs offer a principled way to model dynamical systems with deep networks in continuous-time, the dynamical model F_θ is still a pure data-driven component and suffers from the main drawbacks as pure ML methods, i.e. overfitting in data-scarce contexts and lack of physical plausibility. In this thesis, we explore how to structure the function F with prior physical knowledge.

2.3.2 Physically-constrained machine learning

In recent years, many researchers have explored how to incorporate physical knowledge into ML models to regularize learning and improve performances. A first solution, made popular by the Physics-

[1]This ensures a lower memory footprint for Neural ODEs: intermediate network activations do not need to be stored during the forward pass since they can be recomputed on the fly by solving the backward ODE.

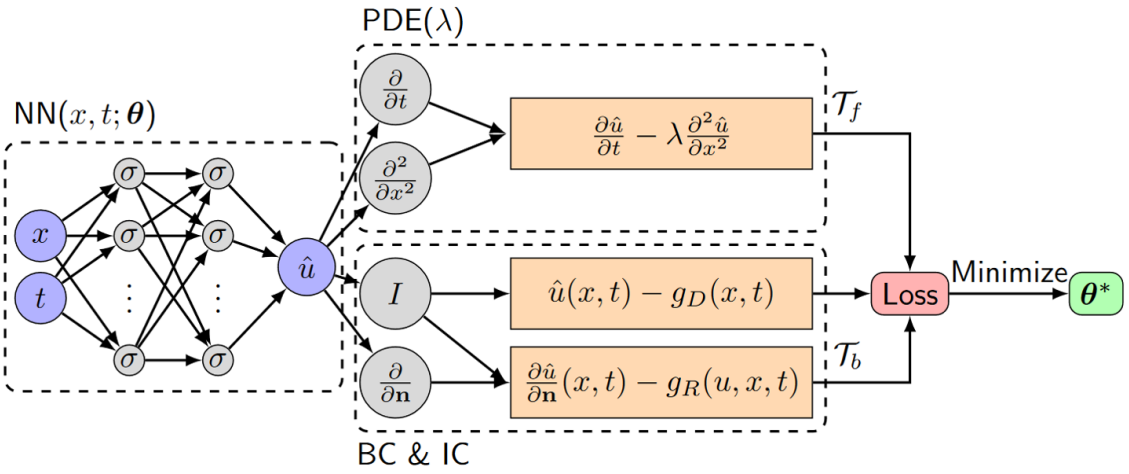


Figure 2.11: Physics-Informed Neural Networks (PINN) for solving the heat equation.

Informed Neural Networks (PINNs) of Raissi *et al.* [199], is to add a physical regularization term in the loss function. Illustrated in Figure 2.11 for solving the heat equation, PINNs are composed of a neural network for predicting the solution $\hat{u}(x,t)$ at a given spatio-temporal location. Partial derivatives are computed during the forward pass by automatic differentiation to form the PDE residual. The total loss function is the sum of the data fidelity term and the adequacy to the PDE constraint and boundary conditions. PINNs are very easy to implement in standard deep learning libraries such as TensorFlow or PyTorch.

In their initial form, PINNs need to be retrained for each new set of the parameters of the PDE. In order to learn a class of PDEs, Sirignano *et al.* [231] propose to add the PDE parameters as inputs of the physics-informed neural network, and the neural operator approaches propose to directly learn the solution operator of a parametric class of PDEs [149, 157, 150, 265]. However, this class of methods only impose soft constraints, i.e. the physical laws are not strictly guaranteed to be respected.

Other works investigate introducing hard physical constraints in the network architectures. Daw *et al.* [57] propose a monotonicity-preserving architecture for modelling lake temperature along depth, by adapting the LSTM with additional variables playing the role of positive increments. Mohan *et al.* [174] impose the divergence-free constraint of incompressible flows by parameterizing the flow as the curl of a learned scalar potential.

For modelling fluids, De Bezenac *et al.* [58] propose a hybrid ML/MB architecture that explicitly exploits the advection-diffusion PDE:

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I. \tag{2.17}$$

Given a sequence of past images, their deep architecture estimates the flow field w and the diffusion coefficient D, which are used in a warping scheme implementing the closed-formed solution of the PDE. The model is learned end-to-end for predicting the next frame, without any supervision for the physical parameters. The authors successfully apply this model to predict Sea Surface Temperature (SST) maps.

Physical systems are often studied through the conservation of energy, which is encoded in a

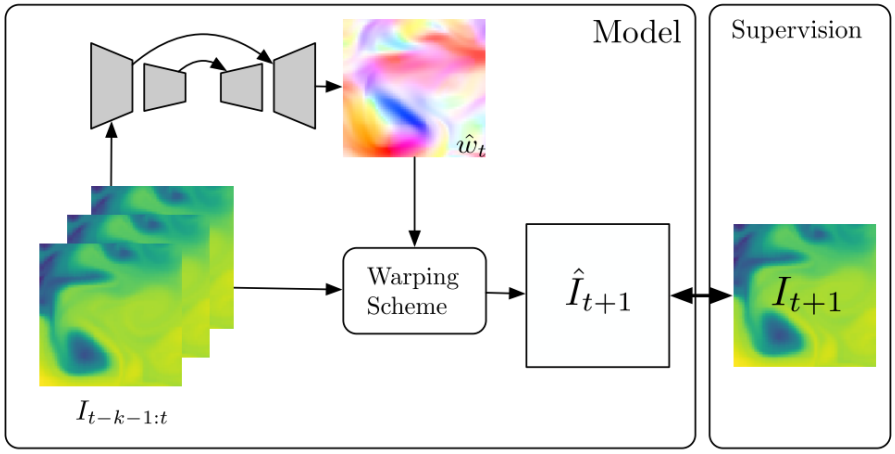


Figure 2.12: Hybrid ML/MB architecture of De Bezenac *et al.* [58] for predicting Sea Surface Temperature with the advection-diffusion PDE.

principled way through Hamiltonian dynamics. Greydanus *et al.* [96] introduce the Hamiltonian Neural Networks to learn physical systems respecting the conservation of energy. With $\mathbf{q}$ the position of a set of particles and $\mathbf{q}$ their momentum, the Hamiltonian $\mathcal{H}(q, p)$ representing the total energy of the systems, obeys the following equations:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}} \quad , \quad \frac{d\mathbf{p}}{dt} = -\frac{\partial \mathcal{H}}{\partial \mathbf{q}}. \tag{2.18}$$

HNNs learn the Hamiltonian with a NN and take in-graph gradients to impose the Hamiltonian dynamics. They show in experiments that it better conserves energy than baselines.

Many of the ML/MB approaches described so far are tailored for specific applications, e.g. fluid dynamics [58], molecular dynamics [38], quantum mechanics [225], robotics [162], and are thus not applicable to other domains. Moreover, they often rely on a complete knowledge of the physical equations, and further assume that these equations directly apply in the input space (observed prior as defined in Chapter 1). In this thesis, we explore general augmentation strategies that can be applied to all levels of prior knowledge, from the more general prior to the most application-specific equations. We also tackle the case of the unobserved prior by learning representations spaces in which the physical laws apply.

2.3.3 Identifying and discovering physical systems

Beyond forecasting physical systems, researchers have also explored machine learning for system identification, which consists in estimating the unknown parameters in parameterized physical equations. A basic example is estimating the length of a damped pendulum from observed trajectories. Automatically identifying and discovering physical laws from observations is a long-standing goal for physicists, with many applications in control [117] or robotics [162]. Many approaches use symbolic regression to search the space of possible mathematical functions, using evolutionary algorithms [222], sparse regression on dictionaries of potential differentiable terms [23, 214, 221], or graph neural networks [54].

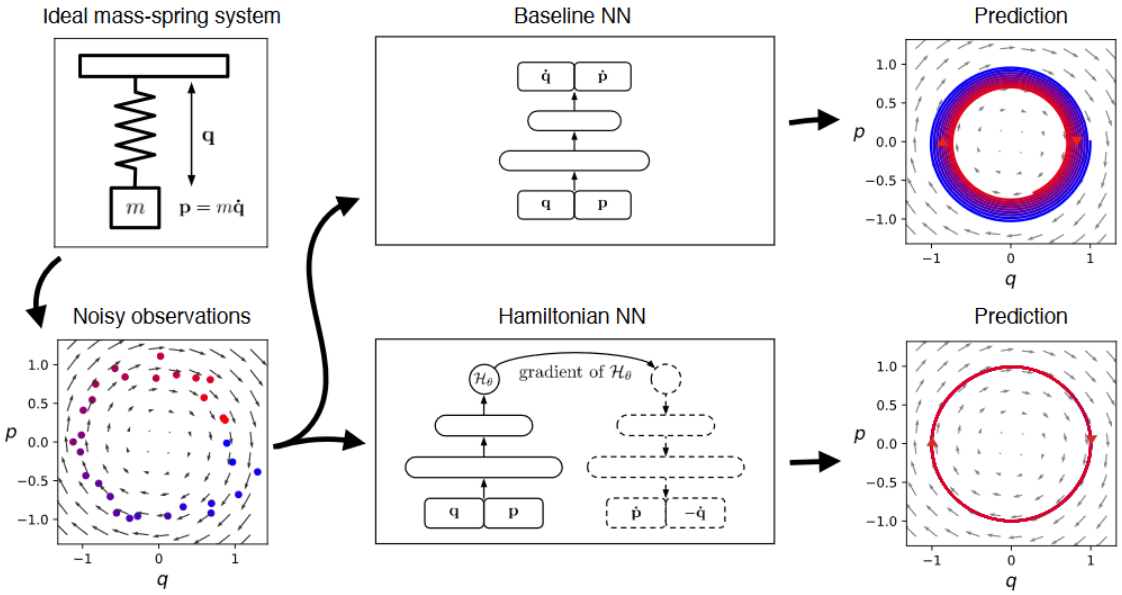


Figure 2.13: Hamiltonian Neural Networks of Greydanus *et al.* [96].

Several architectures attempt to predict and identify the PDE governing physical systems [155, 201], such as the the PDE-Net architecture of Long *et al.* [155, 154]. As shown in Figure 2.14, the basic bloc composing PDE-Net (the δt-bloc) is a residual module implementing one forward Euler discretization step. For solving the PDE $\frac{\partial u}{\partial t} = F(u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \cdots)$, the authors use convolutional filters that are constrained to approximate each spatial differential term (we give details about these constrained convolutions in Appendix D.1.2[2]. Then a symbolic neural network identifies the nonlinear relationships between the spatial derivatives to form the nonlinear function F of the PDE. A skip connection finally provides the prediction of the next timestep $\hat{u}(t+\delta t) = \hat{u}(t) + \delta t \hat{F}$. The complete PDE-Net architecture is composed of several δt-blocs concatenated in time for long-term prediction.

In this thesis, we take inspiration from the PDE-Net architecture for imposing physical dynamics, and we take a step further by assuming incomplete physical models and by modelling the residual dynamics for accurate prediction. We also show that a careful training scheme leads to a better identification of the physical parameters than simplified physical model alone.

[2]They show that the flexibility of learned differential filters boost performances compared to handcrafted filters, an observation that has been noted for other discretization schemes learned from data.

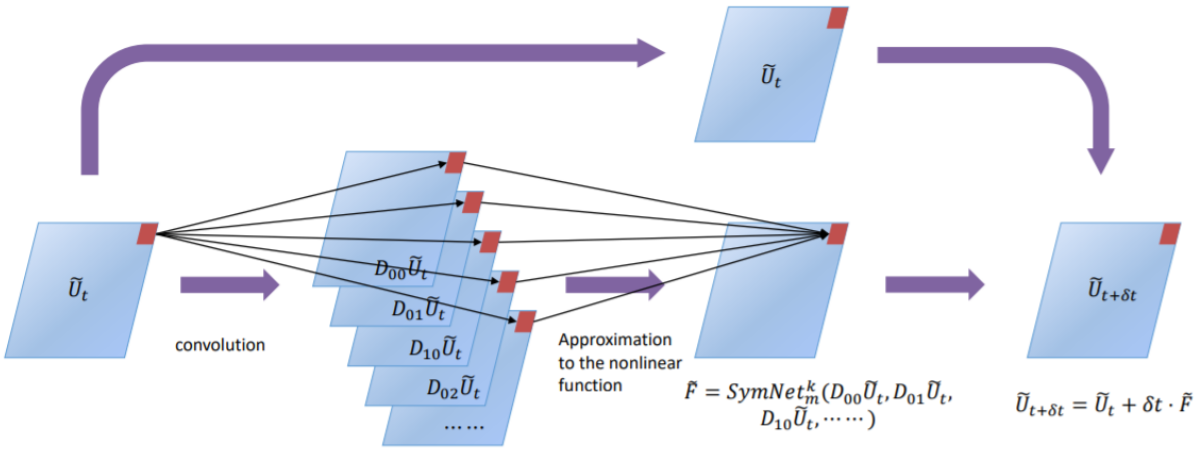


Figure 2.14: The basis δt block composing the PDE-Net architecture implements on step of forward Euler integration. Constrained convolutional filters estimate each spatial derivative term, that are combined by a symbolic network that estimates the dynamical function F. Finally a skip connection provides the solution for the next timestep. Figure taken from [154].

2.3.4 Augmented physical models

There exists an abundant literature on statistical methods for calibrating and predicting physical systems in presence of model inadequacy, often expressed in a Bayesian framework; a review of these methods can be found in [193]. In data assimilation techniques, like the Kalman filter [116], the particle filter [192] or 4D-var [52], the predictions errors are modelled probabilistically with random variables reflecting the noise assumption. A correction step using observed data is performed after each prediction step for filtering the noise. Similar residual correction procedures are commonly used in robotics and optimal control [34, 146]. However, these sequential (two-stage) procedures prevent the cooperation between prediction and correction. Besides, in model-based reinforcement learning, model deficiencies are typically handled by considering only short-term rollouts [108] or by model predictive control [178] consisting in replanning frequently to mitigate error propagation.

In this thesis, we take inspiration from data assimilation ideas to augment incomplete physical models with residual terms. However, in contrast to data assimilation, our residual terms are not assumed to correspond to be a stochastic residual, i.e. noise, but to a systematic unmodelled part of the dynamics that we learned from data. Moreover, we derive a principled training scheme for making the prediction and correction steps cooperate.

The idea of augmenting physical models with neural networks (*gray-box or hybrid* modelling) is not new: in the 1990's, the works [196, 245, 206] use neural networks to estimate the unknown parameters of physical models that are difficult to model from first principles, and a classification of the possible augmentation strategies (serial, parallel, modular) was dressed [245]. The challenge of proper ML/MB cooperation was already raised as a limitation of gray-box approaches but not addressed. Moreover these methods were evaluated on specific applications with a residual targeted to the form of the equation.

In the last few years, there has been a growing interest in deep augmented models that combine physical priors with deep networks [153, 216, 168]. Several ML/MB cooperation schemes with deep

networks were studied in [264, 168]. Again, these approaches do not address the issues of uniqueness of the decomposition or of proper cooperation for correct parameter identification. They are also mostly dedicated to the fully-observable case, whereas we also tackle the non-observable prior context in this thesis. We further detail the literature on augmented physical models in Chapter 7.

Part I

Differentiable shape and time criteria for deterministic and probabilistic forecasting

Abstract

In this part, we tackle the multistep deep time series forecasting problem, in the challenging context of non-stationary series that can present sharp variations. In deep learning, the mainstream research direction concerns developing new neural forecasting architectures. In contrast, the choice of the training loss function is rarely questioned: the surrogate mean squared error (MSE) is used in the vast majority of cases. We propose here to leverage shape and temporal criteria in the training objective. We introduce differentiable similarities and dissimilarities for characterizing shape accuracy and temporal localization error (Chapter 3). We leverage these criteria by introducing two approaches dedicated to deterministic and probabilistic forecasting: we introduce the DILATE loss function for deterministic forecasting that ensures both sharp predictions with accurate temporal localization (Chapter 4), and the STRIPE model for probabilistic forecasting with shape and temporal diversity (Chapter 5). We validate our claims with extensive experiments on synthetic and real-world datasets.

Chapter 3

Differentiable shape and temporal criteria

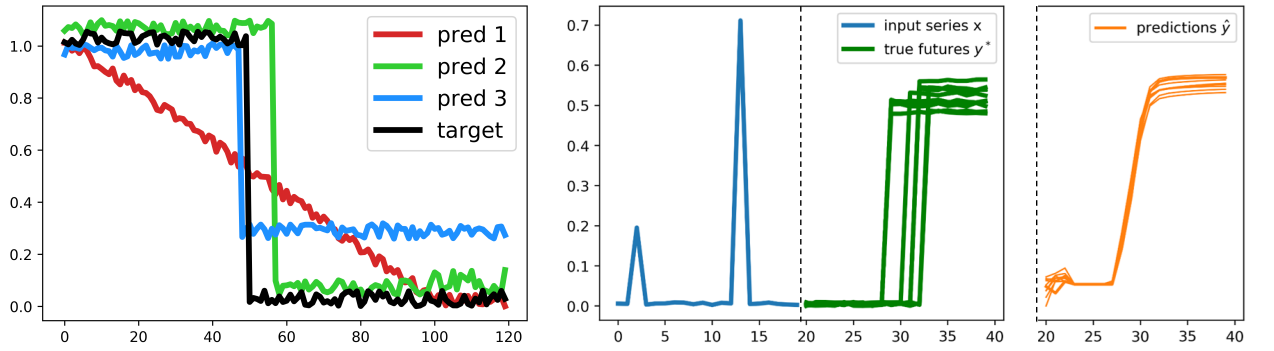
Content

Chapter abstract

In this Chapter, we highlight the limitations of the Mean Squared Error (MSE) loss function dominantly used for time series forecasting. As an alternative, we propose to leverage shape and temporal features at training time. We introduce differentiable similarities and dissimilarities for characterizing shape accuracy and temporal localization error. We characterize the shape with the Dynamic Time Warping (DTW) [217] algorithm and the temporal error with the Temporal Distortion Index (TDI) [83]. We provide an unified view of these criteria by formulating them in terms of dissimilarities (loss functions) and similarities (positive semi-definite kernels). We also insist on their differentiability and efficient computation. The work described in this Chapter is based on the following publication:

- *Vincent Le Guen and Nicolas Thome. "Deep Time Series Forecasting with Shape and Temporal Criteria". IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.*

3.1 Introduction



(a) Deterministic forecasting **(b) Probabilistic forecasting**

Figure 3.1: **MSE limitations in deterministic and probabilistic forecasting.** (a) For deterministic forecasting, the three predictions (1,2,3) have the same MSE with respect to the target (in black). However, one would like to favour prediction 2 (correct shape, slight delay) and 3 (correct timing, inaccurate amplitude) over prediction 1 (which is not very informative). (b) For probabilistic forecasting, state-of-the-art methods trained with variants of the MSE (e.g. [292, 203]) loose the ability to produce sharp forecasts (in orange) compared to the ground truth future trajectories (in green).

Time series forecasting consists in analyzing historical signal correlations to anticipate future behaviour. As discussed in Chapter 2, traditional approaches include linear autoregressive methods [21] or state space models [70], which are simple yet mathematically grounded and benefit from interpretability. They often exploit prior knowledge based on stationarity, e.g. by leveraging trend or seasonality to constrain forecasting.

These grounding assumptions are often violated in many real-world time series that are non-stationary and can present sharp variations such as sudden drops or changes of regime. Long-term multi-step forecasting in this context is particularly challenging and arises in a wide range of important application fields, e.g. analyzing traffic flows [147, 234], medical records [32], predicting sharp variations in financial markets [64] or in renewable energy production [252, 89, 138], *etc.*

We are interested in forecasting multi-step future trajectories with potentially sharp variations in the deterministic and probabilistic cases. Deep neural networks are an appealing solution for this problem [291, 198, 133, 219, 183, 300], due to their automatic feature extraction and complex nonlinear time dependencies modelling. However, the verification criteria typically used in applications are not used at training time because they are mostly not differentiable. We may cite for instance the ramp score [252] for assessing the detection of sharp ramping events, or the Time Distortion Index (TDI) [83] for assessing the time delay of a particular predicted event.

Instead, the huge majority of methods optimize at training time the Mean Squared Error (MSE) or its variants (MAE, quantile loss, *etc*) as a proxy loss function. However, the MSE has important drawbacks in our non-stationary context, as also noted by several other works [252, 257, 284]. This is illustrated in Figure 3.1. Figure 3.1 (a) shows three deterministic predictions, which have the same MSE loss compared to the target step function (in black). Thus, the MSE does not support predictions

(2) and (3) over prediction (1), although they clearly are more adequate for regulation purposes because they do anticipate the drop to come, although with a slight delay (2) or with a slightly inaccurate amplitude (3). For probabilistic forecasting (Figure 3.1 (b)), current state-of-the art probabilistic methods trained with variants of the MSE tend to produce blurry predictions that do not match the sharp steps of the true futures (in green).

We intend to bridge this train/test criterion gap by incorporating shape and temporal features at training time. In this Chapter, we introduce shape and temporal criteria for training deep forecasting models. We characterize the shape of times series with the Dynamic Time Warping (DTW) [217] algorithm and the temporal shift with the Temporal Distortion Index (TDI) [83]. We provide an unified view of these criteria by formulating them both as dissimilarities (loss functions) and similarities (positive semi-definite kernels). Importantly, we insist on their differentiability, which makes them amenable to gradient-based optimization, and on their efficient computation.

3.2 Shape (dis)similarity

3.2.1 Background: Dynamic Time Warping

To assess the shape similarity between two time series, the popular Dynamic Time Warping (DTW) method [217] seeks a minimal cost alignment for handling time distortions. Given two d-dimensional time series $\mathbf{y} \in \mathbb{R}^{d\times n}$ and $\mathbf{z} \in \mathbb{R}^{d\times m}$ of lengths n and m, DTW looks for an optimal warping path represented by a binary matrix $\mathbf{A} \subset \{0,1\}^{n\times m}$ where $\mathbf{A}_{ij} = 1$ if $\mathbf{y}_i$ is associated to $\mathbf{z}_j$ and 0 otherwise. The set of admissible warping paths $\mathcal{A}_{n,m}$ is composed of paths connecting the endpoints $(1,1)$ to (n,m) with the following authorized moves $\rightarrow, \downarrow, \searrow$. The cost of warping path $\mathbf{A}$ is the sum of the costs along the alignment ; this cost can be written as the scalar product $\langle \mathbf{A}, \boldsymbol{\Delta}(\mathbf{y},\mathbf{z})\rangle$, where $\Delta(\mathbf{y},\mathbf{z})$ is a $n \times m$ pairwise dissimilarity matrix whose general term is typically chosen an the Euclidean distance $\boldsymbol{\Delta}(\mathbf{y},\mathbf{z})_{ij} = \|\mathbf{y}_i - \mathbf{z}_j\|_2^2$. DTW computes the minimal cost warping path:

$$\mathrm{DTW}^{\boldsymbol{\Delta}}(\mathbf{y},\mathbf{z}) := \min_{\mathbf{A}\in\mathcal{A}_{n,m}} \langle \mathbf{A}, \boldsymbol{\Delta}(\mathbf{y},\mathbf{z})\rangle \,. \tag{3.1}$$

Although the cardinality of $\mathcal{A}_{n,m}$ increases exponentially in $\min(n,m)$ [1], DTW and the optimal path $\mathbf{A}^*$ can be computed efficiently in $\mathcal{O}(nm)$ by dynamic programming. However, a major limitation of DTW is its non-diffentiability, which prevents its integration in neural network pipelines trained with gradient-based optimization.

3.2.1.1 Smooth DTW shape dissimilarity

For handling the non-differentiability of DTW, Cuturi and Blondel [55] introduced the soft-DTW by replacing the hard-minimum operator by a smooth minimum with the log-sum-exp trick $\min_\gamma(a_1,...,a_n) = -\gamma\log(\sum_i^n \exp(-a_i/\gamma))$:

$$\mathrm{DTW}_\gamma^{\boldsymbol{\Delta}}(\mathbf{y},\mathbf{z}) := -\gamma \log\left(\sum_{\mathbf{A}\in\mathcal{A}_{n,m}} e^{-\langle \mathbf{A}, \boldsymbol{\Delta}(\mathbf{y},\mathbf{z})\rangle/\gamma}\right), \tag{3.2}$$

[1] $|\mathcal{A}_{n,m}|$ is equal to the Delannoy number $Delannoy(n,m)$ which grows exponentially in $\min(n,m)$

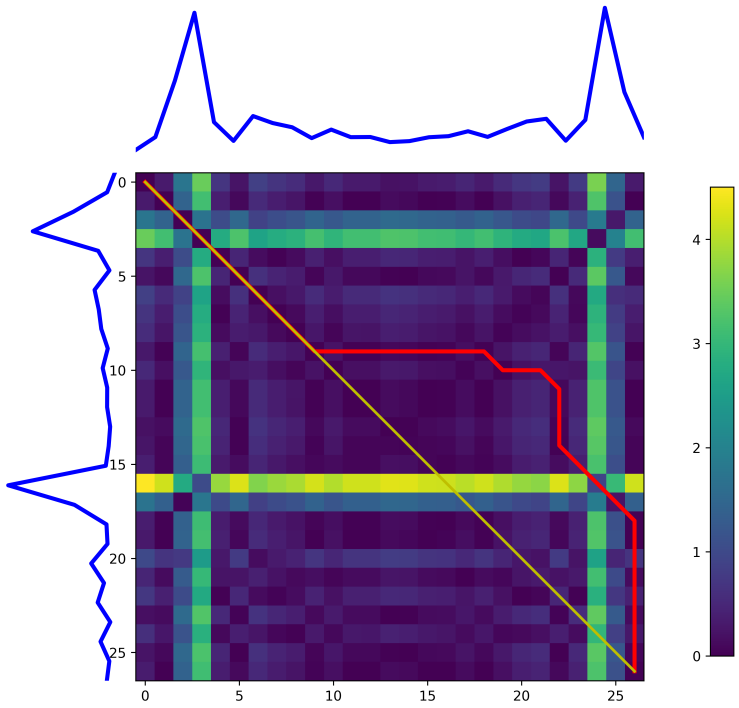


Figure 3.2: **Dynamic Time Warping (DTW)** seeks a path of minimal alignment cost (in red) in the pairwise cost matrix between the two time series.

where $\gamma > 0$ is a smoothing parameter (when $\gamma \to 0$, this converges to the true DTW).

$\text{DTW}_\gamma^{\mathbf{\Delta}}$ as defined in Eq 3.2 is differentiable with respect to $\mathbf{\Delta}$ (and with respect to both series $\mathbf{y}$ and $\mathbf{z}$ by chain's rule, provided a differentiable cost function $\mathbf{\Delta}$).

We can interpret this relaxed DTW version by considering, instead of the unique optimal path $\mathbf{A}^*$, a Gibbs distribution over possible paths:

$$p_\gamma(\mathbf{A}; \mathbf{\Delta}) = \frac{1}{Z} e^{-\langle \mathbf{A}, \mathbf{\Delta}(\mathbf{y},\mathbf{z}) \rangle / \gamma}. \tag{3.3}$$

The soft-DTW is then the negative log-partition of this distribution: $\text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z}) := -\gamma \log Z$.

Since $\text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z})$ can take negative values and is not minimized for $\mathbf{y} = \mathbf{z}$, Mensch and Blondel [171] normalized the soft-DTW to make it a true divergence. We found experimentally that this does not improve performances and is heavier computationally (see Appendix B.2).

3.2.1.2 Shape similarity kernel

Based on the soft-DTW shape dissimilarity defined in Eq 3.2, we define a shape similarity kernel as follows:

$$\mathcal{K}_{shape}(\mathbf{y}, \mathbf{z}) = e^{-\text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y},\mathbf{z})/\gamma}. \tag{3.4}$$

We experiment with the following choices of kernels $\Delta_{ij} = \Delta(\mathbf{y}, \mathbf{z})_{ij}$:

- Half-Gaussian: $\mathbf{\Delta}_{ij} = \|\mathbf{y}_i - \mathbf{z}_j\|_2^2 + \log(2 - e^{-\|\mathbf{y}_i - \mathbf{z}_j\|_2^2})$
- L1: $\mathbf{\Delta}_{ij} = |\mathbf{y}_i - \mathbf{z}_j| \quad$ (for $d = 1$)
- Euclidean: $\mathbf{\Delta}_{ij} = \|\mathbf{y}_i - \mathbf{z}_j\|_2^2$.

$\mathcal{K}_{shape}$ was proven to be positive semi-definite (PSD) for the half-Gaussian[2] and the L1 kernels [56, 17] and is conjectured to be PSD for the Euclidean kernel [17]. Experimentally we observed that these three cost matrices lead to similar behaviour.

3.3 Temporal (dis)similarity

Quantifying the temporal similarity between two time series consists in analyzing the time delays between matched patterns detected in both series. As discussed in introduction, it of great importance for many applications to anticipate sharp variations.

3.3.1 Smooth temporal distortion index

A common temporal similarity is the Temporal Distortion Index (TDI) [83, 252]. The TDI computes the approximate area included between the optimal path $\mathbf{A}^*$ and the first diagonal, characterizing the presence of temporal distortion. A generalized version of the TDI, that we proposed in [137], can be written:

$$\mathrm{TDI}^{\mathbf{\Delta},\mathbf{\Omega}_{\mathbf{dissim}}}(\mathbf{y},\mathbf{z}) := \langle \mathbf{A}^*, \mathbf{\Omega}_{\mathbf{dissim}} \rangle \,, \tag{3.5}$$

where $\mathbf{A}^* = \underset{\mathbf{A}\in\mathcal{A}_{n,m}}{\arg\min} \langle \mathbf{A}, \mathbf{\Delta}(\mathbf{y},\mathbf{z}) \rangle$ is the DTW optimal path and $\mathbf{\Omega}_{\mathbf{dissim}} \in \mathbb{R}^{n\times m}$ is a matrix penalizing the association between $\mathbf{y}_i$ and $\mathbf{z}_j$ for $i \neq j$. We typically choose a quadratic penalization $\mathbf{\Omega}_{\mathbf{dissim}}(i,j) \propto (i-j)^2$, but other variants can encode prior knowledge and penalize more heavily late than early predictions, and *vice-versa*.

The TDI dissimilarity defined in Eq 3.5 is however non-differentiable, since the optimal path $\mathbf{A}^*$ is not differentiable with respect to $\mathbf{\Delta}$. We handle this problem by defining a relaxed optimal path $\mathbf{A}^*_\gamma$ as the gradient of $\mathrm{DTW}^{\mathbf{\Delta}}_\gamma$:

$$\mathbf{A}^*_\gamma := \nabla_{\mathbf{\Delta}} \mathrm{DTW}^{\mathbf{\Delta}}_\gamma(\mathbf{y},\mathbf{z}) = \frac{1}{Z} \sum_{\mathbf{A}\in\mathcal{A}_{n,m}} \mathbf{A}\, e^{-\langle \mathbf{A}, \mathbf{\Delta}(\mathbf{y},\mathbf{z}) \rangle / \gamma}. \tag{3.6}$$

The expression in Eq 3.6 results from a direct computation from Eq. 3.2. Notice that this soft optimal path corresponds to the expected path $\mathbf{A}^*_\gamma = \mathbb{E}_{p_\gamma(\cdot;\mathbf{\Delta})}[\mathbf{A}]$ under the Gibbs distribution in Eq 3.3. Note also that $\mathbf{A}^*_\gamma$ becomes a soft assignment, i.e. $\mathbf{A}^*_\gamma(i,j)$ represents the probability for a path to contain the cell (i,j). An illustration of soft optimal paths with the influence of γ is given in Figure 4.6.

We can now define a differentiable version of the TDI:

$$\mathrm{TDI}^{\mathbf{\Delta},\mathbf{\Omega}_{\mathbf{dissim}}}_\gamma(\mathbf{y},\mathbf{z}) := \left\langle \mathbf{A}^*_\gamma, \mathbf{\Omega}_{\mathbf{dissim}} \right\rangle = \frac{1}{Z} \sum_{\mathbf{A}\in\mathcal{A}_{n,m}} \langle \mathbf{A}, \mathbf{\Omega}_{\mathbf{dissim}} \rangle\, e^{-\frac{\langle \mathbf{A}, \mathbf{\Delta}(\mathbf{y},\mathbf{z}) \rangle}{\gamma}}, \tag{3.7}$$

[2] We denote this kernel "half-Gaussian" since the corresponding k kernel defined in the proof (Appendix A.1) equals $k(\mathbf{y}_i,\mathbf{z}_j) = e^{-\Delta(\mathbf{y}_i,\mathbf{z}_j)} = \left(\frac{1}{2}e^{-\|\mathbf{y}_i-\mathbf{z}_j\|^2}\right) \times \left(1 - \frac{1}{2}e^{-\|\mathbf{y}_i-\mathbf{z}_j\|^2}\right)^{-1}$

which corresponds to the expected value of the TDI under the Gibbs distribution.

3.3.2 Temporal similarity kernel

Based on the temporal dissimilarity in Eq 3.7 and the shape similarity kernel in Eq. 3.4, we can define a time similarity as follows:

$$\mathcal{K}_{time}(\mathbf{y}, \mathbf{z}) := e^{-\mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y},\mathbf{z})/\gamma} \times \mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{sim}}}(\mathbf{y}, \mathbf{z}), \tag{3.8}$$

where in this case, we use a similarity matrix $\mathbf{\Omega_{sim}}$ favoring pairs of time series with low temporal distortion, i.e. with an optimal path near the main diagonal. We typically choose a pointwise inverse of $\mathbf{\Omega_{dissim}}$: $\mathbf{\Omega_{sim}}(i,j) = \frac{1}{(i-j)^2+1}$. We prove that $\mathcal{K}_{time}$ defines a valid PSD temporal kernel (proof in Appendix A.1).

The following table provides an overview of the shape and temporal criteria introduced in this work:

criterion	differentiable loss	PSD similarity kernel
shape	$\mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z})$	$e^{-\mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y},\mathbf{z})/\gamma}$
time	$\mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{dissim}}}(\mathbf{y}, \mathbf{z})$	$e^{-\mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y},\mathbf{z})/\gamma} \times \mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{sim}}}(\mathbf{y}, \mathbf{z})$

3.3.3 Efficient forward and backward computation

The direct computation of the shape loss $\mathrm{DTW}_\gamma^{\mathbf{\Delta}}$ (Eq 3.2) and the temporal loss $\mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{dissim}}}$ (Eq 3.7) is intractable, due to the exponential growth of cardinal of $\mathcal{A}_{n,m}$. We provide a careful implementation of the forward and backward passes in order to make learning efficient.

Shape loss: Regarding $\mathrm{DTW}_\gamma^{\mathbf{\Delta}}$, we rely on [55] to efficiently compute the forward pass with a variant of the Bellmann dynamic programming approach [13]. For the backward pass, we implement the recursion proposed in [55] in a custom Pytorch loss. This implementation is much more efficient than relying on vanilla auto-differentiation, since it reuses intermediate results from the forward pass.

Temporal loss: For $\mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{dissim}}}$, note that the bottleneck for the forward pass in Eq 3.7 is to compute $\mathbf{A}_\gamma^* = \nabla_\Delta \mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z})$, which we implement as explained for the $\mathrm{DTW}_\gamma^{\mathbf{\Delta}}$ backward pass. Regarding $\mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{dissim}}}$ backward pass, we need to compute the Hessian $\nabla^2 \mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z})$. We use the method proposed in [171], based on a dynamic programming implementation that we embed in a custom Pytorch loss. Again, our back-prop implementation allows a significant speed-up compared to standard auto-differentiation. The resulting time complexity of both shape and temporal losses for forward and backward is $\mathcal{O}(nm)$.

Custom backward implementation speedup: We compare in Figure 3.3 the computational time between the standard PyTorch auto-differentiation mechanism and our custom backward pass implementation for calculating $\mathrm{DTW}_\gamma^{\mathbf{\Delta}} + \mathrm{TDI}_\gamma^{\mathbf{\Delta},\mathbf{\Omega_{dissim}}}$ (we will call this quantity the DILATE loss in the

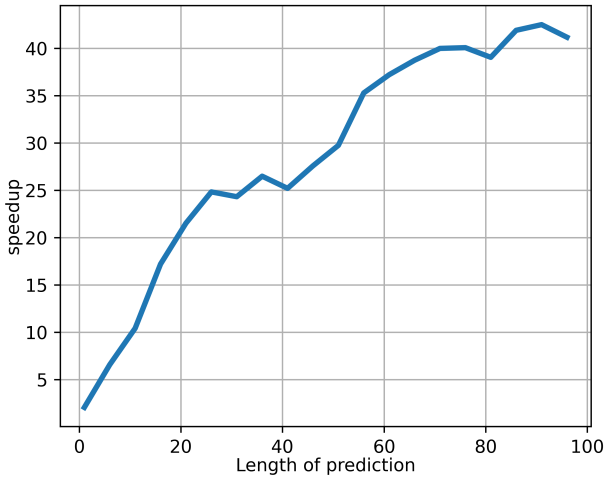


Figure 3.3: Speedup of the custom forward and backward implementation of the DILATE loss introduced in Chapter 4.

next Chapter). We plot the speedup of our implementation with respect to the prediction length H (averaged over 10 random target/prediction tuples). We notice the increasing speedup with respect to H: speedup of $\times$ 20 for 20 steps ahead and up to $\times$ 35 for 100 steps ahead predictions.

3.4 Conclusion

To tackle the multi-step and stationary time series forecasting problem, we question the widely-used MSE training loss that lead to non-sharp predictions. We instead propose to leverage shape and temporal features at training time. In this Chapter, we have introduced differentiable similarities and dissimilarities for characterizing shape accuracy and temporal localization error. Shape is characterized with the Dynamic Time Warping (DTW) [217] algorithm and the temporal error with the Temporal Distortion Index (TDI) [83]. We have provided an unified view of these criteria by formulating them in terms of dissimilarities (loss functions) and similarities (positive semi-definite kernels). We have insisted on their differentiability and efficient computation.

In subsequent Chapters, we provide two implementations for time series forecasting: the DILATE loss function for deterministic forecasting that ensures both sharp predictions with accurate temporal localization (Chapter 4), and the STRIPE model for probabilistic forecasting with shape and temporal diversity (Chapter 5).

Chapter 4

Distortion loss with shape and time

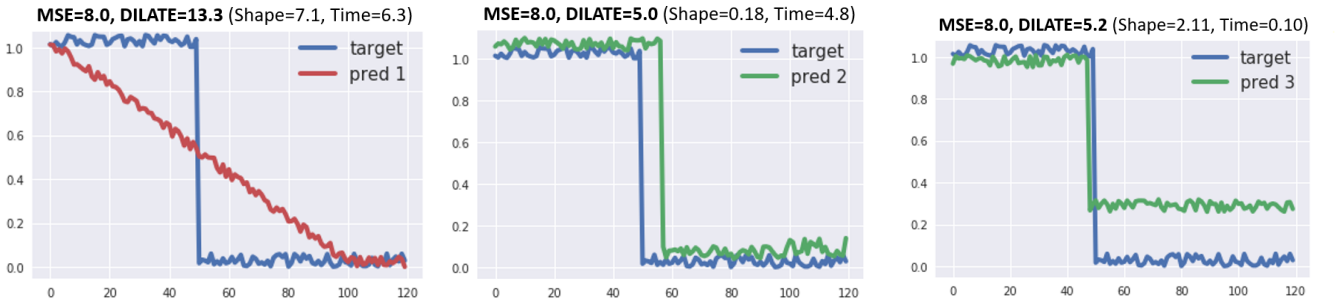
Content

Chapter abstract

In this Chapter, we propose a new differentiable loss function, called DILATE, for training deep multi-step time series forecasting models, in a deterministic context. The DILATE loss builds on the shape and temporal dissimilarities introduced in the previous Chapter. DILATE combines two terms for precise shape and temporal localization of non-stationary signals with sudden changes. The DILATE loss is differentiable, enabling to train any deep forecasting model with gradient-based optimization. We also introduce a variant of DILATE, which provides a smooth generalization of temporally-constrained Dynamic Time Warping (DTW). Extensive experiments on synthetic and real-world datasets show that DILATE is equivalent to the standard MSE loss when evaluated on MSE, and much better when evaluated on several shape and timing metrics. Besides, DILATE improves the performances of state-of-the-art forecasting algorithms trained with the MSE. The work described in this Chapter is based on the following publications:

- *Vincent Le Guen and Nicolas Thome. "Shape and Time Distortion Loss for Training Deep Time Series Forecasting Models". In Advances in Neural Information Processing Systems (NeurIPS 2020).*
- *Vincent Le Guen and Nicolas Thome. "Deep Time Series Forecasting with Shape and Temporal Criteria". IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.*

4.1 Introduction



(a) Non informative prediction (b) Correct shape, time delay (c) Correct time, inaccurate shape

Figure 4.1: **Limitation of the Euclidean (MSE) loss**: when predicting a sudden change (target blue step function), the 3 predictions (a), (b) and (c) have similar MSE but very different forecasting skills. In contrast, the DILATE loss proposed in this work, which disentangles shape and temporal decay terms, supports predictions (b) and (c) over prediction (a) that does not capture the sharp change of regime.

As discussed in the previous Chapter, the Mean Squared Error (MSE) is inadequate in the context of non-stationary time series with sudden variations, as illustrated in Figure 4.1. Here, the target ground truth prediction is a step function (in blue), and we present three predictions, shown in Figure 4.1 (a), (b), and (c), which have a similar MSE loss compared to the target, but very different forecasting skills. Prediction (a) is not adequate for regulation purposes since it doesn't capture the sharp drop to come. Predictions (b) and (c) much better reflect the change of regime since the sharp drop is indeed anticipated, although with a slight delay (b) or with a slight inaccurate amplitude (c).

This Chapter introduces DILATE (DIstortion Loss including shApe and TimE), a new objective function for training deep neural networks in the context of multi-step and non-stationary time series forecasting. DILATE explicitly disentangles into two terms the penalization related to the shape and the temporal localization errors of change detection. The behaviour of DILATE is shown in Figure 4.1: whereas the values of our proposed shape and temporal losses are large in Figure 4.1 (a), the shape (resp. temporal) term is small in Figure 4.1 (b) (resp. Figure 4.1 (c)). DILATE combines shape and temporal terms, and is consequently able to output a much smaller DILATE loss for predictions (b) and (c) than for (a), as expected.

We first present the DILATE loss in section 4.2. We also introduce a variant of DILATE, which provides a smooth generalization of temporally-constrained Dynamic Time Warping (DTW) metrics [217, 110]. Experiments carried out on several synthetic and real non-stationary datasets reveal that models trained with DILATE significantly outperform models trained with the MSE loss function when evaluated with shape and temporal distortion metrics, while DILATE maintains very good performance when evaluated with MSE. Finally, we show that DILATE can be used with various network architectures and can outperform on shape and time metrics state-of-the-art models specifically designed for multi-step and non-stationary forecasting.

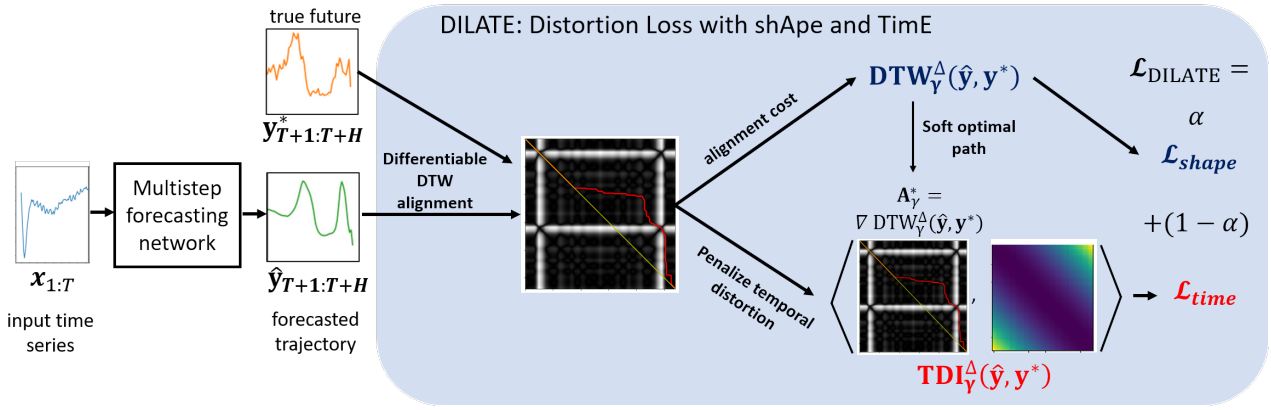


Figure 4.2: **Overview of the DILATE loss:** $\mathcal{L}_{\text{DILATE}}$ for training deterministic deep time series forecasting models is composed of two terms: $\mathcal{L}_{shape}$ based on the soft DTW and $\mathcal{L}_{time}$ that penalizes the temporal distortions visible on the soft optimal path. The overall loss $\mathcal{L}_{\text{DILATE}}$ is differentiable, and we provide an efficient implementation of its forward and backward passes.

4.2 Training Deep Neural Networks with DILATE

Given an input sequence $\mathbf{x}_{1:T} = (\mathbf{x}_1, \ldots, \mathbf{x}_T) \in \mathbb{R}^{p \times T}$, the deterministic multi-step time series forecasting problem consists in predicting a H-steps future trajectory $\hat{\mathbf{y}} = (\hat{\mathbf{y}}_{T+1}, \ldots, \hat{\mathbf{y}}_{T+H}) \in \mathbb{R}^{d \times H}$. As an alternative to the MSE, we introduce here the DIstortion Loss with shApe and TimE (DILATE) for training any deterministic deep multi-step forecasting model. Crucially, the DILATE loss needs to be differentiable in order to train models with gradient-based optimization.

The DILATE objective function, which compares the prediction $\hat{\mathbf{y}} = (\hat{\mathbf{y}}_{T+1}, \ldots, \hat{\mathbf{y}}_{T+H})$ with the actual ground truth future trajectory $\mathbf{y}^* = (\mathbf{y}^*_{T+1}, \ldots, \mathbf{y}^*_{T+H})$, is composed of two terms balanced by the hyperparameter $\alpha \in [0, 1]$:

$$\mathcal{L}_{\text{DILATE}}(\hat{\mathbf{y}}, \mathbf{y}^*) = \alpha\ \mathcal{L}_{shape}(\hat{\mathbf{y}}, \mathbf{y}^*) + (1-\alpha)\ \mathcal{L}_{time}(\hat{\mathbf{y}}, \mathbf{y}^*) \tag{4.1}$$

$$= \alpha\ \text{DTW}^{\mathbf{\Delta}}_{\gamma}(\hat{\mathbf{y}}, \mathbf{y}^*) + (1-\alpha)\ \text{TDI}^{\mathbf{\Delta}, \mathbf{\Omega}_{\text{dissim}}}_{\gamma}(\hat{\mathbf{y}}, \mathbf{y}^*). \tag{4.2}$$

The computational graph of the DILATE loss is illustrated in Figure 4.2. We use for the shape term $\mathcal{L}_{shape}$ the smooth shape dissimilarity $\text{DTW}^{\mathbf{\Delta}}_{\gamma}$ defined in Eq 3.2 and for the temporal term $\mathcal{L}_{time}$ the time dissimilarity $\text{TDI}^{\mathbf{\Delta}, \mathbf{\Omega}_{\text{dissim}}}_{\gamma}$ defined in Eq 3.7.

Tangled DILATE variant A variant of our approach to combine shape and temporal penalization would be to incorporate a temporal term inside our smooth $\mathcal{L}_{shape}$ function in Eq 3.2, leading to a *tangled* version $\mathcal{L}_{\text{DILATE}^t}$:

$$\mathcal{L}_{\text{DILATE}^t}(\hat{\mathbf{y}}_i, \mathbf{y}^*_i) := -\gamma \log \left(\sum_{\mathbf{A} \in \mathcal{A}_{\tau,\tau}} \exp\left(-\frac{\langle \mathbf{A}, \alpha \mathbf{\Delta}(\hat{\mathbf{y}}_i, \mathbf{y}^*_i) + (1-\alpha)\mathbf{\Omega} \rangle}{\gamma} \right) \right). \tag{4.3}$$

We can notice that Eq 4.3 reduces to minimizing $\langle \mathbf{A}, \alpha \mathbf{\Delta}(\hat{\mathbf{y}}_i, \mathbf{y}^*_i) + (1-\alpha)\mathbf{\Omega} \rangle$ when $\gamma \to 0^+$. In

this case, $\mathcal{L}_{\text{DILATE}^t}$ can recover DTW variants studied in the literature to bias the computation based on penalizing sequence misalignment, by designing specific $\mathbf{\Omega}$ matrices:

Sakoe-Chiba DTW band constraint [217]	$\Omega(i,j) = \begin{cases} +\infty \text{ if } \lvert i-j \rvert > T \\ 0 \quad \text{otherwise} \end{cases}$
Weighted DTW [110]	$\Omega(i,j) = f(\lvert i-j \rvert)$ for f increasing function

$\mathcal{L}_{\text{DILATE}^t}$ in Eq 4.3 enables to train deep neural networks with a smooth loss combining shape and temporal criteria. However, $\mathcal{L}_{\text{DILATE}^t}$ presents limited capacities for disentangling the shape and temporal errors, since the optimal path is computed from both shape and temporal terms, i.e. there is no guarantee to recover the true optimal alignment path because of the temporal penalization inside the cost matrix. In contrast, our $\mathcal{L}_{\text{DILATE}}$ loss in Eq 4.1 separates the loss into two shape and temporal components, the temporal penalization being applied to the optimal unconstrained DTW path.

Discussion on most related works We review here the most related works that attempt to train deep forecasting models with alternatives to the MSE. For exploiting the shape of future trajectories, recent works have explored smooth approximations of Dynamic Time Warping (DTW) [55, 171, 1, 256, 17]. Cuturi and Blondel have proposed the soft-DTW [55], which is differentiable loss function that can be computed by dynamic programming with a quadratic complexity. They have shown convincing experiments on time series classification, clustering under the DTW geometry and early experiments on time series forecasting. The soft-DTW was further normalized to ensure a non-negative divergence [17]. However, since DTW is by design invariant to elastic distortions, it completely ignores the temporal localization of the changes. A differentiable timing error loss function based on DTW on the event (binary) space was proposed in [208] ; however it is only applicable for predicting binary time series. Some works explored the use of adversarial losses for time series [289, 279], which can be seen as an implicit way of enforcing semantic criteria learned from data. However, it gives a weaker and non-interpretable control on shape and time criteria and brings additional adversarial training difficulties.

4.3 Experiments

In this section, we evaluate the relevance of DILATE, both quantitatively and qualitatively, compared to generic as well as recent state-of-the-art models trained with the MSE. We also provide an in-depth analysis of the DILATE loss properties.

4.3.1 Datasets

We carry out experiments on 5 synthetic and real-world datasets from various domains to illustrate the broad applicability of our methods. For each dataset, the task is to predict the H-steps ahead future trajectory given a T-steps context window:

- `Synthetic-det` ($T = 20, H = 20$): deterministic dataset consisting in predicting sudden changes (step functions) based on an input signal composed of two peaks. This controlled setup was designed to measure precisely the shape and time errors of predictions. We generate 500 times

series for train, 500 for validation and 500 for test, with 40 time steps each: the first 20 are the inputs, the last 20 are the targets to forecast. In each series, the input range is composed of 2 peaks of random temporal position i_1 and i_2 and random amplitude j_1 and j_2 between 0 and 1, and the target range is composed of a step of amplitude $j_2 - j_1$ and stochastic position $i_2 + (i_2 - i_1) + randint(-3, 3)$. All time series are corrupted by an additive Gaussian white noise of variance 0.01.

- `ECG5000` ($T = 84, H = 56$): this dataset comes from the UCR Time Series Classification Archive [35], and is composed of 5000 electrocardiograms (ECG) (500 for training, 4500 for testing) of length 140. We take the first 84 time steps (60 %) as input and predict the last 56 steps (40 %) of each time series (same setup as in [55]).

- `Traffic` ($T = 168, H = 24$): this dataset is composed of road occupancy rates (between 0 and 1) from the California Department of Transportation (48 months from 2015-2016) measured every 1h. We work on the first univariate series of length 17544 (with the same 60/20/20 train/valid/test split as in [133]), and we train models to predict the 24 future points given the past 168 points (past week)

- `Electricity` ($T = 168, H = 24$): this dataset consists in hourly electricity consumption measurements (kWh) from 370 customers.

- `ETTh1` [300] ($T = 96, H = 96$): dataset of hourly Electricity Transformer Temperature measurements, which is an important indicator for electricity grids. This dataset enables to assess the generalization of our approach on much longer term predictions.

4.3.2 Implementation details

Metrics To evaluate the benefits of our proposed DILATE training loss, we compare it against the widely used Euclidean (MSE) loss, and the soft-DTW introduced in [55, 171]. We use the following multi-step prediction metrics: MSE, DTW (shape), TDI (temporal). To consolidate the evaluation, we also consider two additional (non differentiable) metrics for assessing shape and time. For shape, we compute the ramp score [252]. For time, we compute the Hausdorff distance between a set of detected change points in the target signal $\mathcal{T}^*$ and in the predicted signal $\hat{\mathcal{T}}$:

$$\text{Hausdorff}(\mathcal{T}^*, \hat{\mathcal{T}}) := \max(\max_{\hat{t} \in \hat{\mathcal{T}}} \min_{t^* \in \mathcal{T}^*} |\hat{t} - t^*|, \max_{t^* \in \mathcal{T}^*} \min_{\hat{t} \in \hat{\mathcal{T}}} |\hat{t} - t^*|), \tag{4.4}$$

which corresponds to the largest possible distance between a change point and its prediction. Additional details about these external metrics are given in Appendix B.1.

Neural networks architectures: For the generic neural network architectures, we use a fully connected network (1 layer of 128 neurons), which does not make any assumption on data structure, and a more specialized Seq2Seq model [240] with Gated Recurrent Units (GRU) [39] with 1 layer of 128 units. Each model is trained with PyTorch for a max number of 1000 epochs with Early Stopping with the ADAM optimizer. The smoothing parameter γ of DTW and TDI is set to 10^{-2}.

Table 4.1: **DILATE forecasting results on generic MLP and RNN architectures**, averaged over 10 runs (mean ± standard deviation). Metrics are scaled for readability. For each experiment, best method(s) (Student t-test) in bold.

		Fully connected network (MLP)			**Recurrent neural network (Seq2Seq)**		
Dataset	Train / Eval	MSE	$\text{DTW}_\gamma^{\mathbf{\Delta}}$ [55]	DILATE (ours)	MSE	$\text{DTW}_\gamma^{\mathbf{\Delta}}$ [55]	DILATE (ours)
`Synthetic`	MSE (x1000)	**16.5 ± 1.4**	48.2 ± 4.0	**16.7± 1.8**	**11.0 ± 1.7**	23.1 ± 4.5	**12.1 ± 1.3**
	DTW (x10)	38.6 ± 1.28	**27.3 ± 1.37**	32.1 ± 5.33	**24.6 ± 1.20**	**22.7 ± 3.55**	**23.1 ± 2.44**
	TDI (x10)	15.3 ± 1.39	26.9 ± 4.16	**13.8 ± 0.71**	17.2 ± 1.22	20.0 ± 3.72	**14.8 ± 1.29**
	Ramp (x10)	5.21 ± 0.10	**2.04 ± 0.23**	3.41 ± 0.29	5.80 ± 0.10	**4.27 ± 0.8**	4.99 ± 0.46
	Hausdorff (x1)	4.04 ± 0.28	4.71 ± 0.50	**3.71 ± 0.12**	2.87 ± 0.13	3.45 ± 0.32	**2.70 ± 0.17**
`ECG`	MSE (x100)	**31.5 ± 1.39**	70.9 ± 37.2	37.2 ± 3.59	**21.2 ± 2.24**	75.1 ± 6.30	30.3 ± 4.10
	DTW (x10)	19.5 ± 0.16	18.4 ± 0.75	**17.7 ± 0.43**	17.8 ± 1.62	17.1 ± 0.65	**16.1 ± 0.16**
	TDI (x10)	**7.58 ± 0.19**	17.9 ± 0.7	**7.21 ± 0.89**	8.27 ± 1.03	27.2 ± 11.1	**6.59 ± 0.79**
	Ramp (x1)	**4.9 ± 0.1**	5.1 ± 0.3	**5.0 ± 0.1**	**4.84 ± 0.24**	**4.79 ± 0.37**	**4.80 ± 0.25**
	Hausdorff (x1)	**4.1 ± 0.1**	6.3 ± 0.6	4.7 ± 0.3	**4.32 ± 0.51**	6.16 ± 0.85	**4.23 ± 0.41**
`Traffic`	MSE (x1000)	**6.58 ± 0.11**	25.2 ± 2.3	19.3 ± 0.80	**8.90 ± 1.1**	22.2 ± 2.6	**10.0 ± 2.6**
	DTW (x100)	25.2 ± 0.17	**23.4 ± 5.40**	**23.1 ± 0.41**	24.6 ± 1.85	**22.6 ± 1.34**	**23.0 ± 1.62**
	TDI (x100)	24.8 ± 1.1	27.4 ± 5.01	**16.7 ± 0.51**	**15.4 ± 2.25**	22.3 ± 3.66	**14.4± 1.58**
	Ramp (x10)	6.18 ± 0.1	**5.59 ± 0.1**	**5.6 ± 0.1**	6.29 ± 0.32	**5.78 ± 0.41**	**5.93 ± 0.24**
	Hausdorff (x1)	**1.99 ± 0.2**	**1.91 ± 0.3**	**1.94 ± 0.2**	**2.16 ± 0.38**	**2.29 ± 0.33**	**2.13 ± 0.51**

DILATE hyperparameters: the hyperparameter α balancing $\mathcal{L}_{shape}$ and $\mathcal{L}_{time}$ is determined on a validation set to get comparable DTW shape performance than the $\text{DTW}_\gamma^{\mathbf{\Delta}}$ trained model: $\alpha = 0.5$ for Synthetic and ECG5000, and 0.8 for Traffic, Electricity and ETTh1. The DTW smoothing parameter γ is fixed to 10^{-2}, as further discussed in section 4.3.5.

Our code implementing DILATE is available on line from: `https://github.com/vincent-leguen/DILATE`.

4.3.3 DILATE performances on generic architectures

To demonstrate the broad applicability of our approach, we first perform multi-step forecasting with two generic neural network architectures: a fully connected network (1 layer , which does not make any assumption on data structure, and a more specialized Seq2Seq model with 1 layer of 128 Gated Recurrent Units (GRU). We perform a Student t-test with significance level 0.05 to highlight the best(s) method(s) in each experiment (averaged over 10 runs). Overall results are presented in Table 4.1.

Comparison to MSE training loss: DILATE outperforms MSE when evaluated on shape (DTW) in all experiments, with significant differences on 5/6 experiments. When evaluated on time (TDI), DILATE also performs better in all experiments (significant differences on 3/6 tests). Finally, DILATE is equivalent to MSE when evaluated on MSE on 3/6 experiments.

Comparison to $\text{DTW}_\gamma^{\mathbf{\Delta}}$ training loss: When evaluated on shape (DTW), DILATE performs similarly to $\text{DTW}_\gamma^{\mathbf{\Delta}}$ (2 significant improvements, 1 significant drop and 3 equivalent performances). For time

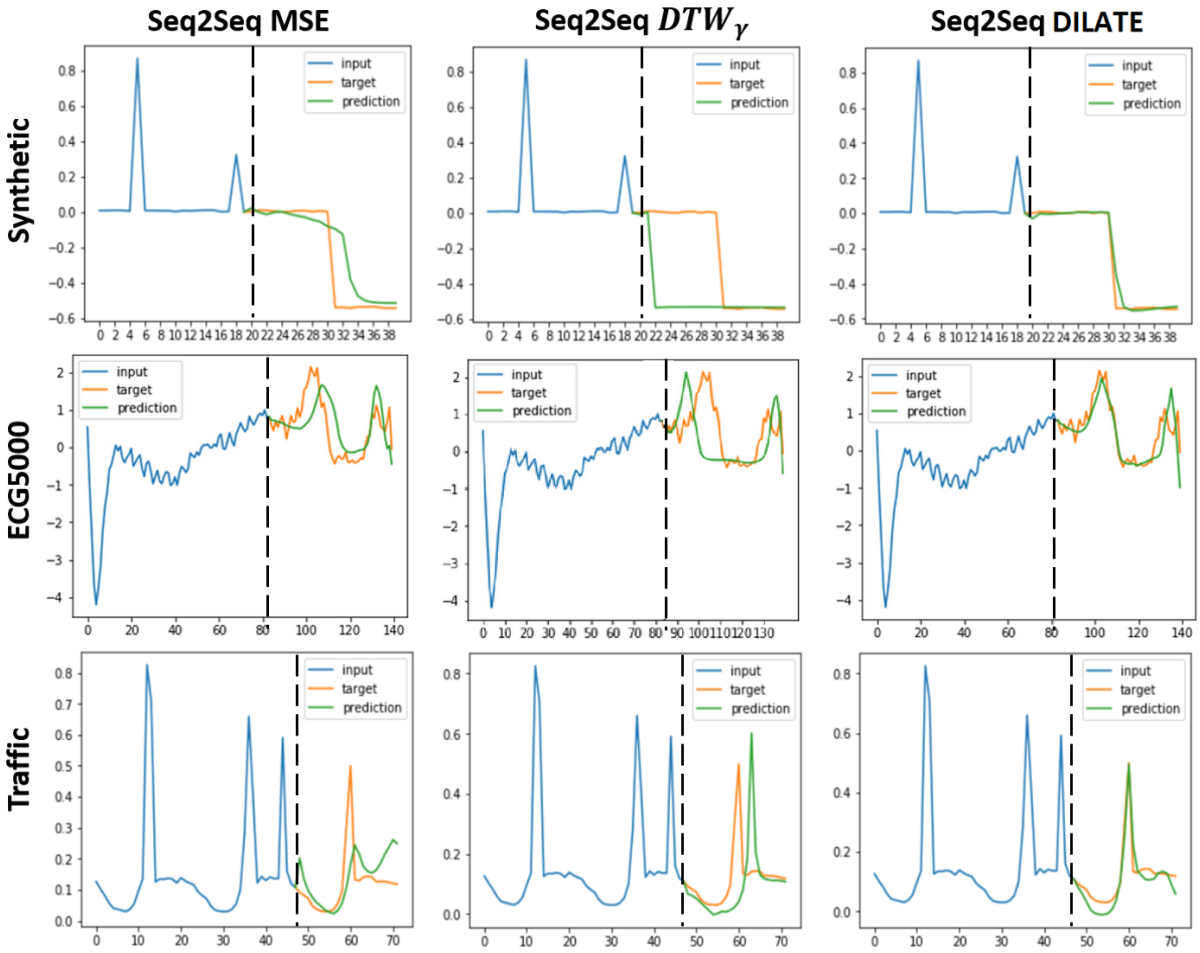


Figure 4.3: **Qualitative prediction results with the DILATE loss.** For each dataset, the MSE training loss leads to non-sharp predictions, whereas the soft-DTW loss can predict sharp variations but has no control over their temporal localization. In contrast, the DILATE loss produces sharp predictions with accurate temporal localization.

(TDI) and MSE evaluations, DILATE is significantly better than $\text{DTW}_\gamma^{\mathbf{\Delta}}$ in all experiments, as expected.

We can notice that the ramp score (resp. the Haussdorff distance) provides the same trends than the shape metric DTW (resp. the time metric TDI). It reinforces our conclusions and shows that DILATE indeed improves shape and temporal accuracy beyond the metrics being optimized.

We display a few qualitative examples for Synthetic, ECG5000 and Traffic datasets in Figure 4.3 (other examples are provided in Appendix B.3). We see that MSE training leads to predictions that are non-sharp, making them inadequate in presence of drops or sharp spikes. $\text{DTW}_\gamma^{\mathbf{\Delta}}$ leads to very sharp predictions in shape, but with a possibly large temporal misalignment. In contrast, our DILATE loss predicts series that have both a correct shape and precise temporal localization.

Table 4.2: **DILATE forecasting results on state-of-the-art architectures N-Beats [183] and Informer [300]**. Evaluation metrics are scaled for readability. Results are averaged over 10 runs, best(s) method(s) in bold (Student t-test).

Dataset	Model	MSE	DTW	Ramp	TDI	Hausdorff	DILATE
`Synthetic`	N-Beats [183] MSE	**13.6 ± 0.5**	24.9 ± 0.6	5.9 ± 0.1	**13.8 ± 1.1**	**2.8 ± 0.1**	**19.3 ± 0.5**
	N-Beats [183] DILATE	**13.3 ± 0.7**	**23.4 ± 0.8**	**4.8 ± 0.4**	**14.4 ± 1.3**	**2.7 ± 0.5**	**18.9 ± 0.8**
	Informer [300] MSE	**10.4 ± 0.3**	20.1 ± 1.1	4.3 ± 0.3	13.1 ± 0.9	**2.5 ± 0.1**	16.6 ± 0.8
	Informer [300] DILATE	11.8 ± 0.7	**18.5 ± 1.2**	**2.4 ± 0.3**	**11.6 ± 0.9**	**2.4 ± 0.9**	**15.1 ± 0.7**
`Electricity`	N-Beats [183] MSE	**24.8 ± 0.4**	**15.6 ± 0.2**	**13.3 ± 0.3**	4.6 ± 0.1	**2.6 ± 0.3**	**13.4 ± 0.2**
	N-Beats [183] DILATE	25.8 ± 0.9	**15.5 ± 0.2**	**13.3 ± 0.3**	**4.4 ± 0.2**	3.1 ± 0.5	**13.2 ± 0.2**
	Informer [300] MSE	**38.1 ± 2.1**	18.9 ± 0.6	13.2 ± 0.2	6.5 ± 0.3	2.1 ± 0.2	16.4 ± 0.5
	Informer [300] DILATE	**37.8 ± 0.8**	**18.5 ± 0.3**	**12.9 ± 0.2**	**5.7 ± 0.2**	**1.9 ± 0.1**	**15.9 ± 0.3**
`ETTH1`	N-Beats [183] MSE	32.5 ± 1.4	3.9 ± 0.2	13.3 ± 2.0	21.6 ± 4.3	**5.7 ± 0.7**	7.4 ± 1.0
	N-Beats [183] DILATE	**26.0 ± 2.8**	**2.9 ± 0.1**	**4.6 ± 0.6**	**11.4 ± 1.7**	**6.4 ± 1.0**	**4.6 ± 0.4**
	Informer [300] MSE	**28.2 ± 2.6**	4.3 ± 0.3	5.8 ± 0.1	21.6 ± 3.3	**6.6 ± 1.9**	7.8 ± 0.9
	Informer [300] DILATE	32.5 ± 3.8	**3.2 ± 0.3**	**4.5 ± 0.3**	**19.1 ± 1.9**	**6.4 ± 1.0**	**6.4 ± 0.6**

4.3.4 DILATE performances with state-of-the-art models

Beyond generic forecasting architectures, we show that DILATE can also improve the performances of state-of-the-art deep architectures. We experiment here with two recent and competitive models: N-Beats [183] and Informer [300]. Results in Table 4.2 are consistent with those in Table 4.1: models trained with DILATE improve over MSE in shape (in DTW and ramp score for 6/6 experiments) and time (in TDI for 5/6 and Hausdorff for 4/6 experiments) and are equivalent to MSE when evaluated in MSE (equivalent or better for 3/6 experiments). We provide qualitative predictions of N-Beats on `Electricity` in Figure 4.4 and `ETTh1` in Figure 4.5. It again confirms that training with DILATE leads to much sharper predictions with a better temporal localization than training with the MSE.

4.3.5 DILATE loss analysis

Influence of α We analyze in Figure 4.6 (a) the influence of the tradeoff parameter α when training a Seq2Seq model on the `Synthetic-det` dataset. When $\alpha = 1$, $\mathcal{L}_{\text{DILATE}}$ reduces to $\text{DTW}_\gamma^{\mathbf{\Delta}}$, with an accurate shape but a large temporal error. When $\alpha \longrightarrow 0$, we only minimize $\mathcal{L}_{time}$ without any shape constraint. Both MSE and shape errors explode in this case, illustrating the fact that $\mathcal{L}_{time}$ is only meaningful in conjunction with $\mathcal{L}_{shape}$. Both the MSE and DILATE error curves present a U-shape ; in this case, $\alpha = 0.5$ seems an acceptable tradeoff for the `Synthetic-det` dataset.

Influence of γ We analyse the influence of the $\text{DTW}_\gamma^{\mathbf{\Delta}}$ smoothing parameter γ in Figure 4.6. We show in Figure 4.6 (c) the assignment probabilities of the $\text{DTW}_\gamma^{\mathbf{\Delta}}$ path between the two test time series from Figure 3.2, the true DTW path being depicted in red. When γ increases, the $\text{DTW}_\gamma^{\mathbf{\Delta}}$ path is more uncertain and becomes multimodal. When $\gamma \to 0$, the soft DTW converges toward the true DTW. However, we see in Figure 4.6 (b) that for small γ values, optimizing $\text{DTW}_\gamma^{\mathbf{\Delta}}$ becomes more difficult, resulting in higher test error and higher variance (on `Synthetic-det`). We fixed $\gamma = 10^{-2}$ in all our experiments, which yields a good tradeoff between an accurate soft optimal path and a low test error.

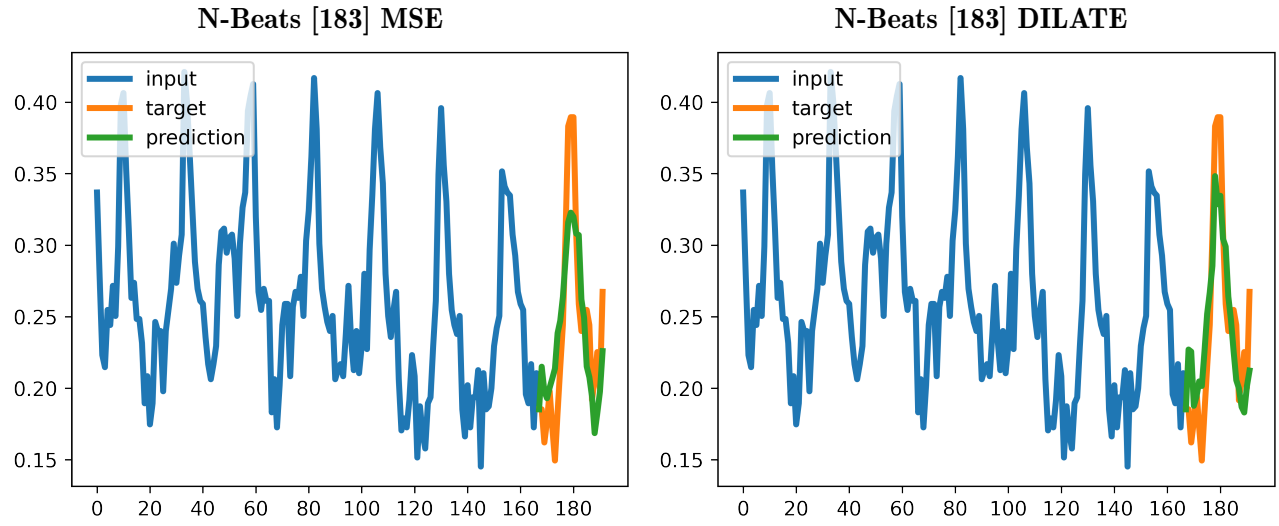


Figure 4.4: Qualitative forecasting results comparing the N-Beats model [183] trained with MSE and the DILATE loss on the `Electricity` dataset.

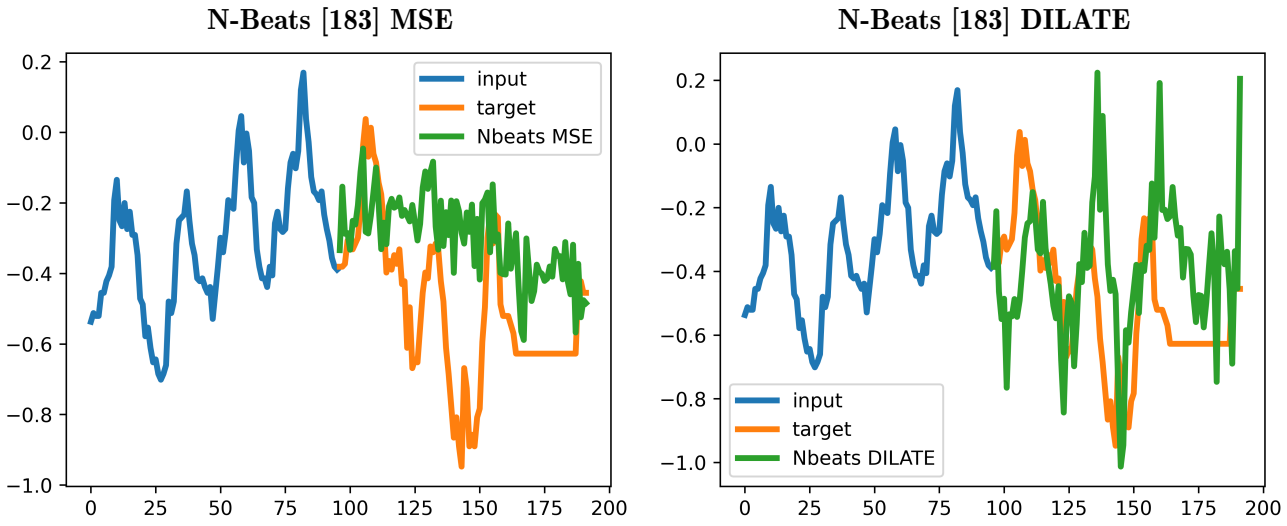
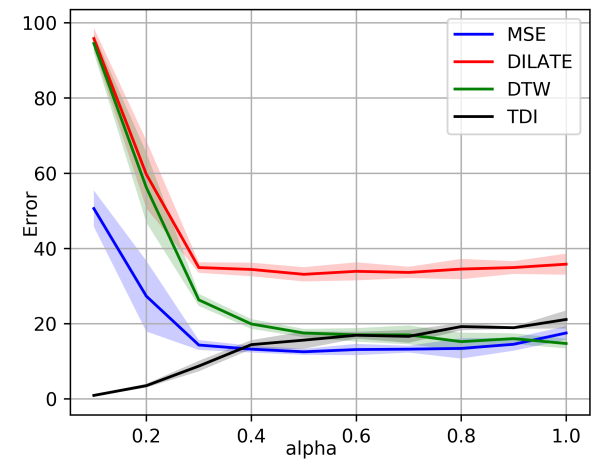
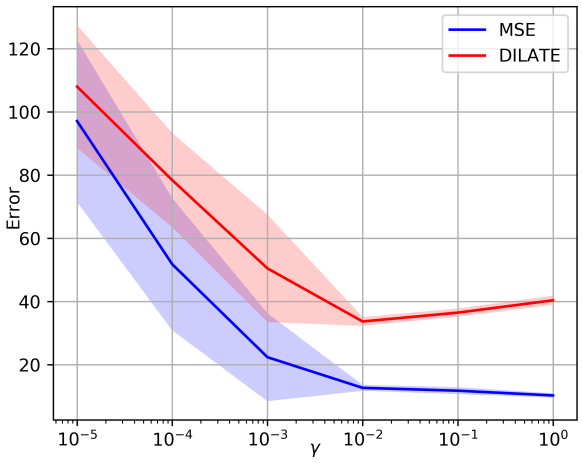


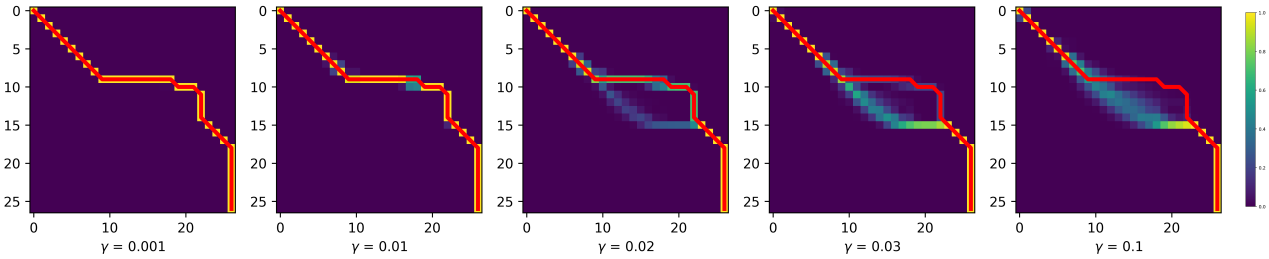
Figure 4.5: Qualitative forecasting results comparing the N-Beats model [183] trained with MSE and the DILATE loss on the `ETTH1` dataset.



(a) Influence of α



(b) Influence of γ



(c) Influence of γ on the soft-DTW optimal path (true path in red)

Figure 4.6: **DILATE loss analysis.** The shaded areas represent ± std computed over 10 runs.

4.4 Conclusion

In this Chapter, we have introduced DILATE, a new differentiable loss function for training deep multi-step time series forecasting models. DILATE combines two terms for precise shape and temporal localization of non-stationary signals with sudden changes. We showed that DILATE is comparable to the standard MSE loss when evaluated on MSE, and far better when evaluated on several shape and timing metrics. DILATE compares favourably on shape and timing to state-of-the-art forecasting algorithms trained with the MSE.

Chapter 5

Probabilistic forecasting with shape and temporal diversity

Content

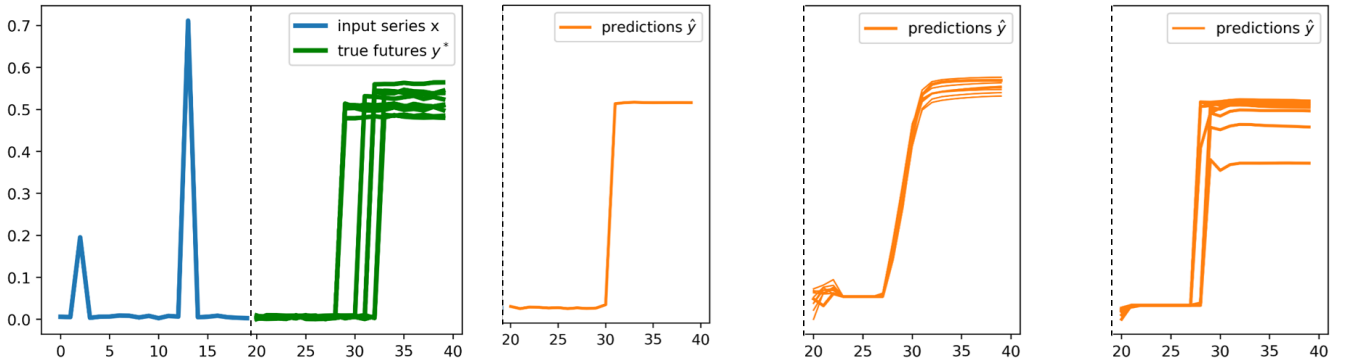
Chapter abstract

In this Chapter, we address the non-stationary time series forecasting problem in the probabilistic setting. To describe the predictive distribution, our goal is to provide a limited set of diverse and accurate scenarios in terms of shape and temporal localization. We introduce the STRIPE forecasting model for representing structured diversity based on shape and time features, ensuring both probable predictions while being sharp and accurate. STRIPE is a forecasting model which outputs multiple predictions by sampling latent variables. STRIPE is equipped with a diversification mechanism relying on determinantal point processes (DPP). Structured diversity is enforced with two shape and temporal semi-definite kernels. We use the two shape and time kernel of Chapter 3, that we prove to be valid PSD kernels, for enforcing structured diversity. Experiments carried out on synthetic datasets show that STRIPE significantly outperforms baseline methods for representing diversity, while maintaining accuracy of the forecasting model. Finally, experiments on real datasets illustrate that STRIPE is able to outperform state-of-the-art probabilistic forecasting approaches in the best sample prediction. The work described in this Chapter is based on the following publications:

- *[140]: Vincent Le Guen and Nicolas Thome. "Probabilistic Time Series Forecasting with Structured Shape and Temporal Diversity". In Advances in Neural Information Processing Systems (NeurIPS 2020).*
- *[141]: Vincent Le Guen and Nicolas Thome. "Deep Time Series Forecasting with Shape and Temporal Criteria". IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.*

Notation: we describe the STRIPE++ model from the publication [141] that we rename STRIPE in this Chapter. It is an improvement over the STRIPE model published in [140].

5.1 Introduction



(a) True predictive distribution (b) DILATE [137] (c) deep stoch model [292] (d) STRIPE (ours)

Figure 5.1: **Probabilistic time series forecasting**: recent advances include the DILATE loss [137] for enabling sharp predictions (b), but are inadequate for producing diverse forecasts. On the other hand, probabilistic forecasting approaches based on generative models [292, 203] loose the ability to generate sharp forecasts (c). The proposed STRIPE model (d) produces both sharp and diverse future forecasts, matching the ground truth distribution (a).

In many applications, producing deterministic forecasts, i.e. a single future trajectory, is not sufficient for decision makers, who need information about the forecast's uncertainty. Probabilistic forecasting consists in modelling the conditional predictive distribution of future trajectories given past values. In this work, our goal is to describe this predictive distribution with a small set (e.g. $N = 10$) of plausible and diverse predictions. This is a different goal than estimating the variance of the predictions or the quantiles of the distribution. Focusing on the non-stationary context with possible sharp variations, the targeted set of predictions should reflect the shape and temporal diversity of the true future trajectories. Our motivation is illustrated in the example of the blue input in Figure 5.1 (a): we aim at performing predictions covering the full distribution of future trajectories, whose samples are shown in green.

State-of-the-art methods for time series forecasting currently rely on deep neural networks, which exhibit strong abilities in modelling complex nonlinear dependencies between variables and time. Recently, increasing attempts have been made for improving architectures for accurate predictions [133, 226**?** , 183, 139] or for making predictions sharper, e.g. by explicitly modelling dynamics [33, 69, 213, 82], or by designing specific loss functions addressing the drawbacks of blurred prediction with

MSE training [55, 208, 137, 256] (e.g. with DILATE). Although Figure 5.1 (b) shows that DILATE produces sharp and realistic forecasts, its deterministic nature leads to to a single trajectory prediction without uncertainty quantification.

Probabilistic methods targeting for producing a diverse set of predictions include generative models [292, 124, 203] that produce multiple trajectories by sampling from a latent space. These approaches are commonly trained using MSE or variants, and consequently often loose the ability to represent sharp predictions, as shown in Figure 5.1 (c) for [292]. These generative models also lack an explicit structure to control the type of diversity in the latent space.

In this Chapter, we introduce the STRIPE model for including Shape and Time diverRsIty in Probabilistic forEcasting. As shown in Figure 5.1 (d), this enables to produce sharp and diverse forecasts, which fit well the ground truth distribution of trajectories in Figure 5.1 (a). STRIPE is a predictive model equipped with a diversification mechanism based on determinantal point processes (DPP). The diversity of predictions is structured with the two shape and temporal semi-definite kernels defined in Chapter 3, and we design explicit schemes to control the quality vs. diversity tradeoff.

We conduct experiments on synthetic datasets to evaluate the ability of STRIPE to match the ground truth trajectory distribution. We show that STRIPE significantly outperforms baseline methods for representing diversity, while maintaining the accuracy of the forecasting model. Experiments on real datasets further show that STRIPE is able to outperform state-of-the-art probabilistic forecasting approaches when evaluating the best sample (i.e. diversity), while being equivalent based on its mean prediction (i.e. quality).

5.2 Related work

In the Section, we pursue the review from Chapter 2 on spatio-temporal forecasting and insist on the most related works for probabilistic forecasting and for imposing structured diversity.

Probabilistic forecasting For describing the conditional distribution of future values given an input sequence, a first class of deterministic methods add variance estimation with Monte Carlo dropout [301, 134] or predict the quantiles of this distribution [275, 86, 274] by minimizing the pinball loss [122, 211] or the continuous ranked probability score (CRPS) [91]. Other probabilistic methods try to approximate the predictive distribution, *explicitly* with a parametric distribution (e.g. Gaussian for DeepAR [219] and variants [202, 218]), or *implicitly* with a generative model with latent variables (e.g. with conditional variational autoencoders (cVAEs) [292], conditional generative adversarial networks (cGANs) [124], normalizing flows [203]). However, these methods lack the ability to produce sharp forecasts by minimizing variants of the MSE (pinball loss, gaussian maximum likelihood), at the exception of cGANs - but which suffer from mode collapse that limits predictive diversity. Moreover, these generative models are generally represented by unstructured distributions in the latent space (e.g. Gaussian), which do not allow to have an explicit control on the targeted diversity.

Structured diversity for prediction For diversifying forecasts, several repulsive schemes were studied such as the variety loss [97, 244] that consists in optimizing the best sample, or entropy regularization [63, 263] that encourages a uniform distribution. Besides, generative models, such as variational autoencoders (VAE) [119], are widely used for producing multiple predictions through sampling from

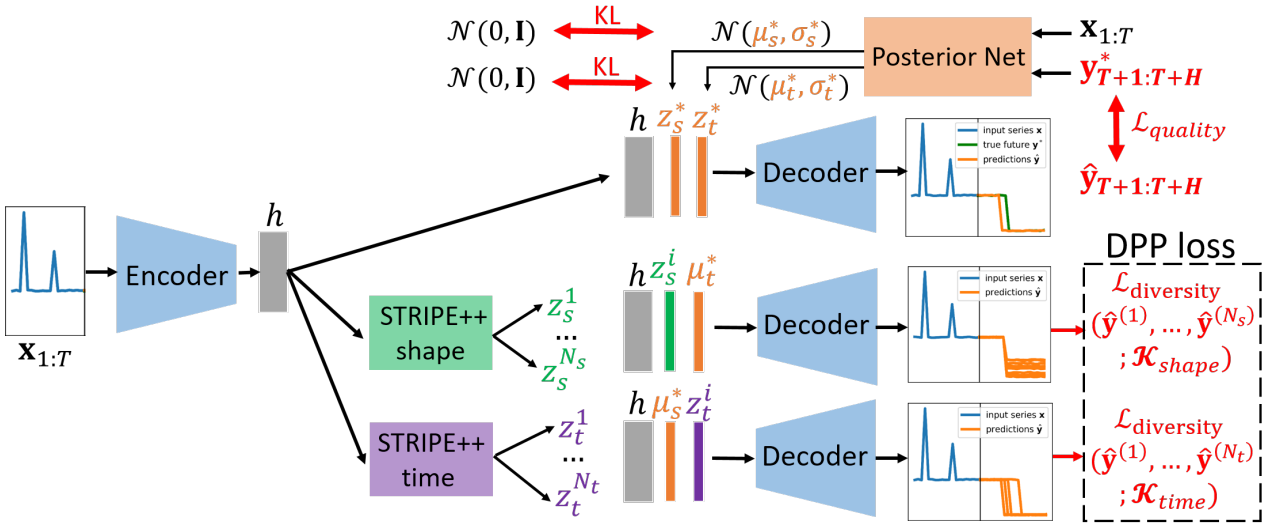


Figure 5.2: **Overview of the STRIPE model:** STRIPE builds on a forecasting architecture trained with a quality loss $\mathcal{L}_{quality}$ enforcing sharp predictions. The latent state is disentangled into a deterministic part h from the encoder and two stochastic codes z_s and z_t that account for the shape and time variations. First step (Figure upper part), we train the predictor with a quality loss, the stochastic codes are sampled from a posterior network. Second step (bottom), we diversify the predictions with two STRIPE shape and time proposal networks trained with a DPP diversity loss (keeping the encoder and decoder frozen).

a latent space. However latent states are typically sampled at test time from a standard Gaussian prior distribution, resulting in an unstructured diversity. To improve this unstructured mechanism, prior works [292, 293] introduced proposal neural networks for generating the latent variables that are trained with a diversity objective.

As discussed in Chapter 2, determinantal point processes (DPPs) are an appealing mathematical solution for characterizing the diversity of a set of items. Efficient algorithms maximizing the diversity of a set of items with a given sampling budget. GDPP [71] proposed by Elfeki *et al.* is based on matching generated and true sample diversity by aligning the corresponding DPP kernels, and thus limits their use in datasets where the full distribution of possible outcomes is accessible. In contrast, our probabilistic forecasting approach is applicable in realistic scenarios where only a single future trajectory is available for each training sample. Yuan and Kitani [292] train their proposal neural networks with a DPP diversity loss. Although we share with [292] the goal to use DPP as diversification mechanism for future trajectories, the main limitation in [292] is to use the MSE loss for training the predictor and the MSE kernel for diversification, leading to blurred prediction, as illustrated in Figure 5.1 (c). In contrast, we design specific shape and time DPP kernels and we show the necessity to decouple the criteria used for quality and diversity.

5.3 Probabilistic forecasting with structured diversity

We consider the multi-step and non-stationary time series forecasting problem in the probabilistic case. Given an input sequence $\mathbf{x}_{1:T} = (\mathbf{x}_1, \dots, \mathbf{x}_T) \in \mathbb{R}^{p \times T}$, we aim at describing the conditional predictive distribution of future trajectories with a set of N future trajectories $\{\hat{\mathbf{y}}^{(i)}\}_{i=1..N} \in \mathbb{R}^{d \times H}$ (corresponding to diverse scenarii sampled from the true future distribution $\hat{\mathbf{y}}^{(i)} \sim p(\cdot|\mathbf{x}_{1:T})$).

We introduce the STRIPE framework (Shape and Time diverRsIty in Probabilistic forEcasting), that extends STRIPE [140]. Depicted in Figure 5.2, STRIPE builds upon a general multi-step forecasting pipeline: the input time series $\mathbf{x}_{1:T}$ is fed into an encoder that summarizes the input into a latent vector h. This context vector h is then transformed by a decoder into a future trajectory.

The key idea of STRIPE is to augment the deterministic latent state h with stochastic diversifying variables z_s (resp. z_t) meant to capture the shape (resp. temporal) variations of the future time series. We distinguish two phases for training the overall model: (i) we train the predictor with a quality loss and (ii) we train the diversifying STRIPE mechanism with a DPP diversity loss (with the weights of the predictor frozen). For both of these steps, we detail now how the diversifying variables are sampled.

5.3.1 Training the predictor with a quality loss

For training the predictor (upper part in Figure 5.2) with possibly multiple admissible futures as supervision, we take inspiration from the probabilistic U-Net [123] and introduce a posterior network from which to sample the diversifying variables z_s^* and z_t^* (which represent the shape and temporal variant attached to a particular future $\mathbf{y}^*$). The posterior net outputs the parameters μ_s^* and σ_s^* of a Gaussian distribution $\mathcal{N}(\mu_s^*, \sigma_s^*)$ for parameterizing the shape posterior distribution $q(z_s|\mathbf{x}, \mathbf{y}^*)$ (and similarly for the temporal posterior distribution).

To train this generative model (encoder, decoder and posterior networks), we resort to variational inference [119] and maximize the evidence lower bound (ELBO) of the log-likelihood, or equivalently, minimize the following prediction loss over all training examples:

$$\mathcal{L}_{prediction}(\hat{\mathbf{y}}, \mathbf{y}^*) = \mathcal{L}_{quality}(\hat{\mathbf{y}}, \mathbf{y}^*) \; + \mathrm{KL}\left(q(z_s|\mathbf{x}, \mathbf{y}^*) \;||\; p(z_s)\right) + \mathrm{KL}\left(q(z_t|\mathbf{x}, \mathbf{y}^*) \;||\; p(z_t)\right). \tag{5.1}$$

In our non-stationary context, we choose the DILATE loss for $\mathcal{L}_{quality}$, in order to guarantee sharp predictions with accurate temporal localization. The Kullback-Leibler (KL) losses enforce that the shape posterior distribution $q(z_s|\mathbf{x}, \mathbf{y}^*)$ matches a prior distribution $p(z_s)$ (we use a Gaussian prior $\mathcal{N}(0, \mathbf{I})$, which is a common choice in variational inference).

5.3.2 Training the STRIPE diversification mechanism

For including structured shape and temporal diversity (lower part in Figure 5.2), we introduce two proposal neural networks $\text{STRIPE}_{\text{shape}}$ and $\text{STRIPE}_{\text{time}}$ that aim to produce a set of N_s shape latent codes $\{z_s^i\}_{i=1..N_s} \in \mathbb{R}^k$ (resp. N_t time codes $\{z_t^i\}_{i=1..N_t} \in \mathbb{R}^k$) dedicated to generate diverse trajectories in terms of shape (resp. time).

When training $\text{STRIPE}_{\text{shape}}$ (the description for $\text{STRIPE}_{\text{time}}$ is similar), we concatenate h with the posterior time latent code μ_t^* and the N_s shape latent codes z_s^i provided by $\text{STRIPE}_{\text{shape}}$, which

leads to N_s future trajectories $\hat{\mathbf{y}}^i = \text{Decoder}\left((h, z_s^i, \mu_t^*)\right)$, $i = 1..N_s$[1]. The shape diversity of this set of N_s trajectories is then enforced by a shape diversity loss that we describe below.

DPP diversity loss: We resort to determinantal point processes (DPP) for their appealing properties for maximizing the diversity of a set of items $\mathcal{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_N\}$ given a fixed sampling budget N and for structuring diversity via the choice of the DPP kernel. Following [292], we minimize the negative expected cardinality of a random subset Y from the DPP:

$$\mathcal{L}_{diversity}(\mathcal{Y}; \mathbf{K}) = -\mathbb{E}_{Y \sim \text{DPP}(\mathcal{K})}|Y| \tag{5.2}$$

$$= -Tr(\mathbf{I} - (\mathbf{K} + \mathbf{I})^{-1}). \tag{5.3}$$

Intuitively, a larger expected cardinality means a more diverse sampled set according to kernel $\mathcal{K}$. This loss is differentiable and can be computed in closed form.

Quality regularizer in the DPP: When training the shape and time proposal networks with the diversity loss, we do not have control over the quality of predictions, which can deteriorate to improve diversity. To address this, we introduce a quality regularization term in the DPP kernels. Crucially, we decouple the criteria used for quality (DILATE) and diversity (shape or time). $\mathcal{K}_{shape}$ maximizes the shape (DTW) diversity, while maintaining a globally low DILATE loss (thus playing on the temporal localization to ensure a good tradeoff). This contrasts with [292] which uses the same MSE criterion for both quality and diversity (see Figure 5.4 (b) for a detailed analysis). In practice, we introduce a quality vector $\mathbf{q} = (q_1, \ldots, q_{N_s})$ between the prediction $\hat{\mathbf{y}}^i$ and the ground truth $\mathbf{y}^*$ [2]. We choose $q_i = \mu(1 - \text{DILATE}(\hat{\mathbf{y}}^i, \mathbf{y}^*))$, where $\mu > 0$ is a hyperparameter to tune the influence of the quality regularization. The modified shape kernel becomes (and similarly for the time kernel):

$$\tilde{\mathbf{K}}_{shape} = \text{Diag}(\mathbf{q})\ \mathbf{K}_{shape}\ \text{Diag}(\mathbf{q}). \tag{5.4}$$

This decomposition enables to sample sets of items of both high quality and diversity:

$$\mathcal{P}_{\tilde{\mathbf{K}}}(\mathbf{Y} = Y) \propto \left(\prod_{i \in Y} q_i^2\right) \det(\mathbf{K}_Y). \tag{5.5}$$

We then train $\text{STRIPE}_{\text{shape}}$ by applying the shape kernel $\tilde{\mathbf{K}}_{shape}$ (Eq 5.4) to the set of N_s shape future trajectories $\mathcal{L}_{diversity}(\hat{\mathbf{y}}^1, \ldots, \hat{\mathbf{y}}^{N_s}; \tilde{\mathbf{K}}_{shape})$ and $\text{STRIPE}_{\text{time}}$ by applying the time kernel $\tilde{\mathbf{K}}_{time}$ to the set of N_t time future trajectories $\mathcal{L}_{diversity}(\hat{\mathbf{y}}^1, \ldots, \hat{\mathbf{y}}^{N_t}; \tilde{\mathbf{K}}_{time})$.

[1] If there exists multiple futures as supervision, we repeat this operation for each posterior latent code $\mu_t^{*,j}$ (it corresponds to consider each tuple $(\mathbf{x}_{1:T}, \mathbf{y}^{*,j})$ as a separate training example).

[2] If there are multiple futures as supervision, we again consider each tuple (input sequence, possible future) as a separate training example.

5.3.3 Diverse trajectory generation at test time

At test time, the posterior network is discarded and we only rely on the trained encoder, $\text{STRIPE}_{\text{shape}}$, $\text{STRIPE}_{\text{time}}$ proposal networks and decoder to generate future predictions. More precisely, we combine the shape and temporal proposals $\{z_s^i\}_{i=1..N_s}$ and $\left\{z_t^j\right\}_{j=1..N_t}$ to obtain $N_s \times N_t$ predictions $\hat{\mathbf{y}}^{i,j} = \text{Decoder}((h, z_s^i, z_t^j))$.

5.4 Experiments

We firstly assess the ability of STRIPE to capture the full predictive distribution of future trajectories. To do so, we need for evaluation the ground truth set of admissible futures for a given input; we construct here the `Synthetic-prob` dataset designed for this purpose. Secondly, on a more realistic setting where we only know one future for each input, we evaluate STRIPE on the `Traffic` and `Electricity` datasets with the best (resp. the mean) sample metrics as a proxy for diversity (resp. quality). We describe the implementation details and neural network architectures (encoder, decoder, posterior net and STRIPE proposal network) in Appendix C.1.

5.4.1 Full predictive distribution evaluation on `Synthetic-prob`

Dataset: In this Chapter, we build the `Synthetic-prob` ($T = 20, H = 20$) dataset with multiple admissible futures for each input series. This is a variant of `Synthetic-det` used in Chapter 4 where for each input series, we generate 10 different future series of length 20 by adding noise on the step amplitude and localization. A sample from this dataset can be observed in Figure 5.1 (a). The dataset is composed of $100 \times 10 = 1000$ time series for each train/valid/test split.

Metrics: To assess the discrepancy between the predicted and true distributions of futures trajectories, we define the two following measures $\text{H}_{quality}(\ell)$ and $\text{H}_{diversity}(\ell)$ (ℓ = DTW, TDI or DILATE in our experiments):

$$\text{H}_{quality}(\ell) := \mathbb{E}_{\mathbf{x}\in\mathcal{D}_{test}} \mathbb{E}_{\hat{\mathbf{y}}} \left[\inf_{\mathbf{y}\in F(\mathbf{x})} \ell(\hat{\mathbf{y}}, \mathbf{y}) \right] \tag{5.6}$$

$$\text{H}_{diversity}(\ell) := \mathbb{E}_{\mathbf{x}\in\mathcal{D}_{test}} \mathbb{E}_{\mathbf{y}\in F(\mathbf{x})} \left[\inf_{\hat{\mathbf{y}}} \ell(\hat{\mathbf{y}}, \mathbf{y}) \right]. \tag{5.7}$$

$\text{H}_{quality}$ penalizes forecasts $\hat{\mathbf{y}}$ that are far away from a ground truth future of $\mathbf{x}$ denoted $\mathbf{y} \in F(\mathbf{x})$ (similarly to the *precision* concept in pattern recognition) whereas $\text{H}_{diversity}$ penalizes when a true future is not covered by a forecast (similarly to *recall*). As a tradeoff balancing quality and diversity, we compute the F1 score defined in Eq 5.8:

$$F1 \text{ score} = \frac{2\ \text{H}_{quality}(\ell) \cdot \text{H}_{diversity}(\ell)}{\text{H}_{quality}(\ell) + \text{H}_{diversity}(\ell)}. \tag{5.8}$$

In addition, we also use the continuous ranked probability score (CRPS) which is a standard *proper scoring rule* [91] for assessing probabilistic forecasts [86]. Intuitively, the CRPS is the pinball loss

Table 5.1: **STRIPE forecasting results on the `Synthetic-prob` dataset with multiple futures**, averaged over 5 runs (mean ± std). Best equivalent methods (Student t-test) shown in bold. Metrics are scaled (MSE × 1000, DILATE ×100, CRPS × 1000).

Methods	$\mathrm{H}_{quality}(\cdot)$ (↓)			$\mathrm{H}_{diversity}(\cdot)$ (↓)			$F1$ score (↓)			CRPS (↓)
	DTW	TDI	DILATE	DTW	TDI	DILATE	DTW	TDI	DILATE	
cVAE DILATE	**11.7 ± 1.5**	9.4 ± 2.2	**14.2 ± 1.5**	18.8 ± 1.3	48.6 ± 2.2	33.9 ± 3.9	14.4	15.7	20.0	62.2 ± 4.2
variety loss [244] DILATE	15.6 ± 3.4	10.2 ± 1.1	16.8 ± 0.9	22.7 ± 4.1	37.7 ± 4.9	30.8 ± 1.0	18.5	16.1	21.7	62.6 ± 3.0
Entropy reg. [63] DILATE	13.8 ± 3.1	8.8 ± 2.2	**15.0 ± 1.6**	20.4 ± 2.8	42.0 ± 7.8	32.6 ± 2.3	16.5	14.5	20.5	62.4 ± 3.9
Diverse DPP [292] DILATE	**12.9 ± 1.2**	9.8 ± 2.1	15.1 ± 1.5	18.6 ± 1.6	42.8 ± 10.1	31.3 ± 5.7	15.2	15.9	20.4	60.7 ± 1.6
GDPP [71] DILATE	14.8 ± 2.9	11.7 ± 8.4	**14.4 ± 2.1**	20.8 ± 2.4	25.2 ± 7.2	23.9 ± 4.5	17.3	15.9	17.9	63.4 ± 6.4
STRIPE	13.5 ± 0.5	9.2 ± 0.5	**15.0 ± 0.3**	**12.9 ± 0.3**	16.3 ± 1.2	**17.9 ± 0.6**	**13.2**	11.7	**16.3**	**48.6 ± 0.6**

integrated over all quantile levels. A key property is that the CRPS attains its minimum when the predicted future distribution equals the true future distribution, making this metric particularly adapted to our context.

Forecasting results: We compare in Table 5.1 our method to 4 recent competing diversification mechanisms (variety loss [244], entropy regularisation [63], diverse DPP [292] and GDPP [71]) based on a conditional variational autoencoder (cVAE) backbone trained with DILATE. We observe that STRIPE obtains the global best performances by improving diversity by a large amount ($\mathrm{H}_{diversity}$(DILATE)=17.9) compared to the backbone cVAE DILATE ($\mathrm{H}_{diversity}$(DILATE)=33.9) and to other diversification schemes (the best competitor GDPP [71] attains $\mathrm{H}_{diversity}$(DILATE)=23.9). This highlights the relevance of the structured shape and time diversity. We can also notice that, in contrast to competing diversification schemes that improve diversity at the cost of a loss in quality, STRIPE maintains high quality predictions. STRIPE is only beaten in $\mathrm{H}_{quality}$(DILATE) by GDPP [71], but this method is significantly worse than STRIPE in diversity, and GDPP requires full future distribution supervision, which it not applicable in real datasets (see section 5.4.2). All in all, the F1 scores summarize the quality vs. diversity tradeoffs, and STRIPE gets the best F1 DILATE score. Moreover, STRIPE outperforms all other methods with the CRPS metric, indicating that the predicted future trajectory distribution is closer to the ground truth one.

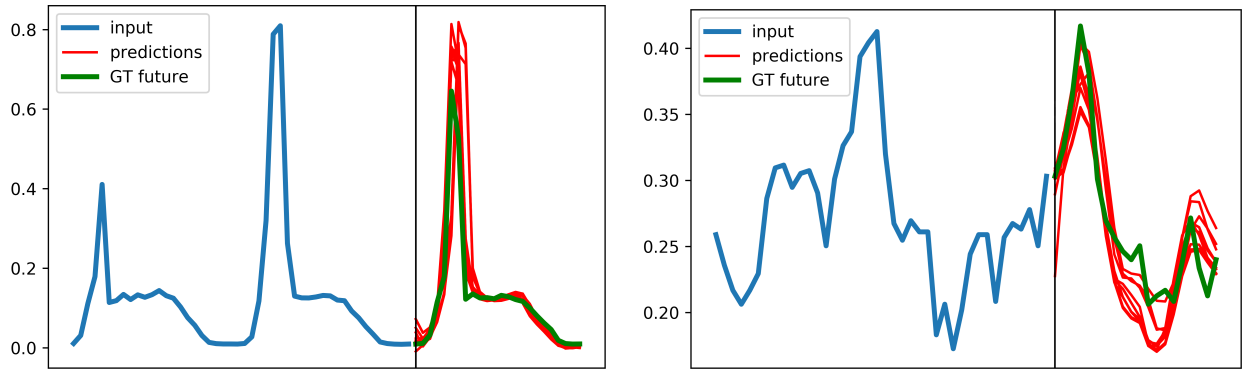
5.4.2 State-of-the-art comparison on real-world datasets

We evaluate here the performances of STRIPE on the two challenging real-world datasets `Traffic` and `Electricity` commonly used as benchmarks in the time series forecasting literature [290, 219, 133, 202, 137, 226] and described in Chapter 4. Contrary to the `Synthetic-prob` dataset, we only dispose of one future trajectory sample $\mathbf{y}^*_{T+1:T+\tau}$ for each input series $\mathbf{x}_{1:T}$. In this case, the metric $\mathrm{H}_{quality}$ (resp. $\mathrm{H}_{diversity}$) defined in section 5.4.1 reduces to the mean sample (resp. best sample), which are common for evaluating stochastic forecasting models [6, 82].

Results in Table 5.2 reveal that STRIPE outperforms all other baselines in the best sample (evaluated in MSE or DILATE). Our method even outperforms in the best sample the state-of-the-art N-Beats algorithm [183] (either trained with MSE or DILATE), which is dedicated to producing high quality deterministic forecasts. In terms of quality (evaluation with the mean sample), STRIPE gets the best (or equivalently best) results in all cases. This contrasts to competing diversification methods, e.g. Diverse DPP [292], that deteriorate quality to improve diversity. Finally we notice that STRIPE is

Table 5.2: **Probabilistic forecasting results on the `Traffic` and `Electricity` datasets**, averaged over 5 runs (mean ± std). Metrics are scaled for readability. Best equivalent method(s) (Student t-test) shown in bold.

	`Traffic`				`Electricity`			
	MSE		DILATE		MSE		DILATE	
Method	mean	best	mean	best	mean	best	mean	best
Nbeats [183] MSE	-	7.8 ± 0.3	-	22.1 ± 0.8	-	24.8 ± 0.4	-	20.2 ± 0.3
Nbeats [183] DILATE	-	17.1 ± 0.8	-	17.8 ± 0.3	-	25.8 ± 0.9	-	19.9 ± 0.5
Deep AR [219]	15.1 ± 1.7	**6.6 ± 0.7**	30.3 ± 1.9	16.9 ± 0.6	67.6 ± 5.1	25.6 ± 0.4	59.8 ± 5.2	17.2 ± 0.3
cVAE DILATE	**10.0 ± 1.7**	8.8 ± 1.6	**19.1 ± 1.2**	17.0 ± 1.1	**28.9 ± 0.8**	27.8 ± 0.8	24.6 ± 1.4	22.4 ± 1.3
Variety loss [244]	**9.8 ± 0.8**	7.9 ± 0.8	**18.9 ± 1.4**	15.9 ± 1.2	29.4 ± 1.0	27.7 ± 1.0	24.7 ± 1.1	21.6 ± 1.0
Entropy regul. [63]	11.4 ± 1.3	10.3 ± 1.4	**19.1 ± 1.4**	16.8 ± 1.3	34.4 ± 4.1	32.9 ± 3.8	29.8 ± 3.6	25.6 ± 3.1
Diverse DPP [292]	11.2 ± 1.8	6.9 ± 1.0	20.5 ± 1.0	14.7 ± 1.0	31.5 ± 0.8	25.8 ± 1.3	26.6 ± 1.0	19.4 ± 1.0
STRIPE	**10.0 ± 0.2**	**6.7 ± 0.3**	**19.0 ± 0.2**	**14.1 ± 0.3**	**29.5 ± 0.3**	**23.6 ± 0.4**	**24.1 ± 0.2**	17.3 ± 0.4



(a) `Traffic` (b) `Electricity`

Figure 5.3: STRIPE qualitative predictions on datasets `Traffic` (a) and `Electricity` (b).

consistently better in diversity and quality than the state-of-the art probabilistic deep AR method [219].

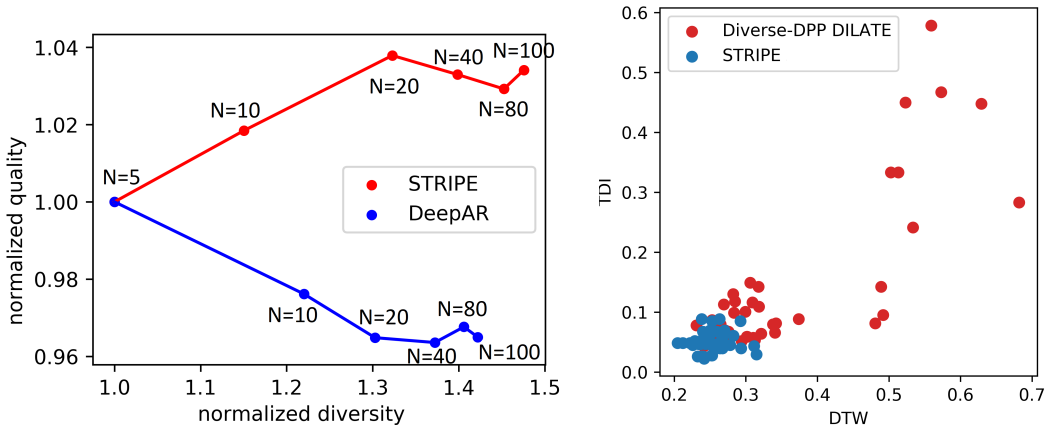
We display a few qualitative forecasting examples of STRIPE on Figure 5.3. We observe that STRIPE predictions are both sharp and accurate: both the shape diversity (amplitude of the peaks) and temporal diversity match the ground truth future.

5.4.2.1 STRIPE analysis: quality-diversity cooperation

We analyze here the quality-diversity tradeoff with respect to the number N of sampled future trajectories. In Figure 5.4 (a) we represent the evolution of performances when N increases from 5 to 100 on the synthetic-prob dataset. As expected, the normalized DILATE diversity $\mathrm{H}_{diversity}(5)/\mathrm{H}_{diversity}(N)$ (higher is better) increases with N for both STRIPE and deepAR models [219]. However we remark that STRIPE does not deteriorate normalized quality (which even increases slightly), in contrast to deepAR which does not have control over the targeted diversity. This again confirms the relevance of our approach that effectively combines an adequate quality loss function and a structured diversity

mechanism.

We also highlight the importance to separate the criteria for enforcing quality and diversity. In Figure 5.4, we represent 50 predictions from the models Diverse DPP DILATE [292] and STRIPE in the plane (DTW,TDI). Diverse DPP DILATE [292] uses a DPP diversity loss based on the DILATE kernel, which is the same than for quality. We clearly see that the two objectives conflict: this model increases the DILATE diversity (by increasing the variance in the shape (DTW) or the time TDI) components) but a lot of these predictions have a high DILATE loss (worse quality). In contrast, STRIPE predictions are diverse in DTW and TDI, and maintain an overall low DILATE loss. STRIPE succeeds in recovering a set of good tradeoffs between shape and time leading a low DILATE loss.



(a) (b)

Figure 5.4: **STRIPE analysis:** (a) Influence of the number N of trajectories on quality (higher is better) and diversity for the `Synthetic-prob` dataset. (b) Scatterplot of 50 predictions in the plane (DTW,TDI), comparing STRIPE v.s. Diverse DPP DILATE [292].

5.5 Conclusion

In this Chapter, we have presented STRIPE, a probabilistic time series forecasting method that introduces structured shape and temporal diversity based on determinantal point processes. Diversity is controlled via two proposed differentiable positive semi-definite kernels for shape and time and exploits a forecasting model with a disentangled latent space. Experiments on synthetic and real-world datasets confirm that STRIPE leads to more diverse forecasts without sacrificing on quality. Ablation studies also reveal the crucial importance to decouple the criteria used for quality and diversity.

Part II

Physics-informed forecasting with incomplete knowledge

Abstract

In this part, we are interested in designing Machine Learning (ML) / Model-Based (MB) augmented models by leveraging incomplete physical knowledge formalized through ODE/PDE. Since physical laws are often not directly applicable at the pixel level nor be sufficient for predicting the whole content of future images in generic videos, we propose to learn a latent space where we suppose that physical dynamics apply. We introduce the PhyDNet model (Chapter 6), which is a two-branch recurrent neural network. One branch is responsible for modelling the physical dynamics while the other branch captures the complementary information required for accurate prediction. We show that PhyDNet reaches state-of-the-art performances on several video prediction benchmarks. Going further, we concentrate on the ML/MB decomposition problem discussed in Chapter 1, which is ill-posed and admits an infinity of solutions. We introduce a principled learning framework, called APHYNITY (Chapter 7). Inspired by the least-action principle, APHYNITY minimizes the norm of the data-driven complement under the constraint of perfect prediction of the augmented model. We provide a theoretical analysis of the decomposition and show that we can ensure existence and uniqueness decomposition guarantees, under mild conditions. We show on several challenging physical dynamics that APHYNITY ensures better forecasting and parameter identification performances than MB or ML models alone, and that competing MB/ML hybrid methods.

Chapter 6

Disentangling physical from residual dynamics for video prediction

Content

Chapter abstract

In this Chapter, we address the video prediction problem with deep learning. To constrain the challenging generation of high-dimensional images at the pixel level, we propose to incorporate physical knowledge described by partial differential equations (PDEs). However, since physics is too restrictive for describing the full visual content of generic videos, we introduce in this Chapter PhyDNet, a two-branch deep architecture, which disentangles PDE dynamics from unknown complementary information. The physical branch is composed of a new recurrent physical cell (PhyCell), inspired from data assimilation techniques, that performs PDE-constrained prediction in latent space. Extensive experiments conducted on four various datasets show the very good performances reached by PhyDNet. Ablation studies also highlight the important gain brought out by both disentanglement and PDE-constrained prediction. Finally, we show that PhyDNet presents interesting features for dealing with missing data and long-term forecasting.
The work described in this Chapter is based on the following publication:

- *[139]: Vincent Le Guen and Nicolas Thome. "Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020).*

6.1 Introduction

Video forecasting consists in predicting the future content of a video conditioned on previous frames. This is of crucial importance in various contexts, such as weather forecasting [281], autonomous driving [132], reinforcement learning [181], robotics [77], or action recognition [152]. In this work, we focus on unsupervised video prediction, where the absence of semantic labels to drive predictions exacerbates the challenges of the task. In this context, a key problem is to design video prediction methods able to represent the complex dynamics underlying raw data.

State-of-the-art methods for training such complex dynamical models currently rely on deep learning, with specific architectural choices based on 2D/3D convolutional [166, 261] or recurrent neural networks [268, 266, 269]. To improve predictions, recent methods use adversarial training [166, 261, 132], stochastic models [28, 173, 82], constraint predictions by using geometric knowledge [77, 112, 283] or by disentangling factors of variation [258, 250, 60, 104].

Another appealing way to model the video dynamics is to exploit prior physical knowledge, e.g. formalized by partial differential equations (PDEs) [58, 227]. Recently, interesting connections between residual networks and PDEs have been drawn [273, 158, 33], enabling to design physically-constrained machine learning frameworks [200, 58, 227, 214]. These approaches are very successful for modelling physical systems, when the underlying dynamics is well described by the physical equations in the input space [200, 214, 155]. However, such assumption is rarely fulfilled in the pixel space for predicting generalist videos.

In this work, we introduce PhyDNet, a deep model dedicated to perform accurate future frame predictions from generalist videos. In such a context, physical laws do not apply in the input pixel space; the goal of PhyDNet is to learn a semantic latent space $\boldsymbol{\mathcal{H}}$ in which they do, and are disentangled from other factors of variation required to perform future prediction. Prediction results of PhyDNet when trained on Moving MNIST [236] are shown in Figure 6.1. The left branch represents the physical dynamics in $\boldsymbol{\mathcal{H}}$; when decoded in the image space, we can see that the corresponding features encode approximate segmentation masks predicting digit positions on subsequent frames. On the other hand, the right branch extracts residual information required for prediction, here the precise appearance of the two digits. Combining both representations eventually makes accurate prediction successful.

Our contributions to the unsupervised video prediction problem with PhyDNet can be summarized as follows:

- We introduce a global sequence to sequence two-branch deep model (section 6.3.1) dedicated to jointly learn the latent space $\boldsymbol{\mathcal{H}}$ and to disentangle physical dynamics from residual information, the latter being modeled by a data-driven (ConvLSTM [281]) method.
- Physical dynamics is modelled by a new recurrent physical cell, PhyCell (section 6.3.2), discretizing a broad class of PDEs in $\boldsymbol{\mathcal{H}}$. PhyCell is based on a prediction-correction paradigm inspired from the data assimilation community [3], enabling robust training with missing data and for long-term forecasting.

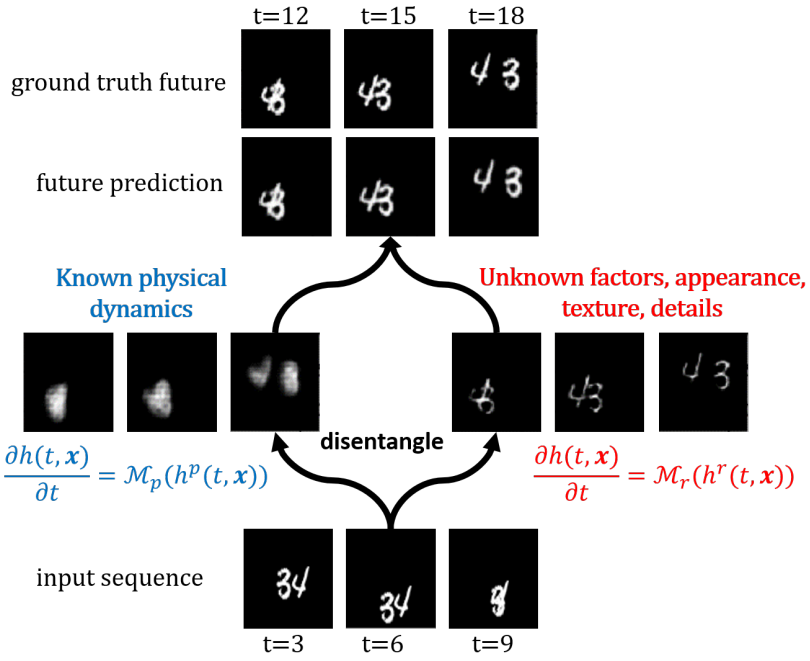


Figure 6.1: PhyDNet is a deep model mapping an input video into a latent space $\boldsymbol{\mathcal{H}}$, from which future frame prediction can be accurately performed. PhyDNet learns $\boldsymbol{\mathcal{H}}$ in an unsupervised manner, such that physical dynamics and unknown factors necessary for prediction, e.g. appearance, details, texture, are disentangled.

- Experiments (section 6.4) reveal that PhyDNet outperforms state-of-the-art methods on four generalist datasets: this is, as far as we know, the first physically-constrained model able to show such capabilities. We highlight the importance of both disentanglement and physical prediction for optimal performances.

6.2 Related work

We review here related multi-step video prediction approaches dedicated to long-term forecasting. We also focus on unsupervised training, i.e. only using input video data and without manual supervision based on semantic labels.

Deep neural networks have recently achieved state-of-the-art performances for data-driven video prediction. Seminal works include the application of sequence to sequence LSTM or Convolutional variants [236, 281], adopted in many studies [77, 156, 282]. Further works explore different architectural designs based on Recurrent Neural Networks (RNNs) [268, 266, 182, 269, 267] and 2D/3D ConvNets [166, 261, 204, 25]. Dedicated loss functions [55, 137] and Generative Adversarial Networks (GANs) have been investigated for sharper predictions [166, 261, 132]. However, the problem of conditioning GANs with prior information, such as physical models, remains an open question.

To constrain the challenging generation of high dimensional images at the pixel level, several methods rather use domain-specific knowledge such as predicting geometric transformations between

frames [77, 112, 283], estimating the optical flow [190, 161, 152**?** , 148] or exploiting the semantics of the scene [12]. This is very effective for short-term prediction, but degrades quickly when the video content evolves, where more complex models and memory about dynamics are required.

Another line of work consists in disentangling independent factors of variations in order to apply the prediction model on lower-dimensional representations. A few approaches explicitly model interactions between objects inferred from an observed scene [73, 125, 285]. Relational reasoning, often implemented with graphs [9, 120, 220, 187, 254], can account for basic physical laws, e.g. drift, gravity, spring [272, 278, 176]. However, these methods are object-centric, only evaluate on controlled settings and are not suited for general real-world video forecasting. Other disentangling approaches factorize the video into independent components [258, 250, 60, 104, 85]. Several disentanglement criteria are used, such as content/motion [258, 144] or deterministic/stochastic [60]. In specific contexts, the prediction space can be structured using additional information, e.g. with human pose [259, 262] or key points [173], which imposes a severe overhead on the annotation budget. In this work, we share with these works the motivation to use disentangled representations, but we disentangle incomplete physical dynamics from residual information required for prediction.

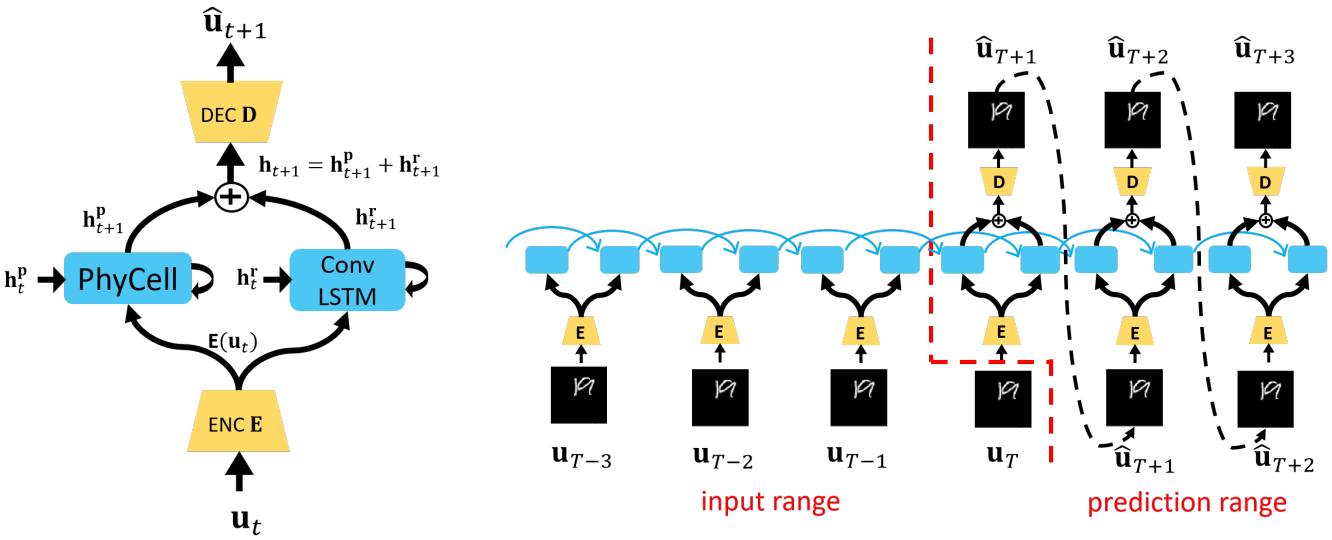
Deep Kalman filters To handle unobserved phenomena, state space models, in particular the Kalman filter [116], have been recently integrated with deep learning, by modelling dynamics in learned latent space [126, 271, 98, 81, 11]. The Kalman variational autoencoder [81] separates state estimation in videos from dynamics with a linear gaussian state space model. The Recurrent Kalman Network [11] uses a factorized high dimensional latent space in which the linear Kalman updates are simplified and don't require computationally-heavy covariance matrix inversions. These methods inspired by the data assimilation community [3, 18] have advantages in missing data or long-term forecasting contexts due to their mechanisms decoupling latent dynamics and input assimilation. However, they assume simple latent dynamics (linear) and don't include any physical prior.

6.3 PhyDNet model for video forecasting

We introduce PhyDNet, a model dedicated to video prediction, which leverages physical knowledge on dynamics, and disentangles it from other unknown factors of variations necessary for accurate forecasting. To achieve this goal, we introduce a disentangling architecture (section 6.3.1), and a new physically-constrained recurrent cell (section 6.3.2).

Problem statement: As discussed in introduction, physical laws do not apply at the pixel level for general video prediction tasks. However, we assume that there exists a conceptual latent space $\boldsymbol{\mathcal{H}}$ in which physical dynamics and residual factors are linearly disentangled. Formally, let us denote as $\mathbf{u} = \mathbf{u}(t, \mathbf{x})$ the frame of a video sequence at time t, for spatial coordinates $\mathbf{x} = (x, y)$. $\mathbf{h}(t, \mathbf{x}) \in \boldsymbol{\mathcal{H}}$ is the latent representation of the video up to time t, which decomposes as $\mathbf{h} = \mathbf{h^p} + \mathbf{h^r}$, where $\mathbf{h^p}$ (resp. $\mathbf{h^r}$) represents the physical (resp. residual) component of the disentanglement. The video evolution in the latent space $\boldsymbol{\mathcal{H}}$ is thus governed by the following partial differential equation (PDE):

$$\frac{\partial \mathbf{h}(t, \mathbf{x})}{\partial t} = \frac{\partial \mathbf{h^p}}{\partial t} + \frac{\partial \mathbf{h^r}}{\partial t} := \boldsymbol{\mathcal{M}}_p(\mathbf{h^p}, \mathbf{u}) + \boldsymbol{\mathcal{M}}_r(\mathbf{h^r}, \mathbf{u}). \tag{6.1}$$



(a) PhyDNet disentangling cell **(b) Global Seq2Seq architecture**

Figure 6.2: **Proposed PhyDNet deep model for video forecasting.** (a) The core of PhyDNet is a recurrent block projecting input images $\mathbf{u_t}$ into a latent space $\boldsymbol{\mathcal{H}}$, where two recurrent neural networks disentangle physical dynamics (PhyCell, section 6.3.2) from residual information (ConvLSTM). Learned physical $\mathbf{h}^{\mathbf{p}}_{t+1}$ and residual $\mathbf{h}^{\mathbf{r}}_{t+1}$ representations are summed before decoding to predict the future image $\hat{\mathbf{u}}_{t+1}$. (b) Unfolded in time, PhyDNet forms a sequence to sequence (seq2seq) architecture suited for multi-step video prediction. Dotted arrows mean that predictions are reinjected as next input only for the ConvLSTM branch, and not for PhyCell, as explained in section 6.3.3.

$\boldsymbol{\mathcal{M}}_p(\mathbf{h}^{\mathbf{p}}, \mathbf{u})$ and $\boldsymbol{\mathcal{M}}_r(\mathbf{h}^{\mathbf{r}}, \mathbf{u})$ represent physical and residual dynamics in the latent space $\boldsymbol{\mathcal{H}}$.

6.3.1 PhyDNet disentangling architecture

The main goal of PhyDNet is to learn the mapping from input sequences to a latent space which approximates the disentangling properties formalized in Eq 9.5.

To reach this objective, we introduce a recurrent bloc which is shown in Figure 6.2 (a). A video frame $\mathbf{u}_t$ at time t is mapped by a deep convolutional encoder $\mathbf{E}$ into a latent space representing the targeted space $\boldsymbol{\mathcal{H}}$. $\mathbf{E}(\mathbf{u}_t)$ is then used as input for two parallel recurrent neural networks, incorporating this spatial representation into a dynamical model.

The left branch in Figure 6.2 (a) models the latent representation $\mathbf{h}^{\mathbf{p}}$ fulfilling the physical part of the PDE in Eq (9.5), i.e. $\frac{\partial \mathbf{h}^{\mathbf{p}}(t,\mathbf{x})}{\partial t} = \boldsymbol{\mathcal{M}}_p(\mathbf{h}^{\mathbf{p}}, \mathbf{u})$. This PDE is modeled by our recurrent physical cell described in section 6.3.2, PhyCell, which leads to the computation of $\mathbf{h}^{\mathbf{p}}_{t+1}$ from $\mathbf{E}(\mathbf{u}_t)$ and $\mathbf{h}^{\mathbf{p}}_t$. From the machine learning perspective, PhyCell leverages physical constraints to limit the number of model parameters, regularizes training and improves generalization.

The right branch in Figure 6.2 (a) models the latent representation $\mathbf{h}^{\mathbf{r}}$ fulfilling the residual part of the PDE in Eq 9.5, i.e. $\frac{\partial \mathbf{h}^{\mathbf{r}}(t,\mathbf{x})}{\partial t} = \boldsymbol{\mathcal{M}}_r(\mathbf{h}^{\mathbf{r}}, \mathbf{u})$. Inspired by wavelet decomposition [163] and recent semi-supervised works [209], this part of the PDE corresponds to unknown phenomena, which do not

correspond to any prior model, and is therefore entirely learned from data. We use a generic recurrent neural network for this task, e.g. ConvLSTM [281] for videos, which computes $\mathbf{h}^{\mathbf{r}}_{t+1}$ from $\mathbf{E}(\mathbf{u}_t)$ and $\mathbf{h}^{\mathbf{r}}_t$.

$\mathbf{h}_{t+1} = \mathbf{h}^{\mathbf{p}}_{t+1} + \mathbf{h}^{\mathbf{r}}_{t+1}$ is the combined representation processed by a deep decoder $\mathbf{D}$ to forecast the image $\hat{\mathbf{u}}_{t+1}$.

Figure 6.2 (b) shows the "unfolded" PhyDNet. An input video $\mathbf{u}_{1:T} = (\mathbf{u}_1, ..., \mathbf{u}_T) \in \mathbb{R}^{T \times n \times m \times c}$ with spatial size $n \times m$ an d c channels is projected into $\boldsymbol{\mathcal{H}}$ by the encoder $\mathbf{E}$ and processed by the recurrent block unfolded in time. This forms a Sequence To Sequence architecture [240] suited for multi-step prediction, outputting H future frame predictions $\hat{\mathbf{u}}_{T+1:T+H}$. Encoder, decoder and recurrent block parameters are all trained end-to-end, meaning that PhyDNet learns itself without supervision the latent space $\boldsymbol{\mathcal{H}}$ in which physics and residual factors are disentangled.

6.3.2 PhyCell: a deep recurrent physical model

PhyCell is a new physical cell, whose dynamics is governed by the PDE response function $\boldsymbol{\mathcal{M}}_p(\mathbf{h}^{\mathbf{p}}, \mathbf{u})$[1]:

$$\boldsymbol{\mathcal{M}}_p(\mathbf{h}, \mathbf{u}) := \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u}), \tag{6.2}$$

where $\Phi(\mathbf{h})$ is a physical predictor modelling only the latent dynamics and $C(\mathbf{h}, \mathbf{u})$ is a correction term modelling the interactions between latent state and input data.

Physical predictor: $\Phi(\mathbf{h})$ in Eq (6.2) is modeled as follows:

$$\Phi(\mathbf{h}(t, \mathbf{x})) = \sum_{i,j:i+j \leq q} c_{i,j} \frac{\partial^{i+j} \mathbf{h}}{\partial x^i \partial y^j}(t, \mathbf{x}). \tag{6.3}$$

$\Phi(\mathbf{h}(t, \mathbf{x}))$ in Eq 9.2 combines the spatial derivatives with coefficients $c_{i,j}$ up to a certain differential order q. This generic class of linear PDEs subsumes a wide range of classical physical models, e.g. the heat equation, the wave equations, the advection-diffusion equations.

Correction: $\mathcal{C}(\mathbf{h}, \mathbf{u})$ in Eq 6.2 takes the following form:

$$\mathcal{C}(\mathbf{h}, \mathbf{u}) := \mathbf{K}(t, \mathbf{x}) \odot [\mathbf{E}(\mathbf{u}(t, \mathbf{x})) - (\mathbf{h}(t, \mathbf{x}) + \Phi(\mathbf{h}(t, \mathbf{x}))] \,. \tag{6.4}$$

Eq 6.4 computes the difference between the latent state after physical motion $\mathbf{h}(t, \mathbf{x}) + \Phi(\mathbf{h}(t, \mathbf{x}))$ and the embedded new observed input $\mathbf{E}(\mathbf{u}(t, \mathbf{x}))$. $\mathbf{K}(t, \mathbf{x})$ is a gating factor, where $\odot$ is the Hadamard product.

6.3.2.1 Discrete PhyCell

We discretize the continuous time PDE in Eq 6.2 with the standard forward Euler numerical scheme [158], leading to the discrete time PhyCell (derivation in Appendix D.1.1):

$$\mathbf{h}_{t+1} = (1 - \mathbf{K}_t) \odot (\mathbf{h}_t + \Phi(\mathbf{h}_t)) + \mathbf{K}_t \odot \mathbf{E}(\mathbf{u}_t). \tag{6.5}$$

[1] In the sequel, we drop the index $\mathbf{p}$ in $\mathbf{h}^{\mathbf{p}}$ for the sake of simplicity

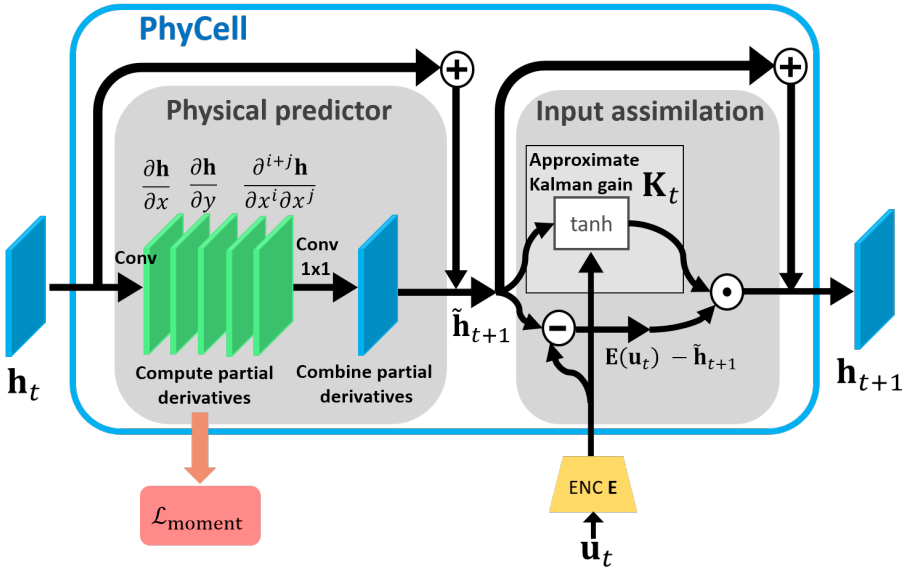


Figure 6.3: PhyCell recurrent cell implements a two-steps scheme: physical prediction with convolutions for approximating and combining spatial derivatives (Eq D.5 and Eq 9.2), and input assimilation as a correction of latent physical dynamics driven by observed data (Eq D.6). During training, the filter moment loss in red (Eq D.1.2) enforces the convolutional filters to approximate the desired differential operators.

Depicted in Figure 6.3, PhyCell is an atomic recurrent cell for building physically-constrained RNNs. In our experiments, we use one layer of PhyCell but one can also easily stack several PhyCell layers to build more complex models, as done for stacked RNNs [268, 266, 269]. To gain insight into PhyCell in Eq (6.5), we write the equivalent two-steps form:

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = \mathbf{h}_t + \Phi(\mathbf{h}_t) & \textbf{Prediction} \quad (6.6) \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \mathbf{K}_t \odot \left(\mathbf{E}(\mathbf{u}_t) - \tilde{\mathbf{h}}_{t+1}\right). & \textbf{Correction} \quad (6.7) \end{cases}$$

The prediction step in Eq D.5 is a physically-constrained motion in the latent space, computing the intermediate representation $\tilde{\mathbf{h}}_{t+1}$. Eq D.6 is a correction step incorporating input data. This prediction-correction formulation is reminiscent of the way to combine numerical models with observed data in the data assimilation community [3, 18], e.g. with the Kalman filter [116]. We show in section 6.3.3 that this decoupling between prediction and correction can be leveraged to robustly train our model in long-term forecasting and missing data contexts. $\mathbf{K}_t$ can be interpreted as the Kalman gain controlling the trade-off between both steps.

6.3.2.2 PhyCell implementation

We now specify how the physical predictor Φ in Eq D.5 and the correction Kalman gain $\mathbf{K}_t$ in Eq D.6 are implemented.

Physical predictor: We implement Φ using a convolutional neural network (left gray box in Figure 6.3), based on the connection between convolutions and differentiations [65, 155].

This offers the possibility to learn a class of filters approximating each partial derivative in Eq 9.2, which are constrained by a kernel moment loss, as detailed in section 6.3.3. As noted by [155], the flexibility added by this constrained learning strategy gives better results for solving PDEs than handcrafted derivative filters. Finally, we use 1×1 convolutions to linearly combine these derivatives with $c_{i,j}$ coefficients in Eq 9.2.

Kalman gain: We approximate $\mathbf{K}_t$ in Eq D.6 by a gate with learned convolution kernels $\mathbf{W}_h$, $\mathbf{W}_u$ and bias $\mathbf{b}_k$:

$$\mathbf{K}_t = \tanh\left(\mathbf{W}_h * \tilde{\mathbf{h}}_{t+1} + \mathbf{W}_u * \mathbf{E}(\mathbf{u}_t) + \mathbf{b}_k\right). \tag{6.8}$$

Note that if $\mathbf{K}_t = \mathbf{0}$, the input is not accounted for and the dynamics follows the physical predictor; if $\mathbf{K}_t = 1$, the latent dynamics is resetted and only driven by the input. This is similar to gating mechanisms in LSTMs or GRUs.

Discussion: With specific Φ predictor, $\mathbf{K}_t$ gain and encoder $\mathbf{E}$, PhyCell recovers recent models from the literature:

model	Φ	$\mathbf{K}_t$	$\mathbf{E}$
PDE-Net [154]	Eq D.5	$\mathbf{0}$	$\mathbf{Id}$
Advection-diffusion flow [58]	advection-diffusion predictor	$\mathbf{0}$	$\mathbf{Id}$
Recurrent Kalman Filter [11]	locally linear, no physical constraint	approximate Kalman gain	deep encoder
PhyDNet (ours)	Eq D.5	Eq 6.8	deep encoder

PDE-Net [155] directly works on raw pixel data (identity encoder $\mathbf{E}$) and assumes Markovian dynamics (no correction, $\mathbf{K}_t = \mathbf{0}$): the model solves the autonomous PDE $\frac{\partial \mathbf{u}}{\partial t} = \Phi(\mathbf{u})$ given in Eq D.5 but in pixel space. This prevents from modelling time-varying PDEs such as those tackled in this work, e.g. varying advection terms. The flow model in [58] uses the closed-form solution of the advection-diffusion equation as predictor ; it is however limited only to this PDE, whereas PhyDNet models a much broader class of PDEs. The Recurent Kalman Filter (RKF) [11] also proposes a prediction-correction scheme in a deep latent space, but their approach does not include any prior physical information, and the prediction step is locally linear, whereas we use deep models. An approximated form of the covariance matrix is used for estimating $\mathbf{K}_t$ in [11], which we find experimentally inferior to our gating mechanism in Eq 6.8.

6.3.3 Training

Given a training set of N videos $\boldsymbol{\mathcal{D}} = \left\{\mathbf{u}^{(i)}\right\}_{i=\{1:N\}}$ and PhyDNet parameters $\mathbf{w} = (\mathbf{w_p}, \mathbf{w_r}, \mathbf{w_s})$, where $\mathbf{w_p}$ (resp. $\mathbf{w_r}$) are parameters of the PhyCell (resp. residual) branch, and $\mathbf{w_s}$ are encoder and

decoder shared parameters, we minimize the following objective function:

$$\mathcal{L}(\boldsymbol{\mathcal{D}}, \mathbf{w}) = \mathcal{L}_{\text{image}}(\boldsymbol{\mathcal{D}}, \mathbf{w}) + \lambda \mathcal{L}_{\text{moment}}(\mathbf{w_p}). \tag{6.9}$$

We use the L^2 loss for the image reconstruction loss $\mathcal{L}_{\text{image}}$, as commonly done in the literature [268, 266, 182, 267, 269].

$\mathcal{L}_{\text{moment}}(\mathbf{w_p})$ imposes physical constraints on the k^2 learned filters $\left\{\mathbf{w}^k_{p,i,j}\right\}_{i,j\leq k}$, such that each $\mathbf{w}^k_{p,i,j}$ of size $k \times k$ approximates $\frac{\partial^{i+j}}{\partial x^i y^j}$. This is achieved by using a loss based on the moment matrix $\mathbf{M}(\mathbf{w}^k_{p,i,j})$ [154], representing the order of the filter differentiation [65]. $\mathbf{M}(\mathbf{w}^k_{p,i,j})$ is compared to a target moment matrix $\mathbf{\Delta}^k_{i,j}$ (see $\mathbf{M}$ and $\mathbf{\Delta}$ computations in Appendix D.1.2), leading to:

$$\mathcal{L}_{\text{moment}} = \sum_{i\leq k} \sum_{j\leq k} ||\mathbf{M}(\mathbf{w}^k_{p,i,j}) - \mathbf{\Delta}^k_{i,j}||_F. \tag{6.10}$$

Prediction mode: An appealing feature of PhyCell is that we can use and train the model in a "prediction-only" mode by setting $\mathbf{K}_t = \mathbf{0}$ in Eq D.6, i.e. by only relying on the physical predictor Φ in Eq D.5. It is worth mentioning that the "prediction-only" mode is not applicable to standard Seq2Seq RNNs: although the decomposition in Eq 6.2 still holds, i.e. $\boldsymbol{\mathcal{M}}_r(\mathbf{h}, \mathbf{u}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u})$, the resulting predictor is naive and useless for multi-step prediction $\tilde{\mathbf{h}}_{t+1} = 0$, see Appendix D.1.3.1).

Therefore, standard RNNs are not equipped to deal with unreliable input data $\mathbf{u}_t$. We show in section 6.4.4 that the gain of PhyDNet over those models increases in two important contexts with unreliable inputs: multi-step prediction and dealing with missing data.

6.4 Experiments

6.4.1 Experimental setup

We evaluate PhyDNet on four datasets from various origins.

Moving MNIST is a standard benchmark in video prediction [236] consisting in two random MNIST digits bouncing on the walls of a 64×64 grid. We predict 10 future frames given 10 input frames. Training sequences are generated on the fly and the test set of 10000 sequences is provided by [236].

Traffic BJ consists in traffic flow data collected by taxicabs in Beijing [296]. Each 32×32 image is a 2-channels heat map with leaving/entering traffic. Video prediction on such real-world complex data require modelling transport phenomena and traffic diffusion. Following the setting of [296, 269, 267], we predict 4 future frames given 4 input frames.

SST consists in daily Sea Surface Temperature (SST) data from the sophisticated simulation engine NEMO (Nucleus for European Modelling of the Ocean), as in [58]. SST evolution is governed by the physical laws of fluid dynamics. We predict 4 frames of size 64×64 given 4 input frames.

Human 3.6 contains 3.6 million images of human actions [107], with complex 3D articulated motions. Following the setting of [269], we use only the "walking" scenario with subjects S1, S5, S6, S7, S8 for training, and S9, S11 for testing. We predict 4 future images of size $128\times128\times3$ given 4 input images.

Network architectures and training: PhyDNet shares a common backbone architecture for all datasets where the physical branch contains 49 PhyCells filters (with kernel of size 7×7) and the residual branch is composed of a 3-layers ConvLSTM with 128 filters in each layer. We set up the trade-off parameter between $\mathcal{L}_{\text{image}}$ and $\mathcal{L}_{\text{moment}}$ to $\lambda = 1$. Detailed architectures and λ impact are given in Appendix D.2.1. Our code is available at `https://github.com/vincent-leguen/PhyDNet`.

Evaluation metrics: We follow evaluation metrics commonly used in state-of-the-art video prediction methods: the Mean Squared Error (MSE), Mean Absolute Error (MAE) and the Structural Similarity (SSIM) [270] that computes the perceived image quality with respect to a reference. Metrics are averaged for each frame of the output sequence. Lower MSE, MAE and higher SSIM indicate better performances.

6.4.2 State of the art comparison

We evaluate PhyDNet against strong recent baselines, including very competitive data-driven RNN architectures: ConvLSTM [281], PredRNN [268], Causal LSTM [266], Memory in Memory (MIM) [269]. We also compare to methods dedicated to specific datasets: DDPAE [104], a disentangling method specialized and state-of-the-art on Moving MNIST ; and the physically-constrained advection-diffusion flow model [58] that is state-of-the-art for the SST dataset.

Overall results presented in Table 6.1 reveal that PhyDNet outperforms significantly all baselines on all four datasets. The performance gain is large with respect to state-of-the-art general RNN models, with a gain of 17 MSE points for Moving MNIST, 6 MSE points for Human 3.6, 3 MSE points for SST

Table 6.1: Quantitative forecasting results of PhyDNet compared to baselines using various datasets. Numbers are copied from original or citing papers. * corresponds to results obtained by running online code from the authors. The first five baseline are general deep models applicable to all datasets, whereas DDPAE [104] (resp. advection-diffusion flow [58]) are specific state-of-the-art models for Moving MNIST (resp. SST). Metrics are scaled to be in a similar range across datasets to ease comparison.

	Moving MNIST			**Traffic BJ**			**Sea Surface Temperature**			**Human 3.6**		
Method	MSE	MAE	SSIM	MSE $\times100$	MAE	SSIM	MSE $\times10$	MAE	SSIM	MSE / 10	MAE /100	SSIM
ConvLSTM [281]	103.3	182.9	0.707	48.5*	17.7*	0.978*	45.6*	63.1*	0.949*	50.4*	18.9*	0.776*
PredRNN [268]	56.8	126.1	0.867	46.4	17.1*	0.971*	41.9	62.1	0.955	48.4	18.9	0.781
Causal LSTM [266]	46.5	106.8	0.898	44.8	16.9*	0.977*	39.1*	62.3*	0.929*	45.8	17.2	0.851
MIM [269]	44.2	101.1	0.910	42.9	16.6*	0.971*	42.1*	60.8*	0.955*	42.9	17.8	0.790
E3D-LSTM [267]	41.3	86.4	0.920	43.2*	16.9*	0.979*	34.7*	59.1*	0.969*	46.4	16.6	0.869
Advection-diffusion [58]	-	-	-	-	-	-	34.1*	54.1*	0.966*	-	-	-
DDPAE [104]	38.9	90.7*	0.922*	-	-	-	-	-	-	-	-	-
PhyDNet	**24.4**	**70.3**	**0.947**	**41.9**	**16.2**	**0.982**	**31.9**	**53.3**	**0.972**	**36.9**	**16.2**	**0.901**

Table 6.2: An ablation study shows the consistent performance gain on all datasets of our physically-constrained PhyCell vs the general purpose ConvLSTM, and the additional gain brought up by the disentangling architecture PhyDNet. * corresponds to results obtained by running online code from the authors.

	Moving MNIST			**Traffic BJ**			**Sea Surface Temperature**			**Human 3.6**		
Method	MSE	MAE	SSIM	MSE × 100	MAE	SSIM	MSE × 10	MAE	SSIM	MSE / 10	MAE / 100	SSIM
ConvLSTM	103.3	182.9	0.707	48.5*	17.7*	0.978*	45.6*	63.1*	0.949*	50.4*	18.9*	0.776*
PhyCell	50.8	129.3	0.870	48.9	17.9	0.978	38.2	60.2	0.969	42.5	18.3	0.891
PhyDNet	**24.4**	**70.3**	**0.947**	**41.9**	**16.2**	**0.982**	**31.9**	**53.3**	**0.972**	**36.9**	**16.2**	**0.901**

and 1 MSE point for Traffic BJ. In addition, PhyDNet also outperforms specialized models: it gains 14 MSE points compared to the disentangling DDPAE model [104] specialized for Moving MNIST, and 2 MSE points compared to the advection-diffusion model [58] dedicated to SST data. PhyDNet also presents large and consistent gains in SSIM, indicating that image quality is greatly improved by the physical regularization. Note that for Human 3.6, a few approaches use specific strategies dedicated to human motion with additional supervision, e.g. human pose in [259]. We perform similarly to [259] using only unsupervised training, as shown in Appendix D.2.2. This is, to the best of our knowledge, the first time that physically-constrained deep models reach state-of-the-art performances on generalist video prediction datasets.

In Figure 6.4, we provide qualitative prediction results for all datasets, showing that PhyDNet properly forecasts future images for the considered horizons: digits are sharply and accurately predicted for Moving MNIST in (a), the absolute traffic flow error is low and approximately spatially independent in (b), the evolving physical SST phenomena are well anticipated in (c) and the future positions of the person is accurately predicted in (d). We add in Figure 6.4(a) a qualitative comparison to DDPAE [104], which fails to predict the future frames properly. Since the two digits overlap in the input sequence, DPPAE is unable to disentangle them. In contrast, PhyDNet successfully learns the physical dynamics of the two digits in a disentangled latent space, leading a correct prediction. In Appendix D.2.5, we detail this comparison to DPPAE, and provide additional visualizations for all datasets.

6.4.3 Ablation Study

We perform here an ablation study to analyse the respective contributions of physical modelling and disentanglement. Results are presented in Table 6.2 for all datasets. We see that a 1-layer PhyCell model (only the left branch of PhyDNet in Figure 6.2(b)) outperforms a 3-layers ConvLSTM (50 MSE points gained for Moving MNIST, 8 MSE points for Human 3.6, 7 MSE points for SST and equivalent results for Traffic BJ), while PhyCell has much fewer parameters (270,000 *vs.* 3 million parameters). This confirms that PhyCell is a very effective recurrent cell that successfully incorporates physical prior in deep models. When we further add our disentangling strategy with the two-branch architecture (PhyDNet), we have another performance gap on all datasets (25 MSE points for Moving MNIST, 7 points for Traffic and SST, and 5 points for Human 3.6), which proves that physical modelling is not sufficient by itself to perform general video prediction and that learning unknown factors is necessary.

To complement the discussion of Table 6.2, we give here in Table 6.3 the approximate number of models parameters of trained models:

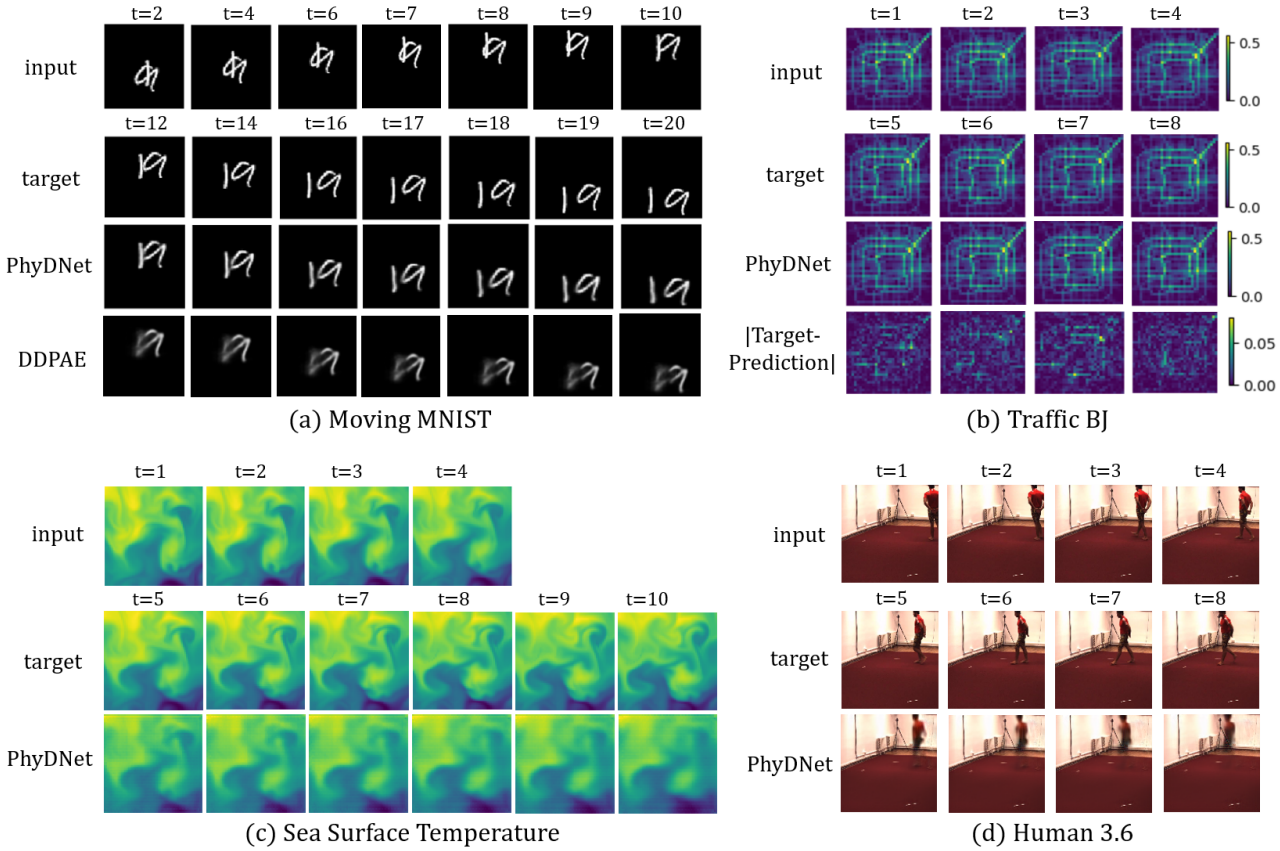


Figure 6.4: Qualitative results of the predicted frames by PhyDNet for all datasets. First line is the input sequence, second line the target and third line PhyDNet prediction. For Moving MNIST, we add a fourth line with the comparison to DDPAE [104] and for Traffic BJ the difference |Prediction-Target| for better visualization.

Table 6.3: Number of parameters of models trained on Moving MNIST.

method	number of parameters
ConvLSTM	3.10^6
PhyCell	370.10^3
PhyDNet	3.10^6

We see that a 1-layer PhyCell with 49 filters has far fewer parameters than a 3-layers ConvLSTM (with 128 filters in each layer) and obtains far better results (gain of 50 MSE points). Then PhyDNet with approximately the same number of parameters as ConvLSTM (3 million) again improves the performances by 25 MSE points, reaching a state-of-the-art MSE score of 24.4.

We qualitatively analyze in Figure D.6 partial predictions of PhyDNet for the physical branch $\hat{\mathbf{u}}^{\mathbf{p}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{p}}_{t+1})$ and residual branch $\hat{\mathbf{u}}^{\mathbf{r}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{r}}_{t+1})$. As noted in Figure 6.1 for Moving MNIST, $\mathbf{h^p}$ captures coarse localisations of objects, while $\mathbf{h^r}$ captures fine-grained details that are not useful for

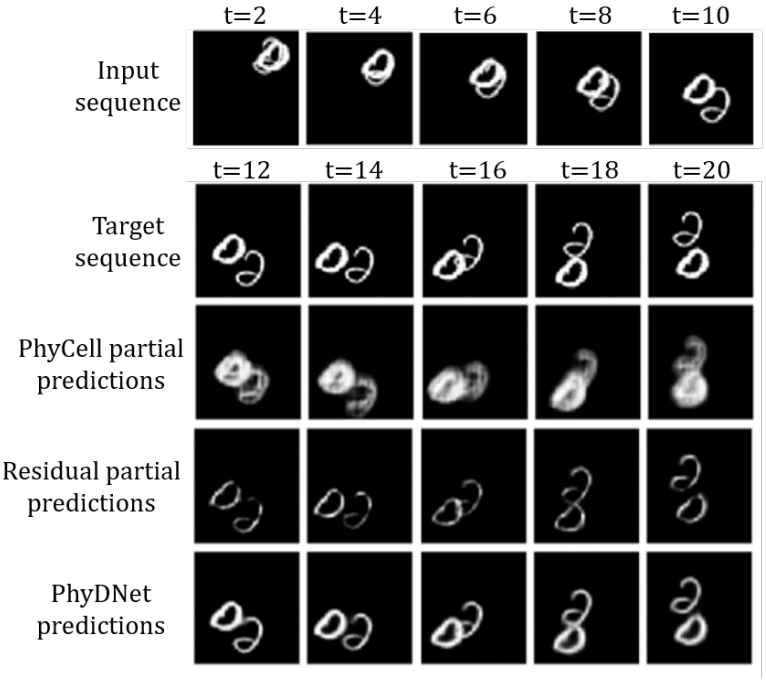


Figure 6.5: Qualitative ablation results on Moving MNIST: partial predictions show that PhyCell captures coarse localisation of digits, whereas the ConvLSTM branch models the fine shape details of digits. Every two frames are displayed.

the physical model. Additional visualizations for the other datasets are provided in Appendix D.2.5.

Influence of physical regularization We conduct in Table 6.4 a finer ablation on Moving MNIST to study the impact of the physical regularization $\mathcal{L}_{\text{moment}}$ on the performance of PhyCell and PhyDNet. When we disable $\mathcal{L}_{\text{moment}}$ for training PhyCell, performances improve by 7 points in MSE. This underlines that physical laws alone are too restrictive for learning dynamics in a general context, and that complementary factors should be accounted for. On the other side, when we disable $\mathcal{L}_{\text{moment}}$ for training our disentangled architecture PhyDNet, performances decrease by 5 MSE points (29 *vs* 24.4) compared to the physically-constrained version. This proves that physical constraints are relevant, but should be incorporated carefully in order to make both branches cooperate. This enables to leverage physical prior, while keeping remaining information necessary for pixel-level prediction. Same conclusions can be drawn for the other datasets, see Appendix D.2.4.

Table 6.4: Influence of physical regularization for Moving MNIST.

Method	MSE	MAE	SSIM
PhyCell	50.8	129.3	0.870
PhyCell without $\mathcal{L}_{\text{moment}}$	43.4	112.8	0.895
PhyDNet	**24.4**	**70.3**	**0.947**
PhyDNet without $\mathcal{L}_{\text{moment}}$	29.0	81.2	0.934

6.4.4 PhyCell analysis

Physical filter analysis With the same general backbone architecture, PhyDNet can express different PDE dynamics associated to the underlying phenomena by learning specific $c_{i,j}$ coefficients combining the partial derivatives in Eq (9.2). In Figure 6.6, we display the mean amplitude of the learned coefficients $c_{i,j}$ with respect to the order of differentiation. For Moving MNIST, the 0^{th} and 1^{st} orders are largely dominant, meaning a purely advective behaviour coherent with the piecewise-constant translation dynamics of the dataset. For Traffic BJ and SST, there is also a global decrease in amplitude with respect to order, we nonetheless notice a few higher order terms appearing to be useful for prediction. For Human 3.6, where the nature of the prior motion is less obvious, these coefficients are more spread across order derivatives.

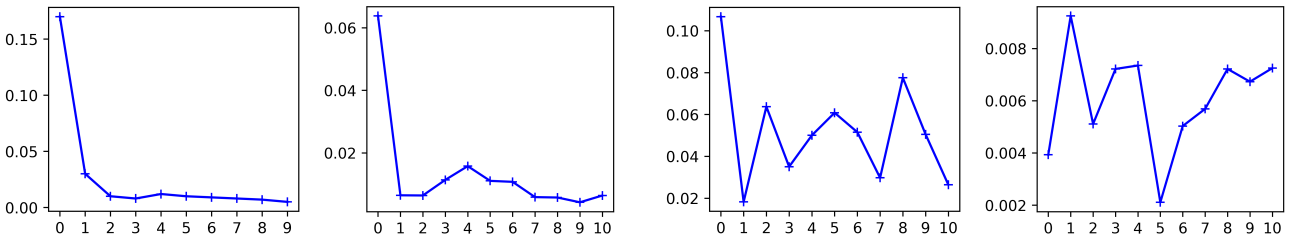
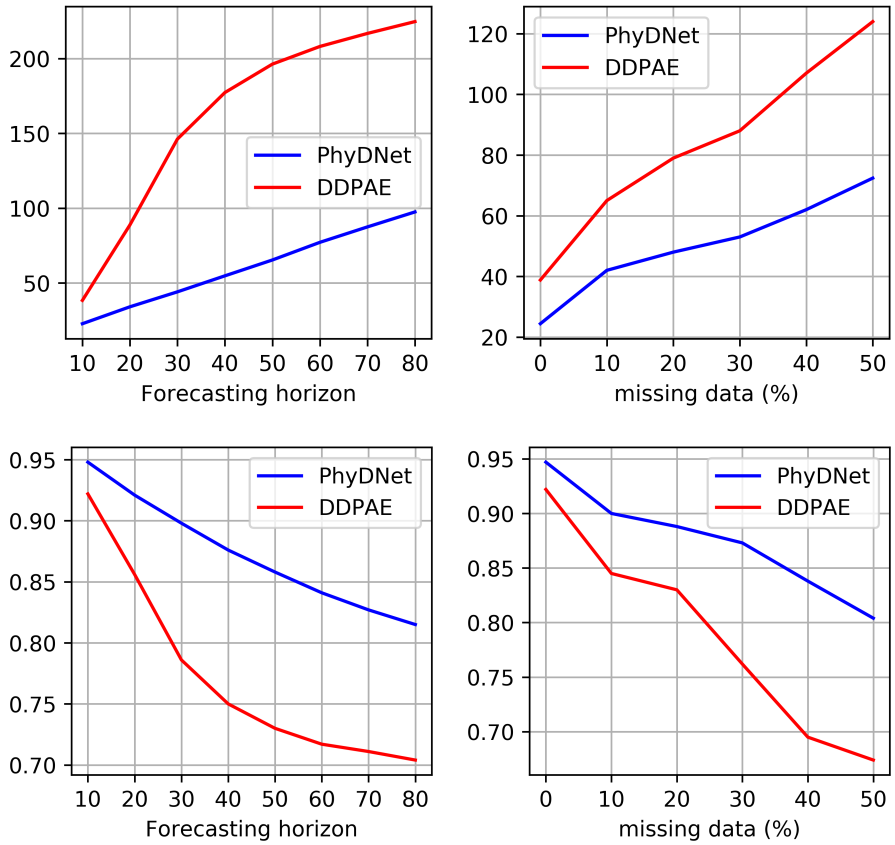


Figure 6.6: Mean amplitude of the combining coefficients $c_{i,j}$ with respect to the order of the differential operators approximated.

Dealing with unreliable inputs We explore here the robustness of PhyDNet when dealing with unreliable inputs, that can arise in two contexts: long-term forecasting and missing data. As explained in section 6.3.3, PhyDNet can be used in a prediction mode in this context, limiting the use of unreliable inputs, whereas general RNNs cannot. To validate the relevance of the prediction mode, we compare PhyDNet to DDPAE [104], based on a standard RNN (LSTM) as predictor module. Figure 6.7 presents the results evaluated in MSE and SSIM obtained by PhyDNet and DDPAE on Moving MNIST.

For long-term forecasting, we evaluate the performances of both methods far beyond the prediction range seen during training (up to 80 frames), as shown in Figure 6.7(a). We can see that the performance drop (MSE increase rate) is approximately linear for PhyNet, whereas it is much more pronounced for DDPAE. For example, PhyDNet for 80-steps prediction reaches similar performances in MSE than DDPAE for 20-steps prediction. This confirms that PhyDNet can limit error accumulation during forecasting by using a powerful dynamical model.

Finally, we evaluate the robustness of PhyDNet on DDPAE on missing data, by varying the ratio of missing data (from 10 to 50%) in input sequences during training and testing. A missing input image is replaced with a default value (0) image. In this case, PhyCell relies only on its latent dynamics by setting $\mathbf{K}_t = 0$, whereas DDPAE takes the null image as input. Figure 6.7(b) shows that the performance gap between PhyDNet and DDPAE increases with the percentage of missing data.



(a) Long-term forecasting (b) Missing data

Figure 6.7: MSE comparison between PhyDNet and DDPAE [104] when dealing with unreliable inputs, for long-term forecasting (a) and in presence of missing data (b).

6.5 Conclusion

We have proposed PhyDNet, a new model for disentangling prior dynamical knowledge from other factors of variation required for video prediction. PhyDNet enables to apply PDE-constrained prediction beyond fully observed physical phenomena in pixel space, and to outperform state-of-the-art performances on four generalist datasets. Our introduced recurrent physical cell for modelling PDE dynamics generalizes recent models and offers the appealing property to decouple prediction from correction.

Chapter 7

Augmenting incomplete physical models for complex dynamics forecasting

Content

Chapter abstract

Forecasting complex dynamical phenomena in settings where only partial knowledge of their dynamics is available is a prevalent problem across various scientific fields. e.g. in climate. While purely data-driven approaches are arguably insufficient in this context, standard physical modelling based approaches tend to be over-simplistic, inducing non-negligible errors. In this Chapter, we introduce the APHYNITY framework, a principled approach for augmenting incomplete physical dynamics described by differential equations with deep data-driven models. It consists in decomposing the dynamics into two components: a physical component accounting for the dynamics for which we have some prior knowledge, and a data-driven component accounting for errors of the physical model. The learning problem is carefully formulated such that the physical model explains as much of the data as possible, while the data-driven component only describes information that cannot be captured by the physical model, no more, no less. This not only provides the existence and uniqueness for this decomposition, but also ensures interpretability and benefits generalization. Experiments made on three important use cases, each representative of a different family of physical phenomena, show that

APHYNITY can efficiently leverage approximate physical models to accurately forecast the evolution of the system and correctly identify relevant physical parameters.
The work described in this Chapter is based on the following publication:

- *[288, 287]: Yuan Yin*, Vincent Le Guen*, Jeremie Dona*, Ibrahim Ayed*, Emmanuel de Bézenac*, Nicolas Thome and Patrick Gallinari. "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting", In International Conference on Learning Representations (ICLR 2021, oral presentation), JSTAT 2021.*

7.1 Introduction

Modelling and forecasting complex dynamical systems is a major challenge in domains such as environment and climate [210], health science [40], and in many industrial applications [247]. As explained in Chapter 1, Model-Based (MB) approaches typically rely on partial or ordinary differential equations (PDE/ODE) and stem from a deep understanding of the underlying physical phenomena. Machine learning (ML) and deep learning methods are more prior agnostic yet have become state-of-the-art for several spatio-temporal prediction. However, pure ML methods are still limited for modelling complex physical dynamics, and cannot properly generalize to new conditions unlike MB approaches.

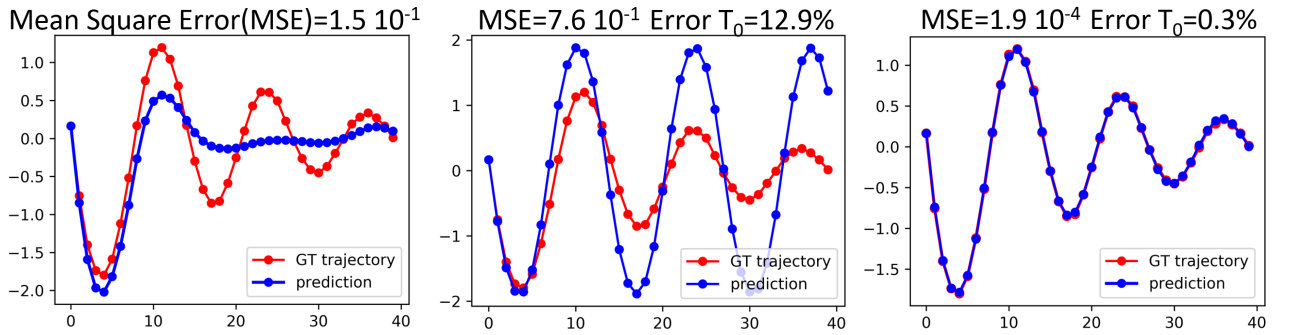
Combining the MB and ML paradigms is an emerging trend to develop the interplay between the two paradigms. For example, [23, 155] learn the explicit form of PDEs directly from data, [201, 231] use NNs as implicit methods for solving PDEs, [228] learn spatial differences with a graph network, [251] introduce continuous convolutions for fluid simulations, [58] learn the velocity field of an advection-diffusion system, [96, 37] enforce conservation laws in the network architecture or in the loss function.

The large majority of aforementioned ML/MB 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 [88], computational constraints prescribing a fine grain modelling of the system [4], unknown external factors, forces and sources which are present [135].

In this Chapter, we aim at leveraging prior dynamical ODE/PDE knowledge in situations where this physical model is *incomplete*, i.e. unable to represent the whole complexity of observed data. To handle this case, we introduce a principled learning framework to Augment incomplete PHYsical models for ideNtIfying and forecasTing complex dYnamics (APHYNITY). The rationale of APHYNITY, illustrated in Figure 7.1 on the pendulum problem, is to *augment* the physical model when—and only when—it falls short.

Designing a general method for combining ML and MB approaches is still a widely open problem, and a clear problem formulation for the latter is lacking [205]. Our contributions towards these goals are the following:

- We introduce a simple yet principled framework for combining both approaches. We decompose the data 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 7.3.1) for the decomposition given mild conditions, and show that this formulation ensures interpretability and benefits generalization.



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

Figure 7.1: Predicted dynamics for the damped pendulum vs. ground truth (GT) trajectories $\mathrm{d}^2\theta/\mathrm{d}t^2 + \omega_0^2 \sin\theta + \alpha \mathrm{d}\theta/\mathrm{d}t = 0$. We show that in (a) the data-driven approach [33] 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 shown in (c) augments the over-simplified physical model in (b) with a data-driven component. APHYNITY improves both forecasting (MSE) and parameter identification (Error T_0) compared to (b).

- We propose a trajectory-based training formulation (Section 7.3.2) along with an adaptive optimization scheme (Section 7.3.3) enabling end-to-end learning for both physical and 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 7.4). 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. Moreover, APHYNITY can also be successfully extended to the non-stationary dynamics context (Section 7.4.3).

7.2 Related work

Correction in data assimilation As discussed in Chapter 2, data assimilation techniques such as the Kalman filter [116, 11] assume that the prediction errors correspond to noise. These errors are modelled probabilistically as random variables, and an optimal correction step is derived after each prediction step. In this sequential two-step scheme, also arising commonly in robotics and optimal control [34, 146], there is no cooperation between prediction and correction. The originality of APHYNITY is to leverage model-based prior knowledge by augmenting it with neurally parameterized dynamics; the residual does not corresponds to noise but to an unknown or unmodelled part of the dynamical model. APHYNITY also ensures an optimal cooperation between the prior model and the augmentation.

Augmented physical models Combining physical models with machine learning (*gray-box or hybrid* modelling) was first explored from the 1990's: [196, 245, 206] use neural networks to predict the

unknown parameters of physical models. The challenge of proper MB/ML cooperation was already raised as a limitation of gray-box approaches but not addressed. Moreover these methods were evaluated on specific applications with a residual targeted to the form of the equation. In the last few years, there has been a renewed interest in deep hybrid models bridging data assimilation techniques and machine learning to identify complex PDE parameters using cautiously constrained forward model [155, 58].

Recently, some approaches have specifically targetted the ML/MB cooperation in the case of incomplete physical models. HybridNet [153] and PhICNet [216] both use data-driven networks to learn additive perturbations or source terms to a given PDE. The former considers the favorable context where the perturbations can be accessed, and the latter the special case of additive noise on the input. [264, 168] propose several empirical fusion strategies with deep neural networks but lack theoretical groundings. Crucially, all the aforementioned approaches do not address the issues of uniqueness of the decomposition or of proper cooperation for correct parameter identification. Besides, we found experimentally that this vanilla cooperation is inferior to the APHYNITY learning scheme in terms of forecasting and parameter identification performances (see experiments in Section 7.4.2).

7.3 The APHYNITY Model

In the following, we study dynamics driven by an equation of the form:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = F(X_t) \tag{7.1}$$

defined over a finite time interval $[0, T]$, where the state X is either vector-valued, i.e. we have $X_t \in \mathbb{R}^d$ for every t (pendulum equations in Section 7.4), or X_t is a d-dimensional vector field over a spatial domain $\Omega \subset \mathbb{R}^k$, with $k \in \{2, 3\}$, i.e. $X_t(x) \in \mathbb{R}^d$ for every $(t, x) \in [0, T] \times \Omega$ (reaction-diffusion and wave equations in Section 7.4). We suppose that we have access to a set of observed trajectories $\mathcal{D} = \{X_\cdot : [0, T] \to \mathcal{A} \mid \forall t \in [0, T], {}^{\mathrm{d}X_t}/_{\mathrm{d}t} = F(X_t)\}$, where $\mathcal{A}$ is the set of X values (either $\mathbb{R}^d$ or vector field). In our case, the unknown F has $\mathcal{A}$ as domain and we only assume that $F \in \mathcal{F}$, with $(\mathcal{F}, \|\cdot\|)$ a normed vector space.

The overall APHYNITY approach is illustrated in Figure 7.2.

7.3.1 Decomposing dynamics into physical and augmented terms

As introduced in 7.1, 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 $F_p \in \mathcal{F}_p \subset \mathcal{F}$. The APHYNITY framework leverages the knowledge of $\mathcal{F}_p$ 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 can write:

$$F = F_p + F_a,$$

where $F_p \in \mathcal{F}_p$ encodes the incomplete physical knowledge and $F_a \in \mathcal{F}$ is the data-driven augmentation term complementing F_p. The incomplete physical prior is supposed to belong to a known family, but the physical parameters (e.g. propagation speed for the wave equation) are unknown and need to be estimated from data. Both F_p and F_a parameters are estimated by fitting the trajectories from $\mathcal{D}$.

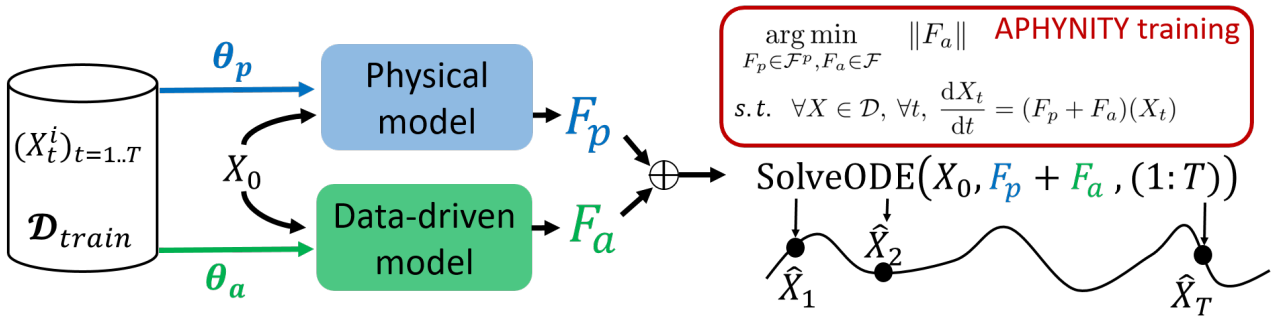


Figure 7.2: The APHYNITY model for learning complex dynamical systems augments an approximate physical model F_p by a deep data-driven model F_a. We propose a decomposition fulfilling uniqueness guarantees (Section 7.3.1). We introduce a trajectory-based formulation for learning the joint ODE $\frac{dX_t}{dt} = (F_p + F_a)(X_t)$, which leads to different and experimentally better identification results than the physical model F_p (Section 7.3.2). APHYNITY is learned end-to-end with an adaptive optimization algorithm (Section 7.3.3) ensuring a meaningful cooperation between physics and augmentation.

The decomposition $F = F_p + F_a$ is in general not unique. For example, all the dynamics could be captured by the F_a component. This decomposition is thus ill-defined, which hampers the interpretability and the extrapolation abilities of the model. In other words, one wants the estimated parameters of F_p to be as close as possible to the true parameter values of the physical model and F_a to play only a complementary role w.r.t F_p, so *as to model only the information that cannot be captured by the physical prior*. For example, when $F \in \mathcal{F}_p$, the data can be fully described by the physical model, and in this case it is sensible to desire F_a 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_a on the dynamics, as measured through its norm, should be as small as possible.

This general idea is embedded in the following optimization problem:

$$\min_{F_p \in \mathcal{F}_p, F_a \in \mathcal{F}} \|F_a\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{dX_t}{dt} = (F_p + F_a)(X_t). \tag{7.2}$$

The originality of APHYNITY is to leverage model-based prior knowledge by augmenting it with neurally parameterized 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 7.2 is indeed well-defined, in other words whether there exists indeed a decomposition with a minimal norm F_a. The answer actually depends on the geometry of $\mathcal{F}_p$, and is formulated in the following proposition proven in Appendix E.2:

Proposition 1 (Existence of a minimizing pair) *If $\mathcal{F}_p$ is a proximinal set*[1]*, there exists a decomposition minimizing Eq 7.2.*

Proximinality is a mild condition which, as shown through the proof of the proposition, cannot be

[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 E.1.

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_p 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 7.2 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 E.2:

Proposition 2 (Uniqueness of the minimizing pair) *If $\mathcal{F}_p$ is a Chebyshev set[1], Eq 7.2 admits a unique minimizer. The F_p in this minimizer pair is the metric projection of the unknown F onto $\mathcal{F}_p$.*

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 $\mathcal{F} = L^2$, $\mathcal{F}_p$ can be any closed convex set, including all finite dimensional subspaces. In particular, all examples considered in the experiments are Chebyshev sets.

Propositions 1 and 2 provide, under mild conditions, the theoretical guarantees for the APHYNITY formulation to infer the correct MB/ML decomposition, thus enabling both recovering the proper physical parameters and accurate forecasting.

7.3.2 Solving APHYNITY with deep neural networks

In the following, both terms of the decomposition are parametrized and are denoted as $F_p^{\theta_p}$ and $F_a^{\theta_a}$. Solving APHYNITY then consists in estimating the parameters θ_p and θ_a. θ_p are the physical parameters and are typically low-dimensional, e.g. 2 or 3 in our experiments for the considered physical models. For F_a, 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 [201, 5].

When learning the parameters of $F_p^{\theta_p}$ and $F_a^{\theta_a}$, 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 7.2 requires estimating the state derivative $\mathrm{d}X_t/\mathrm{d}t$ appearing in the constraint term. One solution is to approximate this derivative using e.g. finite differences as in [23, 96, 53]. 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 E.4. We propose instead to solve Eq 7.2 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 $F_p^{\theta_p} + F_a^{\theta_a}$ 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_p, θ_a):

$$\min_{\theta_p,\theta_a} \quad \left\|F_a^{\theta_a}\right\| \quad \text{subject to} \quad \forall i, \forall k, \widetilde{X}^{(i)}_{k\Delta t} = X^{(i)}_{k\Delta t}, \tag{7.3}$$

where $\widetilde{X}^{(i)}_{k\Delta t}$ is the approximate solution of the integral $\int_{X_0^{(i)}}^{X_0^{(i)}+k\Delta t}(F_p^{\theta_p} + F_a^{\theta_a})(X_s)\,\mathrm{d}X_s$ obtained by a differentiable ODE solver.

In our setting, where we consider situations for which $F_p^{\theta_p}$ only partially describes the physical phenomenon, this coupled ML/MB formulation leads to different parameter estimates than using the MB formulation alone, as analyzed more thoroughly in Appendix E.3.

Interestingly, our experiments show that using this formulation also leads to a better identification of the physical parameters θ_p than when fitting the simplified physical model $F_p^{\theta_p}$ alone (Section 7.4). With only an incomplete knowledge on the physics, θ_p estimator will be biased by the additional dynamics which needs to be fitted in the data. Appendix E.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.

7.3.3 Adaptively constrained optimization

The formulation in Eq 7.3 involves constraints which are difficult to enforce exactly in practice. We considered a variant of the method of multipliers [15] which uses a sequence of Lagrangian relaxations $\mathcal{L}_{\lambda_j}(\theta_p, \theta_a)$:

$$\mathcal{L}_{\lambda_j}(\theta_p, \theta_a) = \|F_a^{\theta_a}\| + \lambda_j \cdot \mathcal{L}_{traj}(\theta_p, \theta_a), \tag{7.4}$$

where $\mathcal{L}_{traj}(\theta_p, \theta_a) = \sum_{i=1}^{N} \sum_{h=1}^{T/\Delta t} \|X_{h\Delta t}^{(i)} - \widetilde{X}_{h\Delta t}^{(i)}\|$.

Algorithm 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**
      $\theta_{j+1} = \theta_j - \tau_1 \nabla \left[\lambda_j \mathcal{L}_{traj}(\theta_j) + \|F_a\|\right]$
    end
  end
  $\lambda_{j+1} = \lambda_j + \tau_2 \mathcal{L}_{traj}(\theta_{j+1})$
end

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 in Eq 7.3. 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 $\theta = (\theta_p, \theta_a)$. This procedure is summarized in Algorithm 1. This adaptive iterative procedure allows us to obtain stable and robust results, in a reproducible fashion, as shown in the experiments.

7.4 Experimental validation

We validate our approach on 3 classes of challenging physical dynamics: the damped pendulum, reaction-diffusion, and wave propagation, representative of various application domains such as chemistry, biology or ecology (for reaction-diffusion) [26, 44, 260] and earth physic, acoustic, electromagnetism or even neuro-biology (for waves equations) [232, 180].

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

The last two dynamics are described by PDEs and 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.

7.4.1 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 $\mathcal{F}_p$ 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 E.5.

Damped pendulum: The evolution of a damped pendulum is governed by the ODE $\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2} + \omega_0^2 \sin\theta + \lambda\frac{\mathrm{d}\theta}{\mathrm{d}t} = 0$, 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 $X = (\theta, \frac{\mathrm{d}\theta}{\mathrm{d}t})$, the ODE can be written as in Eq 7.1 with $F : X \mapsto (\frac{\mathrm{d}\theta}{\mathrm{d}t}, -\omega_0^2 \sin\theta - \lambda\frac{\mathrm{d}\theta}{\mathrm{d}t})$.

We consider the following physical models of increasing complexity:

- *Hamiltonian models* [96, 246], an energy conservative approximation of the system, with $\mathcal{F}_p = \{F_p^{\mathcal{H}} : (u,v) \mapsto (\partial_y \mathcal{H}(u,v), -\partial_x \mathcal{H}(u,v)) \mid \mathcal{H} \in H^1(\mathbb{R}^2)\}$ where $H^1(\mathbb{R}^2)$ is the first order Sobolev space.
- *Param ODE (ω_0)*, the pendulum without friction, with $\mathcal{F}_p = \{F_p^{\omega_0^2} : (u,v) \mapsto (v, -\omega_0^2 \sin u) \mid \omega_0^2 \geq \omega_{\min}^2\}$.
- *Param ODE (ω_0, λ)*, the full pendulum equation (but with unknown parameters), with $\mathcal{F}_p = \{F_p^{\omega_0^2,\lambda} : (u,v) \mapsto (v, -\omega_0^2 \sin u - \lambda v) \mid \omega_0^2 \geq \omega_{\min}^2, \lambda \geq \lambda_{\min} > 0\}$.

Reaction-diffusion equations: We consider a 2D FitzHugh-Nagumo type model [121]. 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: $\mathcal{F}_p = \{F_p^{a,b} : (u, v) \mapsto (a\Delta u, b\Delta v) \mid a \geq a_{\min} > 0, b \geq b_{\min} > 0\}$;
- *Param PDE (a, b, k)* the full PDE with unknown parameters: $\mathcal{F}_p = \{F_p^{a,b,k} : (u, v) \mapsto (a\Delta u + R_u(u, v; k), b\Delta v + R_v(u, v) \mid a \geq a_{\min} > 0, b \geq b_{\min} > 0, k \geq k_{\min} > 0\}$.

Damped wave equations: We investigate the following 2-dimensional 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, as for reaction-diffusion, we consider a compact spatial domain Ω with Neumann homogeneous boundary conditions. Note that this damping differs from the pendulum case, as its effect is global.

The considered physical models are:

- *Param PDE (c)*, without damping term and $\mathcal{F}_p = \{F_p^c : (u, v) \mapsto (v, c^2\Delta u) \mid c \geq c_{\min} > 0\}$;
- *Param PDE (c, k)* with $\mathcal{F}_p = \{F_p^{c,k} : (u, v) \mapsto (v, c^2\Delta u - kv) \mid c \geq c_{\min} > 0, k \geq k_{\min} > 0\}$.

Baselines As purely data-driven baselines, we use Neural ODE [33] for the three problems and PredRNN++ ([266], for reaction-diffusion only) which are competitive models for datasets generated by differential equations and for spatio-temporal data. As ML/MB methods, in the ablations studies (see Appendix E.6), we compare for all problems, to the vanilla ML/MB cooperation scheme found in [264, 168]. 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 [96, 246] and to the the deep Galerkin method (DGM) [231]. See additional details in Appendix E.5.

7.4.2 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.

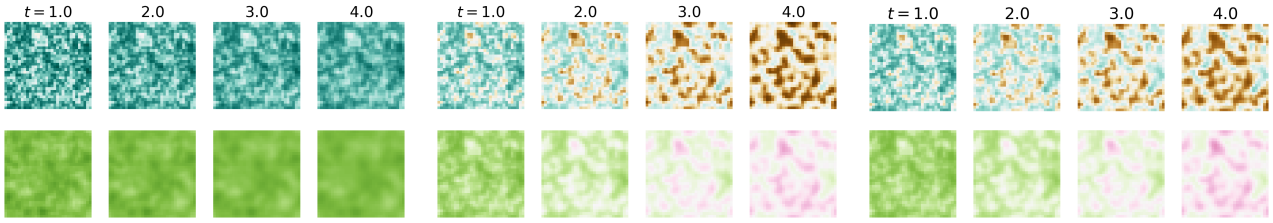
Forecasting accuracy: The data-driven models do not perform well compared to *True PDE/ODE* (all values are test errors expressed as log MSE): -4.6 for PredRNN++ vs. -9.17 for reaction-diffusion, -2.51 vs. -5.24 for wave equation, and -2.84 vs. -8.44 for the pendulum in Table 7.1. The Deep Galerkin method 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 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

Table 7.1: Forecasting and identification results on the (a) damped pendulum, (b) reaction-diffusion and (c) wave equation datasets. We set for (a) $T_0 = 6$, $\alpha = 0.2$, for (b) $a = 1{\times}10^{-3}, b = 5{\times}10^{-3}, k = 5{\times}10^{-3}$, and for (c) $c = 330$, $k = 50$ as true parameters. log MSEs are computed respectively over 40, 25 and 25 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.	$\|F_a\|^2$
(a) Damped pendulum	Data-driven	Neural ODE [33]	-2.84±0.70	n/a	n/a
	Incomplete physics	Hamiltonian [246]	-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) [231]	-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) [231]	-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
(b) Reaction-diffusion	Data-driven	Neural ODE [33]	-3.76±0.02	n/a	n/a
		PredRNN++ [266]	-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
(c) Wave equation	Data-driven	Neural ODE [33]	-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

with the same numerical scheme (which is different from the one used for data generation, hence the non-null errors even for the true equations), e.g. -5.92 vs. -5.24 for wave equation in Table 7.1. 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 E.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 7.1. 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



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

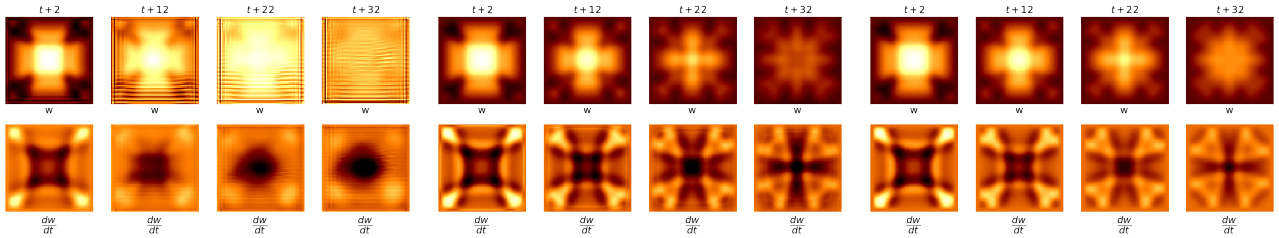
Figure 7.3: Comparison of predictions of two components u (top) and v (bottom) of the reaction-diffusion system. Note that $t = 4$ is largely beyond the dataset horizon ($t = 2.5$).

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 $F = F_p + F_a$ without taking care on the quality of the decomposition, as done in [264, 168]. Results shown in Table 7.1 of Appendix E.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 ($\log \text{MSE} =$-5.10 vs. -4.56) and identification parameter (Error= 2.33% vs. 6.39%) improve. Other ablation results are provided in Appendix E.6 showing the relevance of the the trajectory-based approach described in Section 7.3.2 (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 E.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_a, which is reported in Table 7.1 last column: as expected, $\|F_a\|^2$ 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 $\|F_a\| = 8.5$ for the APHYNITY model to be compared with 132 and 623 for the incomplete models). Thus, we see that the norm of F_a is a good indication of how imperfect the physical models $\mathcal{F}_p$ are. It highlights the flexibility of APHYNITY to successfully adapt to very different levels of prior knowledge. 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 E.5).

Qualitative visualizations: Results in Figure 7.3 for reaction-diffusion show that the incomplete diffusion parametric PDE in Figure 7.3(a) is unable to properly match ground truth simulations: the behavior of the two components in Figure 7.3(a) is reduced to simple independent diffusions due to the lack of interaction terms between u and v. By using APHYNITY in Figure 7.3(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_a can learn the reaction terms and improve

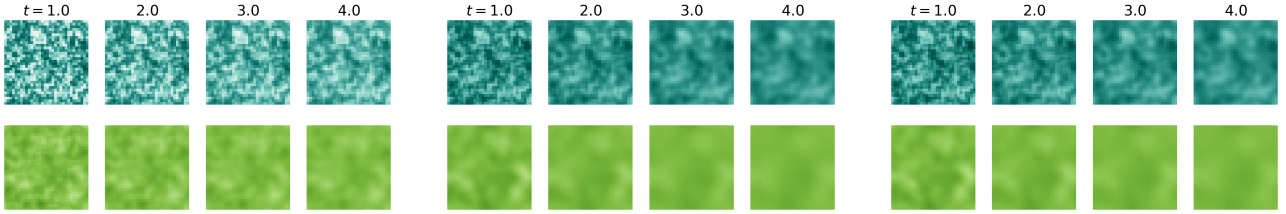


(a) Neural ODE (b) APHYNITY Param PDE (c) (c) Ground truth simulation

Figure 7.4: 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.

prediction quality. In Figure 7.4, we see for the wave equation that the data-driven Neural ODE model fails at approximating $\mathrm{d}w/\mathrm{d}t$ as the forecast horizon increases: it misses crucial details for the second component $\mathrm{d}w/\mathrm{d}t$ which makes the forecast diverge from the ground truth. APHYNITY incorporates a Laplacian term as well as the data-driven F_a thus capturing the damping phenomenon and succeeding in maintaining physically sound results for long term forecasts, unlike Neural ODE.

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 (Figure 7.5(a)) and the true one (Figure 7.5(c)) while APHYNITY gives a fairly good estimation of those diffusion parameters as shown in Figure 7.5(b).



(a) $a = 0.33\times10^{-3}, b = 0.94\times10^{-3}$, 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 7.5: 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

7.4.3 Extension to non-stationary dynamics

We evaluate here the applicability of APHYNITY in a more challenging setting where physical parameters of the equations vary in each sequence. For the damped pendulum equations, instead of fixed parameters ($T_0 = 6, \alpha = 0.2$) and varying initial conditions (Section 7.4.2), 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. 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 [183].

Results shown in Table 7.2 are consistent with those presented in Section 7.4.2. Pure data-driven models Neural ODE [33] and N-Beats [183] 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.

We provide similar experiments for the reaction-diffusion and wave equations in Appendix E.7.

Table 7.2: 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	$\|F_a\|^2$
data-driven	Neural ODE [33]	-4.35±0.9	n/a	n/a	n/a
	N-Beats [183]	-4.57±0.5	n/a	n/a	n/a
Incomplete physics	Hamiltonian [96]	-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

7.5 Conclusion

In this Chapter, we have introduced the APHYNITY framework that can efficiently augment approximate physical models with deep data-driven networks, performing similarly to models with full-known dynamics. We have exhibited 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.

Part III

Application to solar irradiance forecasting

Abstract

In this final part, we tackle the industrial solar energy forecasting problem with fisheye images that we briefly discussed in Chapter 1. We first present in details the use-case, and review the existing traditional methods and the early deep learning approaches (Chapter 8). We also propose a first data-driven deep learning model for solar irradiance estimation and prediction and discuss its limitations. In Chapter 9, we investigate the model-based machine learning cooperation studied in this thesis for improving the model. We propose a new physically-constrained architecture adapted from our PhyDNet video prediction model (Chapter 6). We also evaluate the use of our DILATE loss (Chapter 4) for enforcing predictions with accurate shape and temporal localization, and of our APHYNITY framework (Chapter 7) for optimal ML/MB decomposition.

Chapter 8

Overview of solar irradiance forecasting

Content

Chapter abstract

In this Chapter, we describe in details the industrial solar irradiance forecasting problem with fisheye images at EDF. We first review the traditional image processing and machine learning techniques, and the early deep learning approaches that have recently shown promising results. We propose a first deep learning model for estimating and predicting the future solar irradiance, which will be a strong deep baseline for the following Chapter.
The work described in this Chapter is based on the following publication:

- *[136] Vincent Le Guen and Nicolas Thome. "Prévision de l'irradiance solaire par réseaux de neurones profonds à l'aide de caméras au sol". In: GRETSI 2019.*

8.1 Introduction

To tackle climate change and limit global warming, major world economies agreed in 2015 at the Paris climate conference (COP21) on a restrictive plan to reduce greenhouse gas emissions. In the energy sector, this reinforced massive investments towards renewable energy generation such as solar or wind. However, a limitation of solar and wind energies is their intermittent and non-controllable nature, in contrast to conventional fossil fuel or nuclear energy. This causes major challenges for their integration at scale in the existing electricity grid, since electricity production and consumption must be balanced at every time. Therefore, accurately forecasting the intermittent energy production at various time horizons (from seconds to a few days) becomes a crucial aspect of the energy transition. Many applications could benefit from improved solar energy forecasts, such as the development of smart grids, hybrid solar/conventional power systems, or energy trading.

8.1.1 The solar irradiance components

In this thesis, we are interested in forecasting the solar irradiance, which corresponds to the incoming power of electromagnetic radiation received from the sun (expressed in W/m^2). The Global Horizontal Irradiance (GHI) can be decomposed into the Direct Normal Irradiance (DNI) directly coming from the sun perpendicularly to the photovoltaic (PV) panels, and the Diffuse Horizontal Irradiance (DHI) coming from the diffusion by the clouds and aerosols of the atmosphere or reflection from the ground (see Figure 8.1):

$$\text{GHI} = \text{DHI} + \sin h \times \text{DNI} \tag{8.1}$$

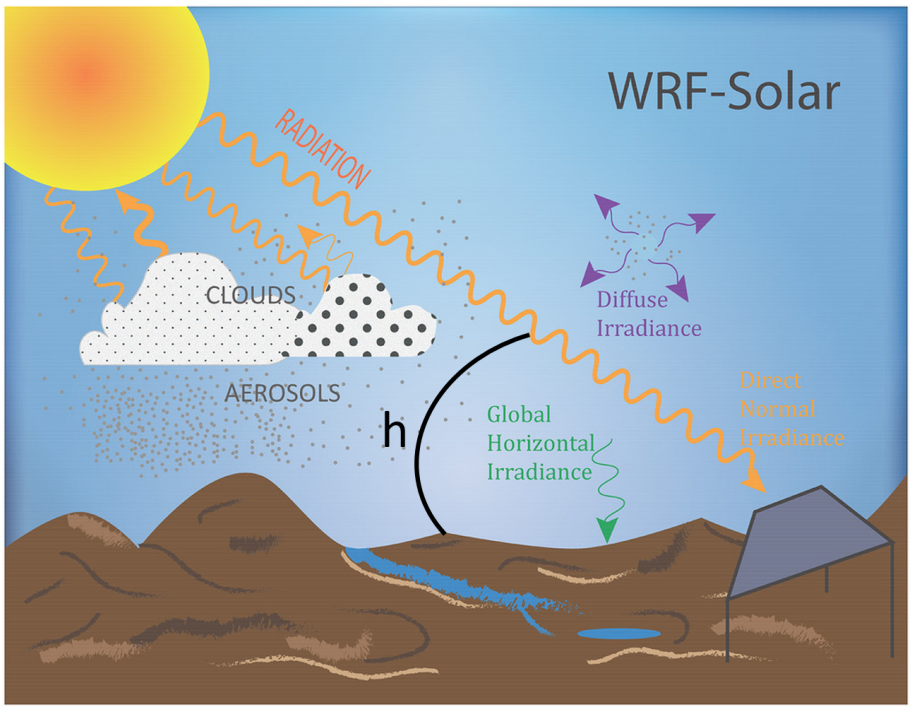


Figure 8.1: The different components of solar irradiance. Figure taken from [113].

where h is the solar elevation angle.

The GHI is the main quantity of interest in this thesis, since it is directly linked to the electric power production expressed in Watts, by knowing the technology, orientation of the photovoltaic panels and the ambient temperature. In practice, before applying any statistical method, the solar irradiance is often normalized by a clear-sky model corresponding to the theoretical irradiance received in cloudless conditions. This normalization compensates for the inherent seasonality of the solar irradiance. In this thesis, we use the ESRA (European Solar Radiation Atlas) clear sky model [207] and we denote KGHI the GHI normalized by its clear-sky values.

8.1.2 The different data sources for solar irradiance forecasting

For solar energy, the main source of variability comes from the occlusion of the sun by clouds. We presented in Figure 1.4 the main classes of methods for forecasting solar irradiance. Although statistical time series forecasting can be directly applied on the 1D solar irradiance series, this strategy is blind to the motions of clouds and thus cannot properly anticipate the variations. To understand the spatio-temporal dynamics of clouds, current methods rely on weather forecasts or sky image analysis. Numerical weather forecasts solve the equations of physics to forecast the dynamics of the atmosphere ; they have a spatial resolution of around 1km and a temporal resolution of 1 to 2h for the AROME model of Meteo France. For shorter forecasting horizons, satellite images can be exploited to provide irradiance forecasts up to a few hours with a 15 min granularity and a 1km spatial scale.

For very short-term horizons ($<$ 20min) at the scale of a PV plant, fisheye cameras pointed towards the sky (see Figure 8.2) have become popular in recent years [87, 43, 42, 164, 223, 129]. They offer an hemispheric view of the sky that enables to assess the evolution of the cloud cover.

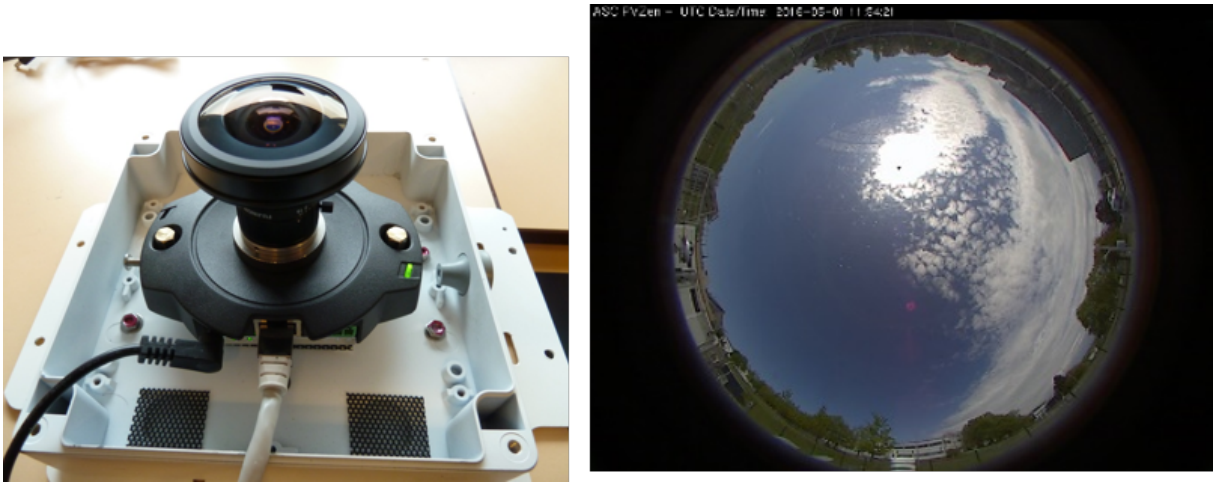


Figure 8.2: Fisheye camera and fisheye image for short-term solar irradiance forecasting.

8.1.3 Meteorological campaign at EDF R&D with fisheye images

EDF Research and Development (R&D) has led a meteorological campaign since 2010 at La Réunion Island with fisheye cameras (Axis PTZ 212) and pyranometers (SPN1) for measuring ground truth

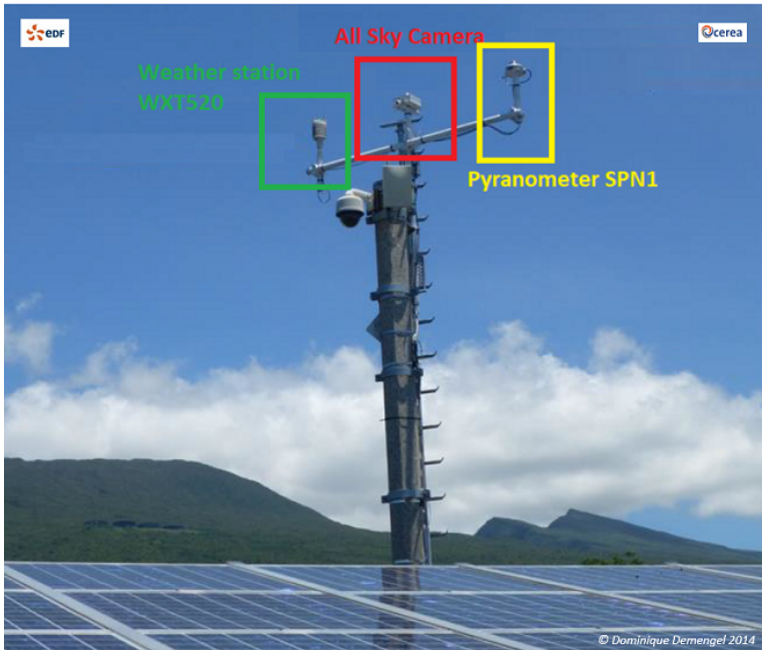


Figure 8.3: EDF scientific test site at La Reunion Island composed of a fisheye camera, a pyranometer and a weather station mounted above a PV power plant.

solar irradiance (see Figure 8.2 and Figure 8.3). A database of more than 7 million images every 10s and corresponding irradiance measurements was collected. The objective is to forecast solar irradiance with fisheye cameras only, which are much cheaper than pyranometers and provide an additional spatial information compared to irradiance time series.

8.2 Related work

In the Section, we review the main existing methods for short-term solar irradiance forecasting.

Persistence and statistical models For very-short term forecasting, a first natural baseline is the persistence, which assumes that the current irradiance level (normalized by the clear-sky) will persist. Persistence is often a competitive baseline, with optimal performances in clear-sky conditions. However, persistence does not anticipate variations by definition. Other statistical models [62, 276, 170] use local information (e.g. past irradiance values, PV production, temperature, weather forecasts) to capture statistical patterns and predict future values with regression or clustering algorithms. However, these methods do not observe the cloud motion and thus fail to anticipate variations due to sun occlusions.

Ground-based images For assessing the cloud coverage and anticipating short-term variations due to sun occlusions, researchers have investigated sky imagery with ground-based cameras from the 2010's. Earlier works have used specific scientific instruments, such as the Total Sky Imager in [41, 164] (spherical mirror with a camera pointing downwards) or suntrackers. Since, low-cost webcam cameras have encountered a great success, leading to a soaring interest from the community [87, 43, 42, 223].

Although many hardware and algorithmic variants exist (e.g. additional sensors, multiple cameras for stereo estimation), all these methods mainly follow a similar traditional image processing pipeline:

1. Camera calibration for determining the distortion parameters of the fisheye objective,
2. Fisheye image acquisition at fixed intervals (e.g. every 10s, 1min), sometimes with several expositions and High Dynamic Range (HDR) processing,
3. Image segmentation with thresholds based on color ratios or other photometric properties. Thresholds are either handcrafted or adaptative. The segmentation can be used for deriving a binary cloud map, or for deriving image features.
4. Cloud motion estimation with optical flow,
5. Cloud motion propagation into the future to generate a predicted irradiance map.

However sophisticated the processing pipeline may be, the challenges of the problem remain: the clouds follow a complex stochastic motion with abrupt variations that is hard to extrapolate. All these methods also rely on some manual engineering that only remains valid in a limited range of conditions.

Deep learning for solar irradiance forecasting Since a few years, deep learning has become an appealing alternative to replace the whole conventional pipeline with a model learned end-to-end from raw fisheye images [195, 295, 235, 239, 179, 186, 299]. However, as Paletta *et al.* [184] has highlighted, standard deep learning methods still struggle to properly understand the cloud motion and do not anticipate sharp variations.

8.3 Proposed models for solar irradiance estimation and forecasting

In this Section, we introduce two deep learning models: one for solar irradiance estimation, the second for forecasting. We define estimation as the prediction of the irradiance r_T associated with the image I_T. Forecasting corresponds to predicting the future irradiance r_{T+H} (or the complete future trajectory $r_{T+1}, \cdots, r_{T+H}$) given a sequence of past images $(I_1, \cdots, I_T)$.

8.3.1 Solar irradiance estimation

For solar irradiance estimation, we use a convolutional neural network that takes as input a fisheye image (without preprocessing) and outputs the estimated solar irradiance for that image. We first propose a handcrafted convolutional architecture (shown in Figure 8.4) working on RBG images resized at 80×80 pixels. This model has approximately 470,000 parameters.

We also propose a much larger model relying on the DenseNet architecture [105] that has reached state-of-art performances on the ImageNet image classification task. The model works with higher resolution images, resized at 224×224 pixels. For adapting the model to this regression task, we replace the final classification layers by fully-connected layers for outputting one irradiance value. The overall model has approximately 18 Million parameters.

Figure 8.4: Small convolutional network used for solar irradiance estimation.

8.3.2 Solar irradiance forecasting

To forecast solar irradiance, we propose a neural network architecture relying on the ConvLSTM model [281] which is a strong baseline for deep video prediction. Depicted in Figure 8.5, our architecture is composed of a ConvLSTM encoder that reads a sequence of T past fisheye images $(I_1, \cdots, I_{T-1}, I_T)$ and encodes them into a context vector. The network has two output branches: one for predicting the future solar irradiance $\hat{r}_{T+F}$ at a given horizon T and the other for the future fisheye image $\hat{I}_{T+H}$.

We empirically verified that this multi-task objective improves performances compared to forecasting irradiance only, due to the richer supervision signal and the cooperation between both tasks.

Our forecasting model is composed of 4 stacked ConvLSTM layers acting on input images resized to 80×80 pixels.

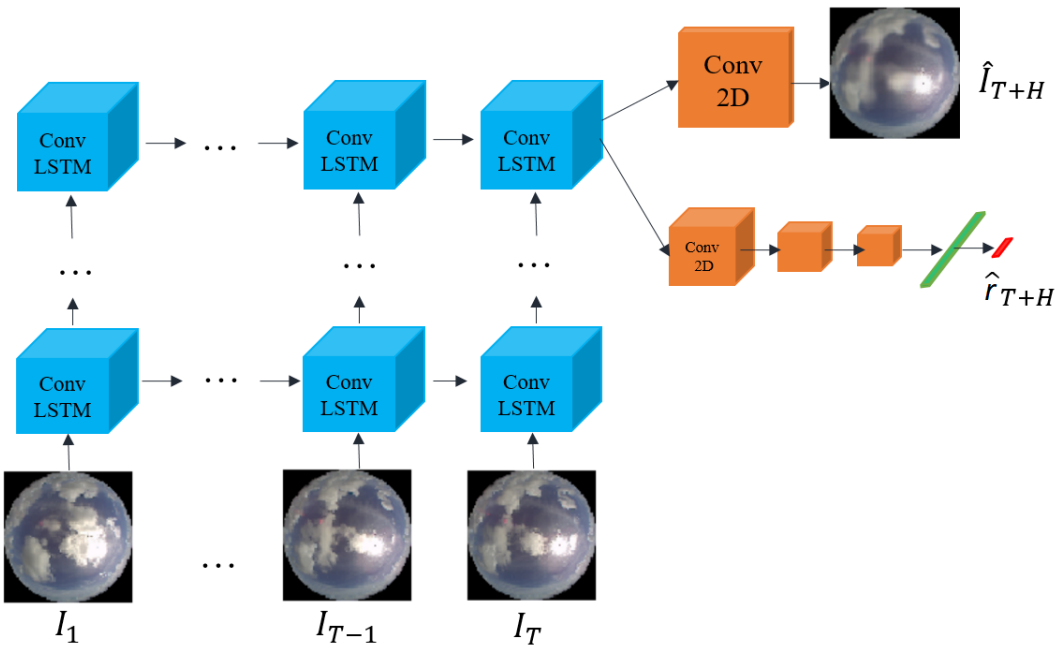


Figure 8.5: Proposed architecture for solar irradiance forecasting based on the ConvLSTM model [281].

8.4 Experimental results

8.4.1 Fisheye image dataset

We conduct experiments on the fisheye image dataset collected by EDF at La Reunion Island. For the estimation task, we use a training set composed of 4,190,064 images from the years 2012 to 2015, and a test set of 1,265,717 images in the year 2016. Images are processed from a solar elevation of 10°, and all irradiance measurements are normalized by the ESRA clear-sky model [207]. We use images resized at 80×80 pixels for the ConvNet model and 224×224 for the DenseNet model.

For the forecasting task, the training set is composed of 180,000 sequences of 10 images spaced by 1min (with the associated ground truth solar irradiance measurements) from the years 2014 to 2016, and the test set of 20,000 sequences during the year 2013 on the same site. We use images resized at 80×80 pixels. We keep 5 images for the input range and predict the 5 following images and solar irradiances.

8.4.2 Solar irradiance estimation results

We present in Table 8.1 the estimation results for the KGHI. We have trained two DenseNet models: one that only predicts the KGHI and the other that jointly predicts the KGHI and KDHI. We compare our proposed deep models with the baseline previously developed at EDF R&D [87]. This traditional method segments the fisheye images with thresholds on the R-B difference and the lumimance, defines 5 features based on the segmentation ratios and applies a Nadaraya-Watson kernel regression [177] for estimating the irradiance.

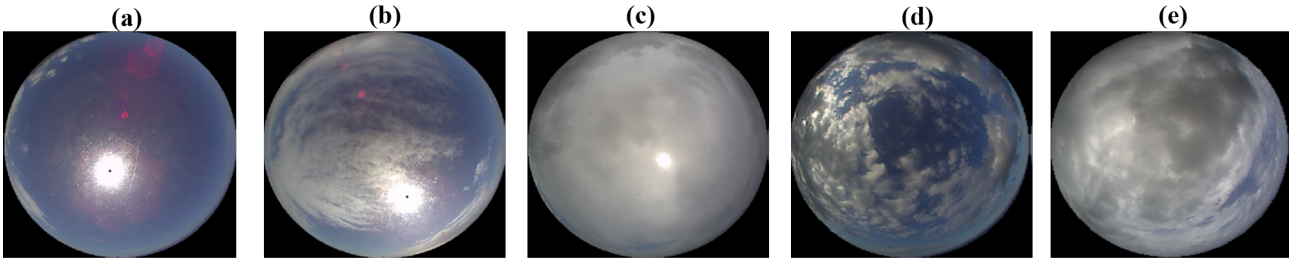
We evaluate the performances with the normalized Mean Absolute Error (nMAE) and normalized Root Mean Squared Error (nRSME). Normalization is performed by dividing by the mean KGHI value over the training set.

Table 8.1: KGHI estimation results on the test set.

model	nMAE	nRMSE
Baseline	14.9 %	21.6 %
ConvNet KGHI	6.59 %	10.3 %
DenseNet KGHI	2.91 %	5.27 %
DenseNet KGHI + KDHI	**2.90 %**	**4.83 %**

Results show that the ConvNet model (depicted in Figure 8.4) yields a large performance improvement (from 21.6 % to 10.3 % in nRMSE) over the baseline. Going deeper with the DenseNet model further gives a large gap in performances (5.27 %). It confirms the ability of deep learning to automatically learn a representation space for approximating a complex mapping from a large dataset of annotated images. Finally, we observe that the DenseNet model that jointly estimates the KGHI and KDHI gives the best performances (4.83 %), indicating that exploiting the correlations between both irradiance components helps in better estimating the KGHI. Intuitively, for two images with similar GHI but different cloud conditions, the differences of diffuse irradiance (DHI) should help to learn more specific cloud features that better generalize for estimating different test images.

We display in Figure 8.6 a few qualitative estimation examples. We can see for several sky conditions that the DenseNet estimations are very close to the measurements, both in GHI and DHI. Interestingly, the difference with the baseline is much higher when the diffuse irradiance (DHI) is high, e.g. for images (c) and (e). It can be explained by the difficulty of the image segmentation with handcrafted thresholds on clouds with different levels of gray; the deep learning approach better learns features for representing the shades of clouds, supervised by the GHI and DHI values.



W/m²	GHI	DHI
Clear-sky	965	136
Mesure	953	87
Baseline	969	107
Densenet	**961**	**88**

W/m²	GHI	DHI
Clear-sky	660	103
Mesure	769	215
Baseline	668	254
Densenet	**768**	**218**

W/m²	GHI	DHI
Clear-sky	983	147
Mesure	521	476
Baseline	460	411
Densenet	**507**	**487**

W/m²	GHI	DHI
Clear-sky	270	92
Mesure	111	103
Baseline	81	120
Densenet	**108**	**110**

W/m²	GHI	DHI
Clear-sky	1075	160
Mesure	278	273
Baseline	370	328
Densenet	**292**	**283**

Figure 8.6: Qualitative fisheye estimation results of the GHI and DHI.

8.4.3 Solar irradiance forecasting results

We then evaluate the forecasting performances of our method on the fisheye image dataset. We compare our ConvLSTM architecture with the optical flow baseline previously developed at EDF R&D [87] (sketched in Figure 2.3), and with the (smart) persistence which consists in copying the current value as the forecast for the future timestep (for quantities normalized by the clear sky).

Global results presented in Table 8.2 show that our proposed deep forecasting model outperforms both the optical flow baseline and the persistence. However, the performance gap with the traditional method is narrower than for estimation, revealing the difficulty of the forecasting task.

To further analyse the differences, we display in Figure 8.7 the model predictions on a particular day of the test set. We can see that the ConvLSTM predictions are much closer to the KGHI ground truth than the optical flow baseline and than the persistence ConvNet (which corresponds to applying the estimation ConvNet).

Interestingly, the optical flow baseline has a worse RMSE than the persistence. However, the optical flow method shows a better ability to anticipate sharp variations (e.g. around the timestep 200), and is therefore better suited for the industrial application. It confirms that the MSE and variants are not adapted to train and evaluate models in this non-stationary context with abrupt changes, which has motivated the contributions of this thesis. In the following Chapter, we will train and evaluate models with our proposed shape and temporal criteria to improve models in this context.

Table 8.2: Forecasting performances of the KGHI (normalized Global Horizontal Irradiance) at a 5min horizon.

Method	normalized RMSE
Optical flow baseline	32.9 %
Persistence	28.5 %
ConvLSTM (ours)	**26.6 %**

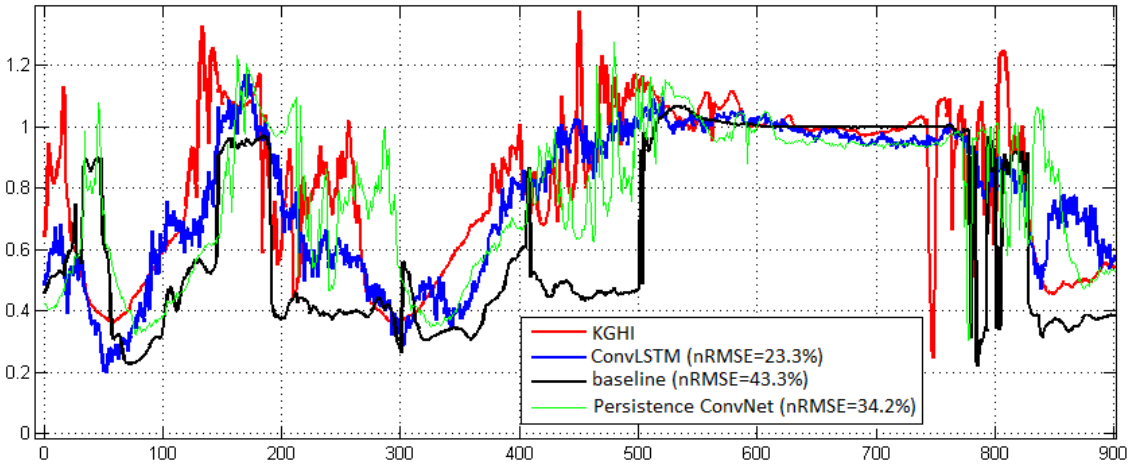


Figure 8.7: Qualitative KGHI forecasting results at 5min on a particular day.

8.5 Conclusion

In this Chapter, we have presented the solar irradiance forecasting problem with fisheye images at EDF, and reviewed the existing methods (traditional and deep). We have proposed first deep models for estimating and forecasting the solar irradiance, that have reached state-of-the-art results compared to traditional methods. However, for the forecasting task, there still exists room for improvement, in particular for modelling the sharp variations and the complex nonlinear cloud dynamics. These limitations will be addressed in the next Chapter.

Chapter 9

Deep learning for solar irradiance forecasting

Content

Chapter abstract

In this Chapter, we continue on the solar irradiance forecasting problem described in the previous Chapter. Based on the observation that common deep learning methods struggle to properly predict the complex cloud dynamics, we apply here the methodological innovations presented in parts I and II of this thesis. We first propose a new physically-constrained deep forecasting architecture based on our PhyDNet model for video prediction. We show that it significantly boosts performances compared to the model of the previous Chapter and to other state-of-the-art deep models. Then we apply the proposed DILATE training loss function (Chapter 4) for enforcing predictions with accurate shape and temporal localization. We also apply our APHYNITY framework (Chapter 7) that guarantees an optimal ML/MB decomposition. We discuss the benefits brought up by each of these mechanisms. The work described in this Chapter is based on the following publication:

- *[138]: Vincent Le Guen and Nicolas Thome. "A Deep Physical Model for Solar Irradiance Forecasting With Fisheye Images". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops 2020 (OmniCV 2020 workshop)*

9.1 Introduction

As discussed in the previous Chapter, forecasting solar irradiance with fisheye images remains a very difficult task for pure deep learning methods, because of the complex non-stationary motion of clouds. In this Chapter, we adapt the methodological contributions of this thesis, namely the DILATE loss function (Chapter 4), the PhyDNet video prediction model (Chapter 6) and the APHYNITY framework (Chapter 7), for solving this problem.

9.2 Proposed forecasting models

Given a dataset of fisheye images $\mathbf{u}_{1:T} = (\mathbf{u}_1, ..., \mathbf{u}_T)$ and associated solar irradiance measurements r_t, our goal is to forecast the future irradiance r_{T+H} for a given horizon H. First, we briefly review the PhyDNet model (Section 9.2.1) and propose an improvement to the architecture for better disentangling the physical and residual components (Section 9.2.2). Then, we propose two implementations of the PhyDNet model for solar irradiance forecasting (Section 9.2.3). The PhyDNet-monostep model is a direct adaptation of the architecture introduced in the previous Chapter, where the ConvLSTM is replaced by PhyDNet; we call this model PhyDNet-mono since we directly predict the future irradiance at the desired horizon r_{T+H}. We also propose the PhyDNet-multistep model, that forecasts the entire trajectory up to the desired horizon $(r_{T+1}, \cdots, r_{T+H})$. This multistep extension allows to exploit the whole intermediate trajectory for learning, for example by using the DILATE loss that compares multistep time series.

9.2.1 Review of the PhyDNet model

As described in Chapter 6, PhyDNet [139] is a deep architecture that leverages partial differential equations (PDEs) for video prediction. Since physics alone is not sufficient for accurate predictions at the pixel level, PhyDNet aims at learning a latent space $\boldsymbol{\mathcal{H}}$ that linearly disentangles physical dynamics from residual factors (such as texture, details,...). The latent state $\mathbf{h}$ is decomposed into physical and residual components $\mathbf{h} = \mathbf{h^p} + \mathbf{h^r}$, and follows the dynamics:

$$\frac{\partial \mathbf{h}(t, \mathbf{x})}{\partial t} = \frac{\partial \mathbf{h^p}}{\partial t} + \frac{\partial \mathbf{h^r}}{\partial t} := \boldsymbol{\mathcal{M}}_p(\mathbf{h^p}, \mathbf{E}(\mathbf{u})) + \boldsymbol{\mathcal{M}}_r(\mathbf{h^r}, \mathbf{E}(\mathbf{u})). \tag{9.1}$$

The physical model $\boldsymbol{\mathcal{M}}_p$ is composed of a PDE in latent space $\Phi_p(\mathbf{h^p})$ and a correction term $\mathcal{C}_p(\mathbf{h^p}, \mathbf{E}(\mathbf{u}))$ with input data (embedded by encoder $\mathbf{E}$): $\boldsymbol{\mathcal{M}}_p(\mathbf{h^p}, \mathbf{E}(\mathbf{u})) = \Phi_p(\mathbf{h^p}) + \mathcal{C}_p(\mathbf{h^p}, \mathbf{E}(\mathbf{u}))$. The physical predictor Φ_p encodes a general class of linear PDEs up to a differential order q:

$$\Phi_p(\mathbf{h^p}(t, \mathbf{x})) = \sum_{i,j:i+j \leq q} c_{i,j} \frac{\partial^{i+j} \mathbf{h^p}}{\partial x^i \partial y^j}(t, \mathbf{x}). \tag{9.2}$$

Partial derivatives are computed by constrained convolutions as in PDE-Net [155] and combined by learned coefficients c_{ij}. Discretizing the PDE $\frac{\partial \mathbf{h^p}}{\partial t}(t, \mathbf{x}) = \boldsymbol{\mathcal{M}}_p(\mathbf{h^p}, \mathbf{E}(\mathbf{u}))$ with the Euler numerical scheme leads to a recurrent neural network cell (PhyCell). PhyCell performs a physical prediction step in latent space (Eq 9.3) followed by a correction with embedded input data $\mathbf{E}(\mathbf{u}_t)$ (Eq 9.4), with a

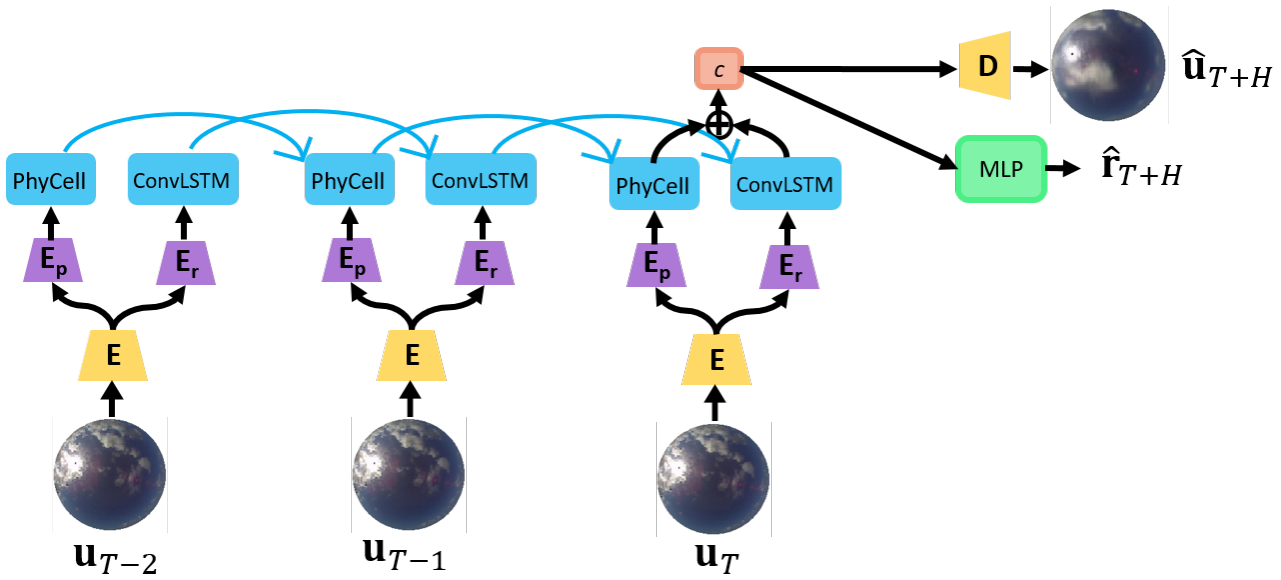


Figure 9.1: **PhyDNet-monostep architecture for solar irradiance forecasting.** Input images are embedded by an encoder $\mathbf{E}$ in a common latent space, followed by specific encoders $\mathbf{E_p}$ and $\mathbf{E_r}$ for extracting physical and residual features. PhyDNet recurrent model is unfolded in time and computes a context vector $c = \mathbf{D_p}(\mathbf{h}_T^{\mathbf{p}}) + \mathbf{D_r}(\mathbf{h}_T^{\mathbf{r}})$, which is used for predicting the future irradiance $\hat{r}_{T+H}$ and image $\hat{\mathbf{u}}_{T+H}$.

tradeoff controlled by the learned Kalman gain $\mathbf{K}_t$.

$$\tilde{\mathbf{h}}_{t+1}^{\mathbf{p}} = \mathbf{h}_t^{\mathbf{p}} + \Phi_p(\mathbf{h}_t^{\mathbf{p}}) \qquad \textbf{Prediction} \tag{9.3}$$

$$\mathbf{h}_{t+1}^{\mathbf{p}} = \tilde{\mathbf{h}}_{t+1}^{\mathbf{p}} + \mathbf{K}_t \odot \left(\mathbf{E}(\mathbf{u}_t) - \tilde{\mathbf{h}}_{t+1}^{\mathbf{p}}\right). \qquad \textbf{Correction} \tag{9.4}$$

The residual model $\boldsymbol{\mathcal{M}}_r(\mathbf{h}^{\mathbf{p}}, \mathbf{E}(\mathbf{u}))$ captures the unknown factors related to unmodelled physics, e.g. appearance, texture, and is fully learned from data (implemented by a general ConvLSTM [281]).

9.2.2 PhyDNet model with separate encoders and decoders

One limitation of PhyDNet model is that images $\mathbf{u}_t$ are embedded by the encoder $\mathbf{E}$ in a common latent space for correcting the dynamics of both physical $\mathcal{C}_p(\mathbf{h}^{\mathbf{p}}, \mathbf{E}(\mathbf{u}))$ and residual models $\mathcal{C}_r(\mathbf{h}^{\mathbf{r}}, \mathbf{E}(\mathbf{u}))$. This limits the disentangling ability of PhyDNet since $\mathbf{E}(\mathbf{u}_t)$ contains both physical and residual features. We thus propose to learn separate latent spaces for both branches, via additional specific encoders $(\mathbf{E_p}, \mathbf{E_r})$ and decoders $(\mathbf{D_p}, \mathbf{D_r})$, leading to the following dynamical model:

$$\frac{\partial \mathbf{h}(t, \mathbf{x})}{\partial t} = \boldsymbol{\mathcal{M}}_p(\mathbf{h}^{\mathbf{p}}, \mathbf{E_p} \circ \mathbf{E}(\mathbf{u})) + \boldsymbol{\mathcal{M}}_r(\mathbf{h}^{\mathbf{r}}, \mathbf{E_r} \circ \mathbf{E}(\mathbf{u})). \tag{9.5}$$

$\mathbf{E_p}$ aims at learning a specific image embedding for controlling the physical dynamics in latent space with correction features uniquely related to physics (and similarly for $\mathbf{E_r}$).

In the following, we denote this model as PhyDNet-dual.

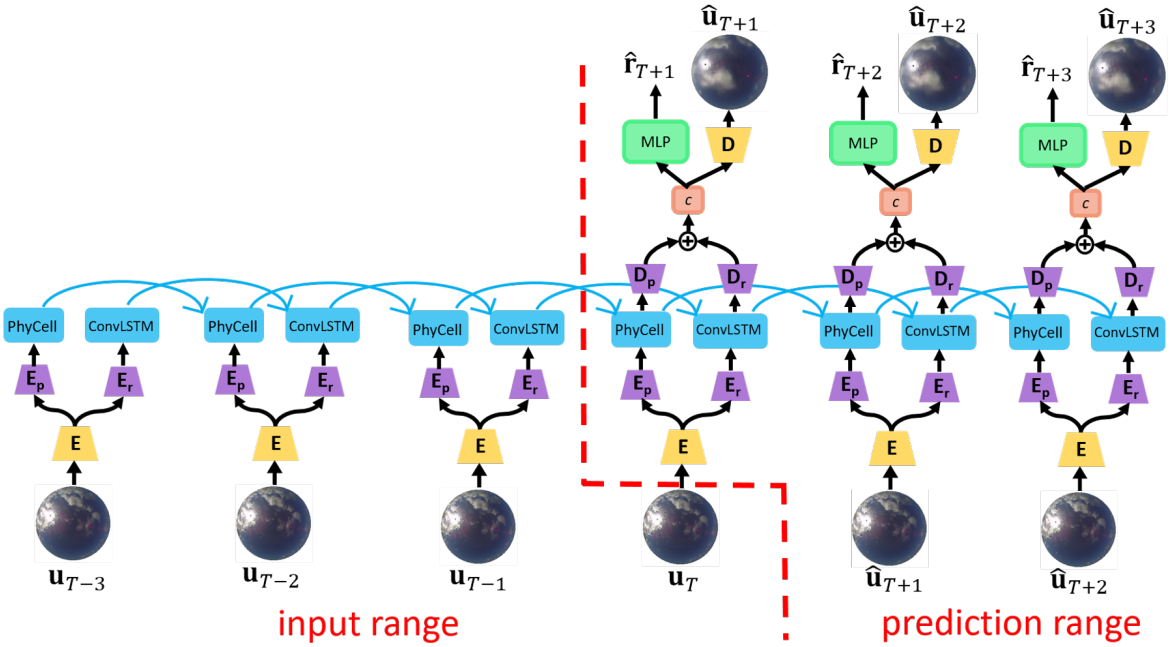


Figure 9.2: **PhyDNet-multistep architecture for solar irradiance forecasting.** This is a Sequence To Sequence architecture with the PhyDNet recurrent neural network. Contrary to PhyDNet-monostep, this model predicts the future solar irradiance and image for each time step of the prediction range.

9.2.3 PhyDNet for solar irradiance forecasting

We first propose the PhyDNet-monostep architecture, which is a direct adaptation of the forecasting model described in Chapter 8. Depicted in Figure 9.1, we replace the ConvLSTM encoding the input sequence $\mathbf{u}_{1:T}$ by the PhyDNet-dual encoder, allowing to extract physically-constrained features. The final physical and residual latent states are decoded by their respective specific decoders $\mathbf{D_p}$ and $\mathbf{D_r}$ and then summed to get a context vector $c = \mathbf{D_p}(\mathbf{h}_T^{\mathbf{p}}) + \mathbf{D_r}(\mathbf{h}_T^{\mathbf{r}})$. Then a multi-layer perceptron (MLP) uses the input context c to forecast the future irradiance $\hat{r}_{T+H}$, and the global decoder D simultaneously forecasts the future image $\mathbf{D}(c) = \hat{\mathbf{u}}_{T+H}$.

We also propose the PhyDNet-multistep shown in Figure 9.2. Instead of directly forecasting the future values from the last step of the input range, PhyDNet-multistep is composed of a PhyDNet-dual recurrent decoder. It provides future image and irradiance predictions for each time step of the prediction range $(T+1, \cdots, T+H)$. This multi-step strategy allows to supervise the model based on a whole predicted trajectory: we evaluate in the experiments the application of the DILATE training loss function instead of the MSE.

9.3 Experimental results

We conduct experiments on the same fisheye dataset as in the previous Chapter. The training dataset for solar irradiance forecasting is composed of 180,000 sequences of 10 images spaced by 1min

(with the associated ground truth solar irradiance measurements) from the years 2014 to 2016 at La Reunion Island, and the evaluation dataset of 20,000 sequences during the year 2013 on the same site. We keep 5 images for the input range and predict the 5 following images and solar irradiances. Images are resized at 80×80 pixels.

9.3.1 Irradiance forecasting with PhyDNet

We forecast solar irradiance at a 5min horizon, given a 5min past context. We compare quantitatively the proposed PhyDNet models against recent competitive video prediction baselines: ConvLSTM [281] (which corresponds to the model presented in Chapter 8) and PredRNN [268]. Each baseline is adapted in the same way for solar irradiance forecasting, in the monostep or multistep settings.

We report in Table 9.1 the normalized RMSE[1] for the predicted irradiance (KGHI) $\hat{r}_{T+5\text{min}}$.

Table 9.1: Solar irradiance (KGHI) forecasting at a 5min horizon.

	irradiance nRMSE
PhyDNet-monostep irradiance only	27.8 %
ConvLSTM-monostep [281]	26.6 %
PredRNN-monostep [268]	25.1 %
PhyDNet-monostep [139]	24.4 %
PhyDNet-dual-monostep	23.5 %
PhyDNet-dual-multistep	**21.5 %**

The first line in Table 9.1 corresponds to a PhyDNet-monostep that only predicts the future irradiance $\hat{r}_{T+5\text{min}}$ and not the future image. It gives the worst performances among compared models, indicating that the the joint image-irradiance multitask setting provides a better supervision for training the forecasting model. All the other models in Table 9.1 jointly predict future images and irradiances.

We observe that, in the monostep setting, the PhyDNet recurrent neural network gives better results (24.4%) compared to the ConvLSTM (26.6 %) and PredRNN (25.1 %). It shows that integrating physical dynamics greatly helps in modelling the cloud motion. With the separate encoders and decoders, PhyDNet-dual-monostep further improves the performances (23.5 %). Finally, we see that with the multistep strategy, PhyDNet-dual-multistep provides another large improvement (21.5 %). The supervision coming for a complete trajectory of future images and irradiances significantly boosts the training process.

We provide in Figure 9.3 a qualitative illustration of the 5min GHI predictions of the PhyDNet-dual-multistep predictions on a particular day. We see that our model closely follows the ground truth measurements and is able to successfully anticipates the sharp irradiance fluctuations, despite the fast alternation of clouds and sun.

[1] nRMSE = Root Mean Squared Error normalized by the mean value of the quantity on the train set, expressed as a percentage.

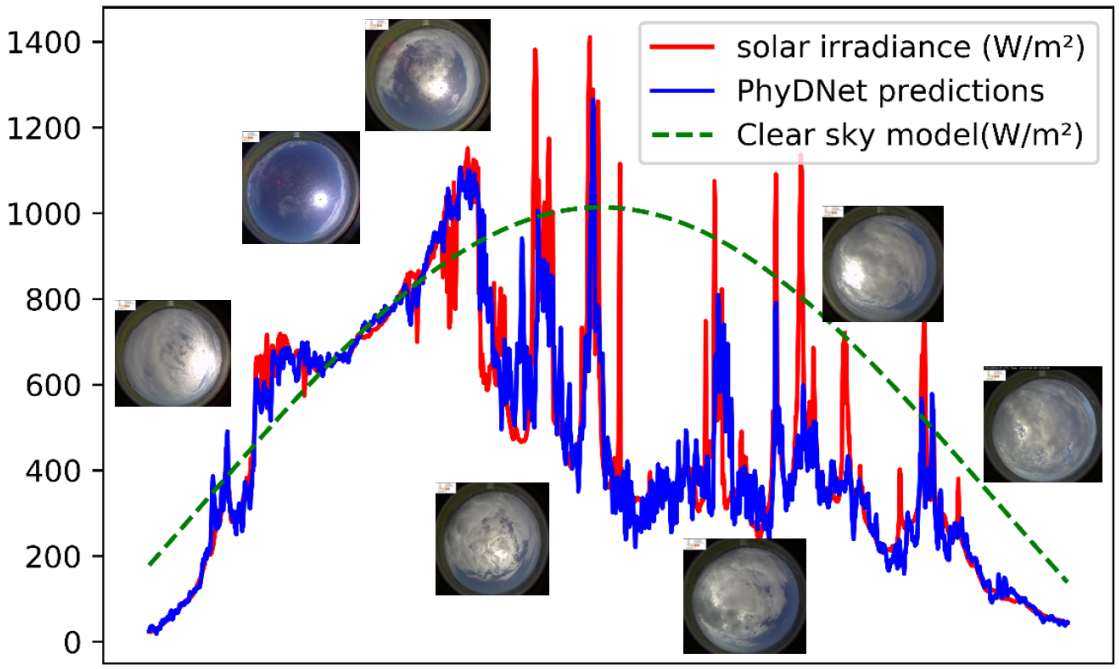


Figure 9.3: 5min ahead solar irradiance forecasts from fisheye images. Our proposed deep model leveraging physical prior knowledge accurately predicts the sharp intra-day solar irradiance fluctuations.

9.3.2 Applications of DILATE and APHYNITY

We evaluate here the application of the DILATE loss function (Chapter 4) and APHYNITY framework (Chapter 7) introduced in this thesis.

We use the DILATE loss at training time instead of the MSE for the predicted irradiance time series (5 predicted points in the future). We experimentally fixed the hyperparameter α balancing the shape and temporal term to 0.95, which yields the best results.

For APHYNITY, we minimize the norm of the residual hidden state $\mathbf{h^r}$ for all time steps. Note that contrary to the APHYNITY models presented in Chapter 7, we do not use here the NeuralODE for extrapolating the trajectory in latent space, but the PhyDNet recurrent neural network. Exploiting a NeuralODE integration is a promising way for future works.

Forecasting results are presented in Table 9.2. We compare the application of DILATE, APHYNITY and the combination of both mechanisms. We can see that these 3 variants lead to approximately similar performances: they improve slightly over the PhyDNet-dual-multistep baseline in normalized RMSE and in the DILATE objective (confirmed with the shape and temporal metrics).

Discussion The performance improvement due to DILATE and APHYNITY exists, but is rather small compared to the performance gap due to the architecture design of PhyDNet-dual and to the multistep training scheme. We discuss here the possible reasons. Concerning DILATE, we apply the loss in our experiments on predicted trajectories of 5 timesteps. This is rather small compared to our experiments in Chapter 4 (the shortest trajectories have 20 timesteps for the `Synthetic` dataset). For

shorter trajectories, dynamic time warping is less relevant, and the sharp variations are more difficult to visualize. Augmenting the forecasting horizon of our method and reducing the time interval between images (up of the 10s sampling frequency) are interesting future directions for better exploiting the DILATE loss.

Regarding APHYNITY, the physical model used in PhyDNet is a class of linear PDEs. This is a very coarse physical prior, more general than in the experiments presented in Chapter 7. Moreover, due to the non-observed prior, the physical model is applied in a learned latent space which is not explicitly controlled, contrary to the fully-visible setting in Chapter 7. This may explain why optimizing the ML/MB decomposition leads to less improvement. An appealing future direction would be to exploit more specific physical laws modelling the cloud motion and/or a more precise description of the input space where the physical laws apply.

Table 9.2: Evaluation of the DILATE loss and the APHYNITY framework on the 5-min solar irradiance forecasting problem.

	nRMSE	DTW	TDI	DILATE	Ramp score
PhyDNet-dual-multistep	21.5 %	34.1	63.3	97.4	78.6
DILATE	**21.2 %**	**33.6**	63.0	96.6	**77.3**
APHYNITY	21.4 %	34.2	62.2	96.4	**77.3**
APHYNITY + DILATE	**21.2 %**	**33.6**	**61.5**	**95.1**	77.9

9.3.3 Video prediction

We then evaluate PhyDNet-dual-multistep on the video prediction task. Given 5 input images with a 1 min interval, we forecast the 5 future images up to $t_0 + 5$min. We compare PhyDNet-dual-multistep with ConvLSTM and Memory In Memory (MIM) [269]. Evaluation metrics are the mean squared error (MSE), mean absolute error (MAE) and the structural similarity index SSIM (higher is better). Results shown in Table 9.3 reveal that PhyDNet-dual-multistep outperforms both baselines for all metrics. It confirms that incorporating physical prior information for modelling cloud motion is beneficial compared to fully data-driven algorithms.

Table 9.3: Quantitative video prediction results.

	MSE	MAE	SSIM
ConvLSTM [281]	83.1	681	0.845
MIM [269]	68.6	635	0.840
PhyDNet-dual-multistep	**68.1**	**629**	**0.862**

We show in Figure 9.4 a video prediction example of PhyDNet-dual model. The future of this sequence presents 2 clouds (circled in blue and green) moving closer between t_0 and $t_0 + 3$min and finally merging at time $t_0 + 4$min. We observe that PhyDNet-dual predicts the same outcome with a good accuracy on cloud location, although clouds become blurry because of uncertainty.

In Figure 9.5, we provide a particular comparison to ConvLSTM [281], which forms the residual

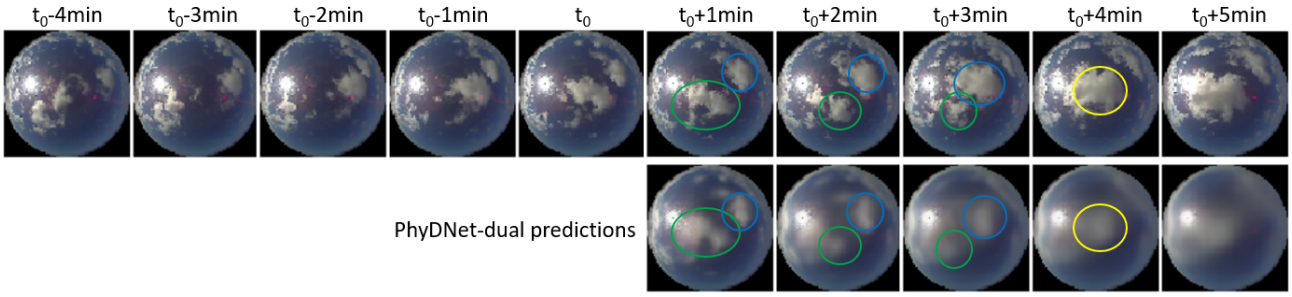


Figure 9.4: Qualitative fisheye video forecasting results up to 5min horizon. The proposed model successfully predicts the motion of the blue and green clouds that move nearer and finally merge into the yellow cloud.

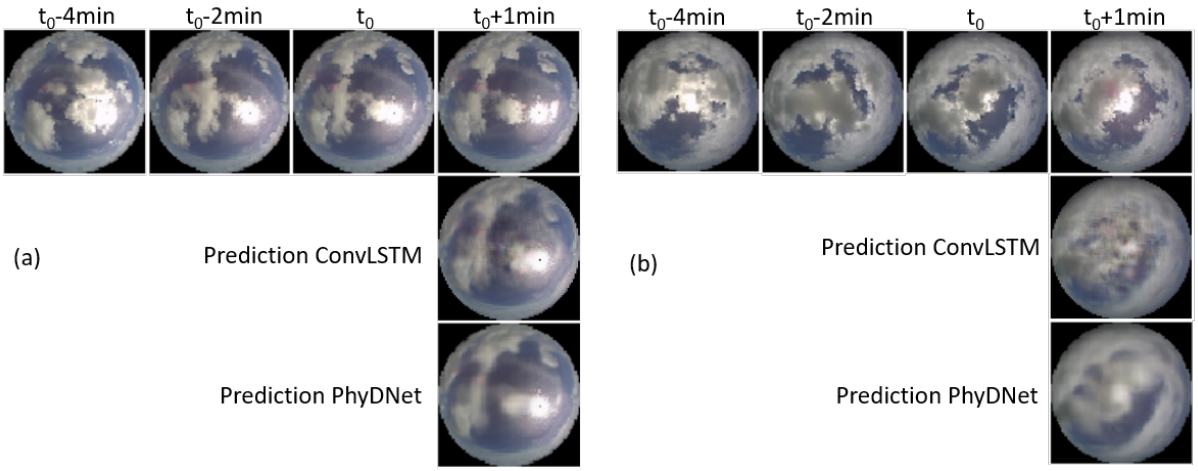


Figure 9.5: Qualitative forecasting comparison between PhyDNet-dual-multistep and ConvLSTM.

branch of PhyDNet. In sequence (a), we see that the shape of the small cloud getting nearer the sun is much better predicted by PhyDNet-dual. In sequence (b), the sun will reappear 1 min in the future. PhyDNet-dual provides a better anticipation by prediction a bright spot at the sun location and better defined cloud shapes. It confirms that incorporating physical dynamics greatly improves the predictions of natural phenomena, with a small amount of additional parameters with respect to ConvLSTM.

9.4 Conclusion

In this Chapter, we have explored the methodological contributions of this thesis for solving the solar irradiance forecasting problem at EDF. We have proposed an improvement of our PhyDNet video prediction model that we have adapted for this task. The PhyDNet model greatly improves the performances compared to competitive pure data-driven, confirming the benefits of the MB/ML integration. We have also highlighted the crucial importance of making multistep instead of monostep predictions. Furthermore, we have applied the DILATE loss function and the APHYNITY framework, which further improve the forecasting performances, albeit slightly.

Chapter 10

Conclusion and perspectives

Content

10.1 Summary of contributions

From a general perspective, we have explored in this thesis how to incorporate prior knowledge into machine learning for improving spatio-temporal forecasting models. More specifically, we have studied two important scientific challenges.

10.1.1 Multistep forecasting of non-stationary dynamics

In many real-world applications, time series present non-stationary dynamics with possible sharp variations, e.g. traffic flows, financial stocks, or solar irradiance time series. Current state-of-the art deep learning methods for multistep deterministic and probabilistic forecasting struggle to properly predict these abrupt events: their predictions often smooth the sharp variations and/or present a temporal missalignment. One of the reasons is that most works focus on neural network architecture design and overlook the choice of the training loss function. The dominantly used loss function is the mean squared error (MSE), that is unable to take into account global information about the multistep dynamics.

In this thesis, we have shown that this is possible to design dedicated multistep loss functions to impose a certain desired behaviour to the output. For time series, we focus on shape and temporal

criteria that are commonly used as assessment metrics in applications. In Chapter 3, we have drawn a panorama of shape and temporal criteria based on smooth approximations of Dynamic Time Warping (DTW) and Time Distortion Index (TDI). We have expressed them both as dissimilarities (loss functions) and similarities (positive semi-definite kernels). We have insisted on their differentiability, which is an important requirement for training models with gradient-based optimization, and propose optimized implementations of these losses for efficient back-propagation training.

We have then applied the proposed shape and time differentiable criteria to two spatio-temporal forecasting contexts. In Chapter 4, we have introduced a differentiable loss function (DILATE), that combines a shape term and a temporal term, for training any deep forecasting model to produce multistep deterministic forecasts. We have shown that training with DILATE produces sharper predictions with a better temporal localization than training with the standard MSE, while maintaining the performances with MSE evaluation.

In Chapter 5, we have proposed the STRIPE model for probabilistic forecasting. In order to produce a limited set of possible scenarios that reflect the shape and temporal variability of ground truth trajectories, the STRIPE model is equipped with a diversification mechanism that structures the output diversity. This is done with a diversity loss relying on determinantal point processes (DPP), with two shape and temporal criteria introduced in Chapter 3. STRIPE leads to more diverse forecasts according to shape and temporal criteria without sacrificing on quality. We have also revealed the crucial importance to decouple the criteria used for quality and diversity.

10.1.2 Exploiting incomplete prior physical knowledge in machine learning models

The extrapolation task underlying spatio-temporal forecasting is quite different and much more challenging for pure data-driven methods than the perception tasks at the origin of the impressive success of deep learning. For example, forecasting complex natural dynamics such as climate remains out of the scope of pure machine learning. An appealing solution is to incorporate external physical knowledge, which is an old research problem that is still open today. In this thesis, we have particularly focused on exploiting *incomplete* physical knowledge, in contrast to mainstream methods that suppose a full prior knowledge. The incomplete case can stem from the difficulty of the phenomenon that remains elusive to a complete description from physical laws, e.g. for modelling all the complex interacting phenomena for predicting the evolution of the atmosphere, or from a non-observable prior context, i.e. when the dynamical model does not apply directly in the input space.

In Chapter 6, we have tackled the problem of generic video prediction. It is an example of a non-observable prior context: although there often exists some physical dynamical prior, for example on the motion of clouds in fisheye images, physical laws do not directly apply at the pixel level. The dynamical model is meaningful in a space where the clouds have previously been identified and segmented. We have introduced the PhyDNet prediction model that automatically learns a latent space in which we suppose that a class of linear partial differential equations apply. PhyDNet is a two-branch architecture: the first branch captures the physical dynamics. Since this prior knowledge is often insufficient to fully describe the content of videos, PhyDNet is composed of a second branch for modelling the complementary information necessary for accurate prediction (e.g. texture, details, *etc*). We have highlighted the ability of PhyDNet to properly disentangle the physical dynamics from these unknown factors.

In Chapter 7, we have further delved into the question of augmenting incomplete physical models

with deep data-driven counterparts. This is an area that has been explored by very few works up to now, and mostly empirically. We have proposed the APHYNITY framework, that consists in decomposing the dynamics in two components: a physical component accounting for the dynamics for which we have some prior knowledge, and a data-driven component accounting for insufficiencies of the physical model. APHYNITY is a principled learning framework minimizing the norm of the data-driven augmentation, that theoretically guarantees a unique decomposition under mild assumptions. APHYNITY is able to seamlessly adapt to different approximation levels of prior physical knowledge, covering the whole range of Machine Learning /Model-Based methods presented in Chapter 1. We have exhibited 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.

10.1.3 Solar irradiance forecasting with fisheye images

Finally, we have proposed solutions to the industrial solar irradiance forecasting problem with fisheye images raised at EDF. In Chapter 8, we have presented the challenges of the problem and proposed a first deep learning model for estimating and forecasting solar irradiance. We have also discussed the limitations of standard deep learning forecasting approaches in this context, that have motivated the contributions of this thesis.

In Chapter 9, we have applied the methodological contributions exposed in parts I and II of this thesis. We have improved and adapted our PhyDNet model for physically-constrained fisheye image prediction. The PhyDNet model greatly improves the performances compared to competitive pure data-driven, confirming the benefits of the physical knowledge integration. Furthermore, we have applied the DILATE loss function and the APHYNITY framework, leading to another (relatively small) performance gain.

10.2 Perspectives

We present here a non-exhaustive list of possible future research directions for different time horizons.

10.2.1 Directions for improving solar irradiance forecasting

Application of DILATE and APHYNITY As discussed in Chapter 9, the main performance improvements compared to pure deep learning methods stem from the application of our physically-constrained PhyDNet architecture. The application of the DILATE loss and the APHYNITY framework further improve the performances, but less significantly.

Concerning the DILATE loss function, we have applied in our experiments the loss on future trajectories of 5 timesteps, which is rather small compared to the experiments in Chapter 4 (the shortest trajectories have 20 timesteps for the `Synthetic` dataset). For short trajectories, the sharp variations are harder to visualize and the use of dynamic time warping (DTW) is less relevant. To fully exploit the capacity of the DILATE loss, an interesting perspective is to augment the length of future trajectories, by reducing the processing interval between images or by augmenting the forecasting

horizon.

Regarding APHYNITY, we use in the PhyDNet model a very general physical prior model: a class of linear PDEs. This is a weaker prior than those used in Chapter 7. Moreover, due to the non-observability of the prior, the physical model is applied in a learned latent space which is not explicitly controlled, contrary to the fully-visible setting in Chapter 7. This may explain why the Machine Learning / Model Based decomposition is more challenging to optimize. An interesting future direction would be to exploit more specific physical laws modelling the cloud motion and/or a more precise description of the input space where the physical laws apply.

Probabilistic forecasting In this thesis, we have forecasted solar irradiance in a deterministic manner with the PhyDNet model. An interesting future work is to extend our contributions on probabilistic forecasting to this problem. An adaptation of the STRIPE model would provide to the decision makers a small set of possible scenarios about the cloud motion (for example if the clouds will occlude the sun or not, and at what temporal horizon).

Handling the rotational distortion of fisheye images Fisheye images present a rotational symmetry along the vertical axis. Clouds in linear translation are observed as a curved motion in fisheye images. To handle this distortion induced by the fisheye camera objective, some forecasting methods preprocess fisheye images by projecting them in a plane where a translational cloud motion is linear. In this thesis, we have instead directly processed raw fisheye images with general convolutional layers commonly used in computer vision for encoding translation equivariance. Future works include applying the plane projection or polar transformation [185] as preprocessing, or evaluating more dedicated neural network layers that handle rotation equivariance, such as spherical CNNs [46, 47].

10.2.2 Applications of deep augmented physical models

Non-stationary dynamics forecasting

In this thesis, our contributions towards non-stationary dynamics forecasting concern rethinking the training process by including shape and temporal criteria, and are thus agnostic to the forecasting architectures. An interesting future perspective would be to also incorporate prior knowledge in the model architectures, as studied in part II of this thesis. For time series, leveraging trend, seasonality and extrinsic prior knowledge (such as special events) [134] could help to better model the non-stationary abrupt changes and measure their impact on diversity and model confidence [84, 49]. The combination between a traditional forecasting model with interpretable and controlled factors (e.g. a ARIMA model) and a data-driven augmentation network would be a possible application case for APHYNITY.

Optical flow

Optical flow estimation is a long-standing problem in computer vision, consisting in estimating the motion field between two frames. It is a core building block for many applications, such as image compression or object tracking. For example, optical flow is used to understand the cloud motion in traditional forecasting methods with fisheye images.

Traditional methods for optical flow, e.g. the Lucas-Kanade [160] and the Horn-Schunk [103] models, are based on the brightness constancy assumption $I_1(\mathbf{x}) = I_2(\mathbf{x}+w)$ that states that the pixel intensity is preserved after advection by the flow field w. Linearising this equation leads to the celebrated optical flow PDE:

$$\frac{\partial I}{\partial t}(t,\mathbf{x}) = -w(t,\mathbf{x}) \cdot \nabla I(t,\mathbf{x}). \tag{10.1}$$

The PDE in Eq 10.1 is a simplified physical model, since the brightness constancy assumption is violated in several conditions, e.g. in presence of occluded objects, local, global illumination changes or specular reflexions.

Other traditional methods exploit different prior physical models for optical flow in specific contexts, e.g. the PDE continuity equation for fluid flows [51].

More recently, deep learning approaches have proposed learning optical flow in an end-to-end fashion and have become state-of-the-art [67, 238, 243, 237]. Two classes of methods exist: supervised and unsupervised ones. In the supervised context [67, 238, 243], deep learning methods do not exploit the brightness constancy hypothesis anymore, or indirectly (through the computation of a cost volume). Instead, they rely on large synthetic datasets of annotated image pairs, making their generalization to real-world datasets not obvious.

On the other side, unsupervised deep learning approaches [109, 151, 237] are closer in spirit to traditional approaches. Without ground truth labels for optical flow, they rely on a photometric reconstruction loss. The reason deep unsupervised methods outperform traditional methods is that they fully exploit the correlations from the training dataset, instead of independently optimizing a flow field for each image pair. Typical photometric losses include the L1 loss that directly assumes intensity constancy, or more robust losses such the Charbonnier loss, the structural similarity (SSIM) [115] or the census loss [169] that is robust to global illumination changes. Although adequate losses may address some limitations of the brightness constancy assumption, they do not overcome all failure cases. Therefore the photometric constancy assumption also represents a simplified physical model.

In this context, an appealing research perspective is to explicitly exploit the simplified optical flow PDE in Eq 10.1 in a deep augmented model. This is a favorable case for the application of our APHYNITY framework. This ML/MB integration could regularize and boost the performances of deep supervised estimation models, in particular for generalizing to new datasets. It could also be applied in a semi-supervised context, where the learned data-driven augmentation could complement the simplified photometric constancy for non-annotated images.

Model-Based Reinforcement Learning

Reinforcement Learning (RL) [241] is a branch of machine learning that studies how autonomous agents make decisions in the environment in order to maximize their cumulative reward. Combined with deep learning, RL has encountered impressive successes for example by reaching super-human performance at the game of Go [230].

There are two main modelling approaches in RL: *model-based* and *model-free*. In the model-based approach, the agent uses an internal predictive model of the world to simulate the consequences of its actions, and choose the best action accordingly. In contrast, in the model-free approach, the control policy is learned directly from experienced trajectories, without any dynamical model.

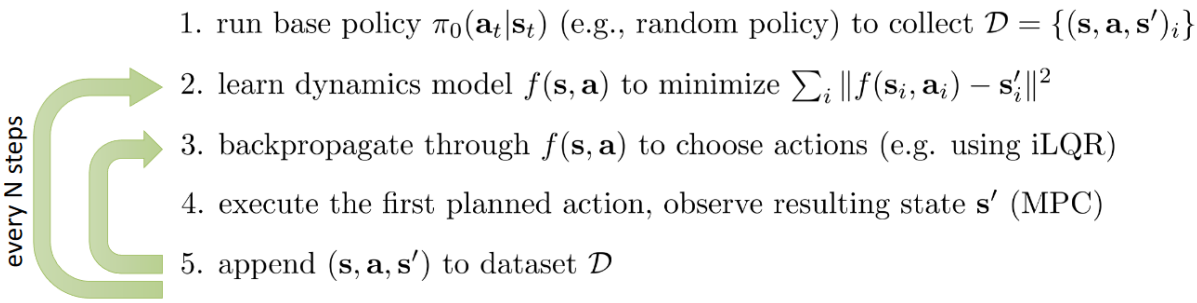


Figure 10.1: Principle of Model-Based Reinforcement Learning. Illustration from Sergey Levine[1].

The principle of Model-Based Reinforcement Learning (MBRL) is illustrated in Figure 10.1. It consists in planning through a dynamical model $f(\mathbf{s}_t, \mathbf{a}_t)$, where $\mathbf{s}_t$ is the current state and $\mathbf{a}_t$ the chosen action. The dynamical model is learned to minimize the future (discounted) cumulative cost:

$$\min_{a_{t_0},\dots,a_\infty} \quad \sum_{t=t_0}^{\infty} \gamma^{t-t_0} c(\hat{\mathbf{s}}_t, \mathbf{a}_t) \quad \text{subject to} \quad \forall t \geq t_0, \frac{\mathrm{d}\mathbf{s}_t}{\mathrm{d}t} = f(\mathbf{s}_t, \mathbf{a}_t). \tag{10.2}$$

where c is a cost function and $\gamma < 1$ a discount factor.

The dynamical model f can be a simple linear (or locally linear) model, a physical model, or a pure data-driven model parameterized by a deep neural network[2]. In all cases, the model f is often too simplified to perfectly extrapolate the future trajectories.

A common solution for nonetheless exploiting the incomplete model is to consider short-term rollouts and perform Model Predictive Control (MPC) [178, 108], which consists in replanning frequently to mitigate the error propagation in the forecasted trajectories.

An interesting future direction would be to explore deep augmented models in this MBRL case. A simplified prior dynamical model of the system could be augmented with a data-driven counterpart and learned together with the APHYNITY framework. This cooperation could improve the accuracy of the predictive model, enabling to perform more truthworthy long-term rollouts, and to replan less frequently.

An other appealing direction concerns improving the exploration process in Reinforcement Learning with a diversity-promoting mechanism [189, 74, 145]; this mechanism could be implemented with determinantal point processes with adequate kernels to represent structured diversity.

10.2.3 Long-term perspectives

The field of spatio-temporal forecasting is still a very active area of research in the AI community, and has not reached yet the degree of maturity of deep learning in computer vision or language. Forecasting complex dynamics remains highly challenging for pure machine learning, due to the relative current scarcity of data for learning complex natural phenomena such as climate. The quantity of

[2]Please note that in the RL community, the term *model-based* denotes the presence of a dynamical model f, that can either be a pure data-driven model (denoted as *Machine Learning* in this thesis) or a model with a physical prior (denoted as *Model-Based* in this thesis).

training data will likely continue to grow in future years, yet it is not clear at which point it will become sufficient. Relying on this growing data accumulation, the exploration of bigger and bigger models to overcome the underfitting phenomenon is a possible way, which is faced with many computational challenges.

The other way, which was explored in this thesis, is to incorporate external knowledge to regularize machine learning models, in the form of loss functions, model architectures or training strategies. We hope that the contributions of this thesis will open the way towards hybrid and more flexible Machine Learning/Model-Based models for tackling complex real-world applications, e.g. in climate science, robotics or reinforcement learning. In particular, the augmentation strategy explored in this thesis - a linear combination - is rather particular. For many incomplete models, there exists high-order interactions between the simplified model and the residual information. Exploring more general augmentations schemes, linked with the growing field of neural architecture search [72], is an appealing direction for future years.

Bibliography

[1] Abubakar Abid and James Zou. Learning a warping distance from unlabeled time series using sequence autoencoders. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[2] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning (ICML)*, pages 173–182. PMLR, 2016.

[3] Mark Asch, Marc Bocquet, and Maëlle Nodet. *Data assimilation: methods, algorithms, and applications*, volume 11. SIAM, 2016.

[4] Ibrahim Ayed, Nicolas Cedilnik, Patrick Gallinari, and Maxime Sermesant. EP-Net: Learning cardiac electrophysiology models for physiology-based constraints in data-driven predictions. In *10th International Conference on Functional Imaging and Modeling of the Heart*, volume 11504, pages 55–63. Springer, 2019.

[5] Ibrahim Ayed, Emmanuel de Bézenac, Arthur Pajot, Julien Brajard, and Patrick Gallinari. Learning dynamical systems from partial observations. *arXiv preprint arXiv:1902.11136*, 2019.

[6] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. *International Conference on Learning Representations (ICLR)*, 2018.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*, 2015.

[8] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:690–701, 2019.

[9] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4502–4510, 2016.

[10] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[11] Philipp Becker, Harit Pandya, Gregor Gebhardt, Cheng Zhao, C James Taylor, and Gerhard Neumann. Recurrent Kalman networks: Factorized inference in high-dimensional deep feature spaces. In *International Conference on Machine Learning (ICML)*, pages 544–552, 2019.

[12] Xinzhu Bei, Yanchao Yang, and Stefano Soatto. Learning semantic-aware dynamics for video prediction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 902–912, 2021.

[13] Richard Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences of the United States of America*, 38(8):716, 1952.

[14] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[15] Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods (Optimization and Neural Computation Series)*. Athena Scientific, 1 edition, 1996.

[16] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[17] Mathieu Blondel, Arthur Mensch, and Jean-Philippe Vert. Differentiable divergences between time series. *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

[18] Marc Bocquet, Julien Brajard, Alberto Carrassi, and Laurent Bertino. Data assimilation as a learning tool to infer ordinary differential equation representations of dynamical models. *Nonlinear Processes in Geophysics*, 26(3):143–162, 2019.

[19] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[20] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[21] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[22] EH Bristol. Swinging door trending: Adaptive trend recording? In *ISA National Conf. Proc., 1990*, pages 749–754, 1990.

[23] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[24] John Charles Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016.

[25] Wonmin Byeon, Qin Wang, Rupesh Kumar Srivastava, and Petros Koumoutsakos. ContextVP: Fully context-aware video prediction. In *European Conference on Computer Vision (ECCV)*, pages 753–769, 2018.

[26] Robert Stephen Cantrell and Chris Cosner. *Spatial ecology via reaction-diffusion equations*. John Wiley & Sons, 2004.

[27] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pages 213–229. Springer, 2020.

[28] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional VRNNs for video prediction. In *International Conference on Computer Vision (ICCV)*, 2019.

[29] Kanad Chakraborty, Kishan Mehrotra, Chilukuri K Mohan, and Sanjay Ranka. Forecasting the behavior of multivariate time series using neural networks. *Neural networks*, 5(6):961–970, 1992.

[30] Bo Chang, Minmin Chen, Eldad Haber, and Ed H Chi. Antisymmetricrnns: a dynamical system view on recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[31] Xiaobin Chang, Frederick Tung, and Greg Mori. Learning discriminative prototypes with dynamic time warping. In *Computer Vision and Pattern Recognition (CVPR)*, pages 8395–8404, 2021.

[32] Sucheta Chauhan and Lovekesh Vig. Anomaly detection in ecg time signals via deep long short-term memory networks. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2015.

[33] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[34] Wen-Hua Chen. Disturbance observer based control for nonlinear systems. *IEEE/ASME transactions on mechatronics*, 9(4):706–710, 2004.

[35] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The UCR time series classification archive. 2015.

[36] Yitian Chen, Yanfei Kang, Yixiong Chen, and Zizhuo Wang. Probabilistic forecasting with temporal convolutional neural network. *Neurocomputing*, 399:491–501, 2020.

[37] Zhengdao Chen, Jianyu Zhang, Martin Arjovsky, and Léon Bottou. Symplectic recurrent neural networks. *International Conference on Learning Representations (ICLR)*, 2020.

[38] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

[39] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[40] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[41] Chi Wai Chow, Bryan Urquhart, Matthew Lave, and Anthony et al Dominguez. Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed. *Solar Energy*, 2011.

[42] Yinghao Chu, Mengying Li, and C Coimbra. Suntracking imaging system for intra-hour DNI forecasts. *Renewable Energy*, 2016.

[43] Yinghao Chu, Hugo TC Pedro, and Carlos FM Coimbra. Hybrid intra-hour dni forecasts with sky image processing enhanced by stochastic learning. *Solar Energy*, 98:592–603, 2013.

[44] Jessica M. Chung and Enrique Peacock-López. Bifurcation diagrams and turing patterns in a chemical self-replicating reaction-diffusion system with cross diffusion. *The Journal of Chemical Physics*, 127(17):174903, 2007.

[45] Michael A Cohen and Stephen Grossberg. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE transactions on systems, man, and cybernetics*, (5):815–826, 1983.

[46] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning (ICML)*, pages 2990–2999. PMLR, 2016.

[47] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. In *International Conference on Learning Representations (ICLR)*, 2018.

[48] Jerome T Connor, R Douglas Martin, and Les E Atlas. Recurrent neural networks and robust time series prediction. *IEEE Transactions on Neural Networks*, 5(2):240–254, 1994.

[49] Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. Addressing failure prediction by learning model confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[50] Thomas Corpetti, Patrick Héas, Etienne Mémin, and Nicolas Papadakis. Pressure image assimilation for atmospheric motion estimation. *Tellus A: Dynamic Meteorology and Oceanography*, 61(1):160–178, 2009.

[51] Thomas Corpetti, Étienne Mémin, and Patrick Pérez. Dense estimation of fluid flows. *IEEE Transactions on pattern analysis and machine intelligence*, 24(3):365–380, 2002.

[52] Philippe Courtier, J-N Thépaut, and Anthony Hollingsworth. A strategy for operational implementation of 4d-var, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society*, 120(519):1367–1387, 1994.

[53] Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks. *ICLR 2020 Deep Differential Equations Workshop*, 2020.

[54] Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020.

[55] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International Conference on Machine Learning (ICML)*, pages 894–903, 2017.

[56] Marco Cuturi, Jean-Philippe Vert, Oystein Birkenes, and Tomoko Matsui. A kernel for time series based on global alignments. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2007.

[57] Arka Daw, R Quinn Thomas, Cayelan C Carey, Jordan S Read, Alison P Appling, and Anuj Karpatne. Physics-guided architecture (pga) of neural networks for quantifying uncertainty in lake temperature modeling. In *Proceedings of the 2020* SIAM *International Conference on Data Mining*, pages 532–540. SIAM, 2020.

[58] Emmanuel de Bezenac, Arthur Pajot, and Patrick Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. *International Conference on Learning Representations (ICLR)*, 2018.

[59] Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. Normalizing kalman filters for multivariate time series analysis. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[60] Emily L Denton et al. Unsupervised learning of disentangled representations from video. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4414–4423, 2017.

[61] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[62] Maimouna Diagne, Mathieu David, Philippe Lauret, John Boland, and Nicolas Schmutz. Review of solar irradiance forecasting methods and a proposition for small-scale insular grids. *Renewable and Sustainable Energy Reviews*, 27:65–76, 2013.

[63] Adji B Dieng, Francisco JR Ruiz, David M Blei, and Michalis K Titsias. Prescribed generative adversarial networks. *arXiv:1910.04302*, 2019.

[64] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[65] Bin Dong, Qingtang Jiang, and Zuowei Shen. Image restoration: Wavelet frame shrinkage, nonlinear evolution PDEs, and beyond. *Multiscale Modeling & Simulation*, 15(1):606–660, 2017.

[66] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[67] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision (ICCV)*, 2015.

[68] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

[69] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[70] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*. Oxford university press, 2012.

[71] Mohamed Elfeki, Camille Couprie, Morgane Riviere, and Mohamed Elhoseiny. GDPP: learning diverse generations using determinantal point process. *International Conference on Machine Learning (ICML)*, 2019.

[72] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

[73] SM Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Geoffrey E Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3225–3233, 2016.

[74] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *International Conference on Learning Representations (ICLR)*, 2019.

[75] Ronan Fablet, Said Ouala, and Cédric Herzet. Bilinear residual neural network for the identification and forecasting of geophysical dynamics. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1477–1481. IEEE, 2018.

[76] Chenyou Fan, Yuze Zhang, Yi Pan, Xiaoyue Li, Chi Zhang, Rong Yuan, Di Wu, Wensheng Wang, Jian Pei, and Heng Huang. Multi-horizon time series forecasting with temporal attention learning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2019.

[77] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 64–72, 2016.

[78] James Fletcher and Warren Moors. Chebyshev sets. *Journal of the Australian Mathematical Society*, 98:161–231, 04 2014.

[79] Anthony Florita, Bri-Mathias Hodge, and Kirsten Orwig. Identifying wind and solar ramping events. In *IEEE GreenTech Conference*, 2013.

[80] Ian Fox, Lynn Ang, Mamta Jaiswal, Rodica Pop-Busui, and Jenna Wiens. Deep multi-output forecasting: Learning to accurately predict blood glucose trajectories. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2018.

[81] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3601–3610, 2017.

[82] Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari. Stochastic latent residual video prediction. *International Conference on Machine Learning (ICML)*, 2020.

[83] Laura Frías-Paredes, Fermín Mallor, Martín Gastón-Romeo, and Teresa León. Assessing energy forecasting inaccuracy by simultaneously considering temporal and absolute errors. *Energy Conv. and Management*, 2017.

[84] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1050–1059, 2016.

[85] Hang Gao, Huazhe Xu, Qi-Zhi Cai, Ruth Wang, Fisher Yu, and Trevor Darrell. Disentangling propagation and generation for video prediction. In *International Conference on Computer Vision (ICCV)*, pages 9006–9015, 2019.

[86] Jan Gasthaus, Konstantinos Benidis, Yuyang Wang, Syama Sundar Rangapuram, David Salinas, Valentin Flunkert, and Tim Januschowski. Probabilistic forecasting with spline quantile function RNNs. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[87] Charlotte Gauchet, Philippe Blanc, Bella Espinar, Bruno Charbonnier, and Dominique Demengel. Surface solar irradiance estimation with low-cost fish-eye camera. In *COST WIRE*, 2012.

[88] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis. Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, 45(11):5742–5751, 2018.

[89] Amir Ghaderi, Borhan M Sanandaji, and Faezeh Ghaderi. Deep forecast: Deep learning-based spatio-temporal forecasting. In *ICML Time Series Workshop*, 2017.

[90] Jennifer A Gillenwater, Alex Kulesza, Emily Fox, and Ben Taskar. Expectation-maximization for learning determinantal point processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[91] Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2007.

[92] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[93] Raul González-García, Ramiro Rico-Martìnez, and Ioannis G Kevrekidis. Identification of distributed parameter systems: A neural net based approach. *Computers & chemical engineering*, 22:S965–S968, 1998.

[94] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[95] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations (ICLR)*, 2018.

[96] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 15353–15363, 2019.

[97] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[98] Tuomas Haarnoja, Anurag Ajay, Sergey Levine, and Pieter Abbeel. Backprop KF: Learning discriminative deterministic state estimators. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4376–4384, 2016.

[99] Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse problems*, 34(1):014004, 2017.

[100] Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. Liquid time-constant networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7657–7666, 2021.

[101] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[102] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[103] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[104] Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Niebles. Learning to decompose and disentangle representations for video prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 517–526, 2018.

[105] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, 2017.

[106] Rob Hyndman, Anne B Koehler, J Keith Ord, and Ralph D Snyder. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

[107] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2013.

[108] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12519–12530, 2019.

[109] J Yu Jason, Adam W Harley, and Konstantinos G Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *European Conference on Computer Vision (ECCV)*, pages 3–10. Springer, 2016.

[110] Young-Seon Jeong, Myong K Jeong, and Olufemi A Omitaomu. Weighted dynamic time warping for time series classification. *Pattern Recognition*, 2011.

[111] Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:9847–9858, 2019.

[112] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 29:667–675, 2016.

[113] Pedro A Jimenez, Joshua P Hacker, Jimy Dudhia, Sue Ellen Haupt, Jose A Ruiz-Arias, Chris A Gueymard, Gregory Thompson, Trude Eidhammer, and Aijun Deng. Wrf-solar: Description and clear-sky assessment of an augmented nwp model for solar power prediction. *Bulletin of the American Meteorological Society*, 97(7):1249–1264, 2016.

[114] Gordon G Johnson. A nonconvex set which has the unique nearest point property. *Journal of Approximation Theory*, 51(4):289 – 332, 1987.

[115] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *European Conference on Computer Vision (ECCV)*, pages 557–572. Springer, 2020.

[116] RE Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, D*, 82:35–44, 1960.

[117] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[118] Diederik P Kingma and Jimmy Lei Ba. Adam: Amethod for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2014.

[119] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[120] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning (ICML)*, pages 2693–2702, 2018.

[121] Gene A. Klaasen and William C. Troy. Stationary wave solutions of a system of reaction-diffusion equations derived from the fitzhugh–nagumo equations. *SIAM Journal on Applied Mathematics*, 44(1):96–110, 1984.

[122] Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15(4):143–156, 2001.

[123] Simon Kohl, Bernardino Romera-Paredes, Clemens Meyer, Jeffrey De Fauw, Joseph R. Ledsam, Klaus Maier-Hein, S. M. Ali Eslami, Danilo Jimenez Rezende, and Olaf Ronneberger. A probabilistic u-net for segmentation of ambiguous images. In *International Conference on Artificial Intelligence and Statistics (NeurIPS)*, 2018.

[124] Alireza Koochali, Andreas Dengel, and Sheraz Ahmed. If you like it, GAN it. probabilistic multivariate times series forecast with GAN. *arXiv preprint arXiv:2005.01181*, 2020.

[125] Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8606–8616, 2018.

[126] Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

[127] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1097–1105, 2012.

[128] Chung-Ming Kuan and Tung Liu. Forecasting exchange rates using feedforward and recurrent neural networks. *Journal of applied econometrics*, 10(4):347–364, 1995.

[129] Pascal Kuhn, Bijan Nouri, Stefan Wilbert, Christoph Prahl, Nora Kozonek, Thomas Schmidt, Zeyad Yasser, Lourdes Ramirez, Luis Zarzalejo, Angela Meyer, et al. Validation of an all-sky imager–based nowcasting system for industrial pv plants. *Progress in Photovoltaics: Research and Applications*, 26(8):608–621, 2018.

[130] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012.

[131] Vitaly Kuznetsov and Zelda Mariet. Foundations of sequence-to-sequence modeling for time series. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[132] Yong-Hoon Kwon and Min-Gyu Park. Predicting future frames using retrospective cycle GAN. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1811–1820, 2019.

[133] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *International* ACM SIGIR *Conference on Research and Development in Information Retrieval*, 2018.

[134] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. Time-series extreme event forecasting with neural networks at Uber. In *International Conference on Machine Learning (ICML)*, 2017.

[135] William Large and Stephen Yeager. Diurnal to decadal global forcing for ocean and sea-ice models: The data sets and flux climatologies, 05 2004.

[136] Vincent Le Guen and Nicolas Thome. Prévision de l'irradiance solaire par réseaux de neurones profonds à l'aide de caméras au sol. In *GRETSI*, 2019.

[137] Vincent Le Guen and Nicolas Thome. Shape and time distortion loss for training deep time series forecasting models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4191–4203, 2019.

[138] Vincent Le Guen and Nicolas Thome. A deep physical model for solar irradiance forecasting with fisheye images. In *CVPR OmniCV workshop*, 2020.

[139] Vincent Le Guen and Nicolas Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 11474–11484, 2020.

[140] Vincent Le Guen and Nicolas Thome. Probabilistic time series forecasting with structured shape and temporal diversity. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[141] Vincent Le Guen and Nicolas Thome. Deep time series forecasting with shape and temporal criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[142] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[143] KY Lee, YT Cha, and JH Park. Short-term load forecasting using an artificial neural network. *IEEE transactions on power systems*, 7(1):124–132, 1992.

[144] Sangmin Lee, Hak Gu Kim, Dae Hwi Choi, Hyung-Il Kim, and Yong Man Ro. Video prediction recalling long-term motion context via memory alignment learning. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3054–3063, 2021.

[145] Edouard Leurent, Denis Efimov, and Odalric-Ambrym Maillard. Robust estimation, prediction and control with linear dynamics and generic costs. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[146] Shihua Li, Jun Yang, Wen-Hua Chen, and Xisong Chen. *Disturbance observer-based control: methods and applications*. CRC press, 2014.

[147] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *International Conference on Learning Representations (ICLR)*, 2018.

[148] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Flow-grounded spatial-temporal video prediction from still images. In *European Conference on Computer Vision (ECCV)*, pages 600–615, 2018.

[149] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.

[150] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *International Conference on Learning Representations (ICLR)*, 2021.

[151] Liang Liu, Jiangning Zhang, Ruifei He, Yong Liu, Yabiao Wang, Ying Tai, Donghao Luo, Chengjie Wang, Jilin Li, and Feiyue Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 6489–6498, 2020.

[152] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *International Conference on Computer Vision (ICCV)*, pages 4463–4471, 2017.

[153] Yun Long, Xueyuan She, and Saibal Mukhopadhyay. Hybridnet: integrating model-based and data-driven learning to predict evolution of dynamical systems. *Conference on Robot Learning (CoRL)*, 2018.

[154] Zichao Long, Yiping Lu, and Bin Dong. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, page 108925, 2019.

[155] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-Net: Learning PDEs from data. In *International Conference on Machine Learning (ICML)*, pages 3214–3222, 2018.

[156] Chaochao Lu, Michael Hirsch, and Bernhard Scholkopf. Flexible spatio-temporal networks for video prediction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 6523–6531, 2017.

[157] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[158] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning (ICML)*, pages 3282–3291, 2018.

[159] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. *International Conference on Machine Learning (ICML)*, 2018.

[160] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. Vancouver, British Columbia, 1981.

[161] Zelun Luo, Boya Peng, De-An Huang, Alexandre Alahi, and Li Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2203–2212, 2017.

[162] M Lutter, C Ritter, and Jan Peters. Deep lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations (ICLR)*, 2019.

[163] Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

[164] Ricardo Marquez and Carlos FM Coimbra. Intra-hour DNI forecasting based on cloud tracking image analysis. *Solar Energy*, 91:327–336, 2013.

[165] Stefano Massaroli, Michael Poli, Jinkyoo Park, Atsushi Yamashita, and Hajime Asma. Dissecting neural ODEs. In *Advances in Neural Information Processing Systems (NeurIPS)*. The Neural Information Processing Systems, 2020.

[166] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *International Conference on Learning Representations (ICLR)*, 2015.

[167] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[168] Viraj Mehta, Ian Char, Willie Neiswanger, Youngseog Chung, and Jeff Schneider. Neural dynamical systems. *ICLR 2020 Deep Differential Equations Workshop*, 2020.

[169] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI Conference on Artificial Intelligence*, 2018.

[170] Adel Mellit. Artificial intelligence technique for modelling and forecasting of solar radiation data: a review. *International Journal of Artificial intelligence and Soft Computing*, 1(1):52–76, 2008.

[171] Arthur Mensch and Mathieu Blondel. Differentiable dynamic programming for structured prediction and attention. *International Conference on Machine Learning (ICML)*, 2018.

[172] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[173] Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin Murphy, and Honglak Lee. Unsupervised learning of object structure and dynamics from videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[174] Arvind T Mohan, Nicholas Lubbers, Daniel Livescu, and Michael Chertkov. Embedding hard physical constraints in neural network coarse-graining of 3d turbulence. *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

[175] Michael C Mozer. A focused back-propagation algorithm for temporal pattern recognition. *Complex systems*, 3(4):349–381, 1989.

[176] Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li F Fei-Fei, Josh Tenenbaum, and Daniel L Yamins. Flexible neural representation for physics prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8799–8810, 2018.

[177] Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

[178] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[179] Yuhao Nie, Yuchi Sun, Yuanlei Chen, Rachel Orsini, and Adam Brandt. Pv power output prediction from sky images using convolutional neural network: The comparison of sky-condition-specific sub-models and an end-to-end model. *Journal of Renewable and Sustainable Energy*, 12(4):046101, 2020.

[180] Paul L. Nunez. The brain wave equation: a model for the EEG. *Mathematical Biosciences*, 21(3):279 – 297, 1974.

[181] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in Atari games. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2863–2871, 2015.

[182] Marc Oliu, Javier Selva, and Sergio Escalera. Folded recurrent neural networks for future video prediction. In *European Conference on Computer Vision (ECCV)*, pages 716–731, 2018.

[183] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *International Conference on Learning Representations (ICLR)*, 2020.

[184] Quentin Paletta, Guillaume Arbod, and Joan Lasenby. Benchmarking of deep learning irradiance forecasting models from sky images–an in-depth analysis. *Solar Energy*, 2021.

[185] Quentin Paletta, Anthony Hu, Guillaume Arbod, Philippe Blanc, and Joan Lasenby. Spin: Simplifying polar invariance for neural networks application to vision-based irradiance forecasting. *arXiv preprint arXiv:2111.14507*, 2021.

[186] Quentin Paletta and Joan Lasenby. A temporally consistent image-based sun tracking algorithm for solar energy forecasting applications. *arXiv preprint arXiv:2012.01059*, 2020.

[187] Rasmus Palm, Ulrich Paquet, and Ole Winther. Recurrent relational networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3368–3378, 2018.

[188] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning (ICML)*, pages 1310–1318. PMLR, 2013.

[189] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *CVPR workshop*, 2017.

[190] Viorica Patraucean, Ankur Handa, and Roberto Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR 2016 Workshop Track*, 2015.

[191] Patrick Pérez, Carine Hue, Jaco Vermaak, and Michel Gangnet. Color-based probabilistic tracking. In *European Conference on Computer Vision (ECCV)*, pages 661–675. Springer, 2002.

[192] Patrick Perez, Jaco Vermaak, and Andrew Blake. Data fusion for visual tracking with particles. *Proceedings of the IEEE*, 92(3):495–513, 2004.

[193] Pascal Pernot and Fabien Cailliez. A critical review of statistical calibration/prediction models handling data inconsistency and model inadequacy. *AIChE Journal*, 63(10):4642–4665, 2017.

[194] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

[195] Dinesh Pothineni, Martin R Oswald, Jan Poland, and Marc Pollefeys. Kloudnet: Deep learning for sky image analysis and irradiance forecasting. In *German Conference on Pattern Recognition*, pages 535–551. Springer, 2018.

[196] Dimitris C Psichogios and Lyle H Ungar. A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, 38(10):1499–1511, 1992.

[197] Tong Qin, Kailiang Wu, and Dongbin Xiu. Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics*, 2019.

[198] Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison W Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

[199] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[200] Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

[201] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566*, 2017.

[202] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[203] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf. Multi-variate probabilistic time series forecasting via conditioned normalizing flows. *International Conference on Learning Representations (ICLR)*, 2021.

[204] Fitsum A Reda, Guilin Liu, Kevin J Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. SDC-Net: Video prediction using spatially-displaced convolution. In *European Conference on Computer Vision (ECCV)*, pages 718–733, 2018.

[205] Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and & Prabhat. Deep learning and process understanding for data-driven Earth system science. *Nature*, 566:195–204, 2019.

[206] R Rico-Martinez, JS Anderson, and IG Kevrekidis. 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*, pages 596–605. IEEE, 1994.

[207] Christelle Rigollier, Olivier Bauer, and Lucien Wald. On the clear sky model of the ESRA (European Solar Radiation Atlas) with respect to the heliosat method. *Solar energy*, 2000.

[208] François Rivest and Richard Kohar. A new timing error cost function for binary time series prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[209] Thomas Robert, Nicolas Thome, and Matthieu Cord. Hybridnet: Classification and reconstruction cooperation for semi-supervised learning. In *European Conference on Computer Vision (ECCV)*, pages 153–169, 2018.

[210] David Rolnick, Priya L Donti, Lynn H Kaack, Kelly Kochanski, Alexandre Lacoste, Kris Sankaran, Andrew Slavin Ross, Nikola Milojevic-Dupont, Natasha Jaques, Anna Waldman-Brown, et al. Tackling climate change with machine learning. In *NeurIPS 2019 workshop on Climate Change with Machine Learning*, 2019.

[211] Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[212] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[213] Yulia Rubanova, Tian Qi Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[214] Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

[215] Lars Ruthotto and Eldad Haber. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 62(3):352–364, 2020.

[216] Priyabrata Saha, Saurabh Dash, and Saibal Mukhopadhyay. PhICNet: Physics-incorporated convolutional recurrent neural networks for modeling dynamical systems. *arXiv preprint arXiv:2004.06243*, 2020.

[217] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Readings in speech recognition*, 159:224, 1990.

[218] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[219] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

[220] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning (ICML)*, pages 4467–4476, 2018.

[221] Hayden Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

[222] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[223] Thomas Schmidt, John Kalisch, Elke Lorenz, and Detlev Heinemann. Evaluating the spatio-temporal performance of sky-imager-based solar irradiance analysis and forecasts. *Atmospheric chemistry and physics*, 16(5):3399–3412, 2016.

[224] MG Schultz, Clara Betancourt, Bing Gong, Felix Kleinert, Michael Langguth, LH Leufen, Amirpasha Mozaffari, and Scarlet Stadtler. Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A*, 379(2194):20200097, 2021.

[225] Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, 8(1):1–8, 2017.

[226] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[227] Sungyong Seo and Yan Liu. Differentiable physics-informed graph networks. *ICLR 2020 workshop*, 2019.

[228] Sungyong Seo, Chuizheng Meng, and Yan Liu. Physics-aware difference graph networks for sparsely-observed dynamics. *International Conference on Learning Representations (ICLR)*, 2020.

[229] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 802–810, 2015.

[230] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[231] Justin Sirignano and Konstantinos Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. *Journal of computational physics*, 375:1339–1364, 2018.

[232] J. C. Slater. Damped electron waves in crystals. *Phys. Rev.*, 51, 1937.

[233] Slawek Smyl and N Grace Hua. Machine learning methods for gefcom2017 probabilistic load forecasting. *International Journal of Forecasting*, 2019.

[234] Corey Snyder and Minh Do. Streets: A novel camera network dataset for traffic flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[235] Robin Spiess, Felix Berkenkamp, Andreas Krause, and Jan Poland. Learning to compensate photovoltaic power fluctuations from images of the sky by imitating an optimal policy. In *2019 18th European Control Conference (ECC)*, pages 9–15. IEEE, 2019.

[236] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using LSTMs. In *International Conference on Machine Learning (ICML)*, pages 843–852, 2015.

[237] Austin Stone, Daniel Maurer, Alper Ayvaci, Anelia Angelova, and Rico Jonschkowski. Smurf: Self-teaching multi-frame unsupervised raft with full-image warping. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3887–3896, 2021.

[238] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Computer Vision and Pattern Recognition (CVPR)*, pages 8934–8943, 2018.

[239] Yuchi Sun, Vignesh Venugopal, and Adam R Brandt. Short-term solar power forecast with deep learning: Exploring optimal input and output configuration. *Solar Energy*, 188:730–741, 2019.

[240] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3104–3112, 2014.

[241] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[242] Zaiyong Tang and Paul A Fishwick. Feedforward neural nets as models for time series forecasting. *ORSA journal on computing*, 5(4):374–385, 1993.

[243] Zachary Teed and Jia Deng. RAFT: recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020.

[244] Luca Anthony Thiede and Pratik Prabhanjan Brahma. Analyzing the variety loss in the context of probabilistic trajectory prediction. In *International Conference on Computer Vision (ICCV)*, 2019.

[245] Michael L Thompson and Mark A Kramer. Modeling chemical processes using prior knowledge and neural networks. *AIChE Journal*, 40(8):1328–1340, 1994.

[246] Peter Toth, Danilo Jimenez Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins. Hamiltonian generative networks. *International Conference on Learning Representations (ICLR)*, 2020.

[247] Jean-François Toubeau, Jérémie Bottieau, François Vallée, and Zacharie De Grève. Deep learning-based multivariate probabilistic forecasting for short-term scheduling in power markets. *IEEE Transactions on Power Systems*, 34(2):1203–1215, 2018.

[248] Charles Truong, Laurent Oudre, and Nicolas Vayatis. A review of change point detection methods. *arXiv preprint arXiv:1801.00718*, 2018.

[249] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Supervised kernel change point detection with partial annotations. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2019.

[250] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1526–1535, 2018.

[251] Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. *International Conference on Learning Representations (ICLR)*, 2020.

[252] Loïc Vallance, Bruno Charbonnier, Nicolas Paul, Stéphanie Dubost, and Philippe Blanc. Towards a standardized procedure to assess solar forecast accuracy: A new ramp and time alignment metric. *Solar Energy*, 2017.

[253] Aäron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[254] Sjoerd van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. In *International Conference on Learning Representations (ICLR)*, 2018.

[255] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[256] Titouan Vayer, Laetitia Chapel, Nicolas Courty, Rémi Flamary, Yann Soullard, and Romain Tavenard. Time series alignment with global invariances. *arXiv preprint arXiv:2002.03848*, 2020.

[257] Hadrien Verbois, Philippe Blanc, Robert Huva, Yves-Marie Saint-Drenan, Andrivo Rusydi, and Alexandre Thiery. Beyond quadratic error: Case-study of a multiple criteria approach to the performance assessment of numerical forecasts of solar irradiance in the tropics. *Renewable and Sustainable Energy Reviews*, 117:109471, 2020.

[258] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *International Conference on Learning Representations (ICLR)*, 2017.

[259] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to generate long-term future via hierarchical prediction. In *International Conference on Machine Learning (ICML)*, pages 3560–3569, 2017.

[260] V. Volpert and S. Petrovskii. Reaction–diffusion waves in biology. *Physics of Life Reviews*, 6(4):267 – 310, 2009.

[261] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems (NeurIPS)*, pages 613–621, 2016.

[262] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *International Conference on Computer Vision (ICCV)*, pages 3332–3341, 2017.

[263] Dilin Wang and Qiang Liu. Nonlinear Stein variational gradient descent for learning diversified mixture models. In *International Conference on Machine Learning (ICML)*, 2019.

[264] Qi Wang, Feng Li, Yi Tang, and Yan Xu. Integrating model-driven and data-driven methods for power system frequency stability assessment and control. *IEEE Transactions on Power Systems*, 34(6):4557–4568, 2019.

[265] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021.

[266] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S. Yu. PredRNN++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In *International Conference on Machine Learning (ICML)*, 2018.

[267] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3D LSTM: A model for video prediction and beyond. In *International Conference on Learning Representations (ICLR)*, 2019.

[268] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. PredRNN: Recurrent neural networks for predictive learning using spatiotemporal LSTMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 879–888, 2017.

[269] Yunbo Wang, Jianjin Zhang, Hongyu Zhu, Mingsheng Long, Jianmin Wang, and Philip S Yu. Memory in memory: A predictive neural network for learning higher-order non-stationarity from spatiotemporal dynamics. In *Computer Vision and Pattern Recognition (CVPR)*, pages 9154–9162, 2019.

[270] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[271] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2746–2754, 2015.

[272] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4539–4547, 2017.

[273] E Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

[274] Ruofeng Wen and Kari Torkkola. Deep generative quantile-copula models for probabilistic forecasting. *ICML Time Series Workshop*, 2019.

[275] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *NeurIPS Time Series Workshop*, 2017.

[276] Björn Wolff, Elke Lorenz, and Oliver Kramer. Statistical learning for short-term photovoltaic power predictions. In *Computational sustainability*, pages 31–45. Springer, 2016.

[277] Haixu Wu, Zhiyu Yao, Jianmin Wang, and Mingsheng Long. Motionrnn: A flexible model for video prediction with spacetime-varying motions. In *Computer Vision and Pattern Recognition (CVPR)*, pages 15435–15444, 2021.

[278] Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 153–164, 2017.

[279] Sifan Wu, Xi Xiao, Qianggang Ding, Peilin Zhao, Ying Wei, and Junzhou Huang. Adversarial sparse transformer for time series forecasting. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[280] Yuxin Wu and Kaiming He. Group normalization. In *European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[281] Shi Xingjian, Zhourong Chen, and Hao et al Wang. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[282] Jingwei Xu, Bingbing Ni, Zefan Li, Shuo Cheng, and Xiaokang Yang. Structure preserving video prediction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1460–1469, 2018.

[283] Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2016.

[284] Dazhi Yang, Stefano Alessandrini, Javier Antonanzas, Fernando Antonanzas-Torres, Viorel Badescu, Hans Georg Beyer, Robert Blaga, John Boland, Jamie M Bright, Carlos FM Coimbra, et al. Verification of deterministic solar forecasts. *Solar Energy*, 210:20–37, 2020.

[285] Yufei Ye, Maneesh Singh, Abhinav Gupta, and Shubham Tulsiani. Compositional video prediction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 10353–10362, 2019.

[286] Cagatay Yildiz, Markus Heinonen, and Harri Lahdesmaki. Ode2vae: Deep generative second order odes with bayesian neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:13412–13421, 2019.

[287] Yuan Yin, Vincent Le Guen, Jérémie Dona, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. *Journal of Statistical Mechanics: Theory and Experiments (JSTAT)*, 2021(12):124012, dec 2021.

[288] Yuan Yin, Vincent Le Guen, Jérémie Dona, Ibrahim Ayed, Emmanuel de Bézenac, Nicolas Thome, and Patrick Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. In *International Conference on Learning Representations (ICLR)*, 2021.

[289] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[290] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[291] Rose Yu, Stephan Zheng, and Yan Liu. Learning chaotic dynamics using tensor recurrent neural networks. In *ICML Workshop on Deep Structured Prediction*, volume 17, 2017.

[292] Ye Yuan and Kris Kitani. Diverse trajectory forecasting with determinantal point processes. *International Conference on Learning Representations (ICLR)*, 2020.

[293] Ye Yuan and Kris Kitani. Dlow: Diversifying latent flows for diverse human motion prediction. In *European Conference on Computer Vision (ECCV)*, 2020.

[294] Huaguang Zhang, Zhanshan Wang, and Derong Liu. A comprehensive review of stability analysis of continuous-time recurrent neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 25(7):1229–1262, 2014.

[295] Jinsong Zhang, Rodrigo Verschae, Shohei Nobuhara, and Jean-François Lalonde. Deep photovoltaic nowcasting. *Solar Energy*, 176:267–276, 2018.

[296] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[297] Tianjun Zhang, Zhewei Yao, Amir Gholami, Joseph E Gonzalez, Kurt Keutzer, Michael W Mahoney, and George Biros. Anodev2: A coupled neural ode framework. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:5151–5161, 2019.

[298] Zheng Zhang, Romain Tavenard, Adeline Bailly, Xiaotong Tang, Ping Tang, and Thomas Corpetti. Dynamic time warping under limited warping path length. *Information Sciences*, 2017.

[299] Zhao Zhen, Xuemin Zhang, Shengwei Mei, Xiqiang Chang, Hua Chai, Rui Yin, and Fei Wang. Ultra-short-term irradiance forecasting model based on ground-based cloud image and deep learning algorithm. *IET Renewable Power Generation*, 2021.

[300] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *AAAI Conference on Artificial Intelligence*, 2021.

[301] Lingxue Zhu and Nikolay Laptev. Deep and confident prediction for time series at Uber. In *International Conference on Data Mining Workshops (ICDMW)*, pages 103–110. IEEE, 2017.

[302] Mai Zhu, Bo Chang, and Chong Fu. Convolutional neural networks combined with Runge-Kutta methods. In *International Conference on Learning Representations (ICLR)*, 2019.

Résumé de la thèse

10.3 Introduction

Cette thèse aborde le problème de la prédiction spatio-temporelle par apprentissage profond. Cela correspond à la tâche de prédiction de phénomènes complexes sous forme de séries temporelles ou de vidéos, ce qui nécessite de modéliser des dépendances temporelles complexes avec d'importantes corrélations spatiales. Ce sujet est d'une importance cruciale pour de nombreuses applications, telle que la prévision climatique, le diagnostic médical, l'évolution des marchés financiers, la demande pour des produits en commerce ou la maintenance prédictive dans l'industrie. A Électricité de France (EDF), l'application qui motive cette thèse est la prévision à court-terme de la production photovoltaïque à l'aide d'images fisheye. Cette tâche est habituellement résolue à l'aide d'algorithmes basés sur les prévisions météo et les images satellite. Toutefois ces sources de données ont une résolution spatiale et temporelle insuffisante pour prédire l'irradiance solaire à très court-terme (< 20min) à l'échelle d'un parc de production photovoltaïque particulier.

Dans cette thèse, nous abordons ces tâches de prédiction avec des méthodes d'intelligence artificielle, en particulier l'apprentissage statistique et l'apprentissage profond. Ces dernières années, l'apprentissage profond a connu un rebond de popularité impressionnant avec le succès du réseau de neurones profond AlexNet [127] qui a surpassé toutes les méthodes d'apprentissage machine traditionnel lors de la compétition de classification d'images ImageNet. Depuis, l'apprentissage profond s'est imposé comme le paradigme état de l'art pour de nombreuses tâches liées à la perception, telle que la vision par ordinateur, la reconnaissance vocale ou le traitement du langage naturel. Malgré ces impressionnants succès, les méthodes d'apprentissage entièrement basées sur les données sont limitées pour extrapoler l'évolution de systèmes physiques complexes, particulièrement quand la volumétrie de données est faible et pour des séries temporelles non-stationnaires avec des possibles variations brusques. La tâche d'extrapolation sous-jacente est par nature très différente des tâches de perception pour lesquelles l'apprentissage profond est très efficace, et nécessite de modéliser des dynamiques complexes.

Pour pallier à ces problèmes, nous proposons dans cette thèse d'exploiter de l'information physique a priori en combinaison avec les méthodes d'apprentissage basées données. Il s'agit d'une question très étudiée dans la littérature mais qui reste toujours largement ouverte. Les différents contextes de prévision sont illustrés sur la Figure 10.2. D'un côté les méthodes basées modèle (*model-based, MB)* supposent une bonne compréhension mathématique ou physique des phénomènes, souvent formalisée sous forme d'équations différentielles ordinaires ou partielles. A partir de données pour les conditions initiales et aux limites, la prédiction est effectuée par la résolution numérique des équations. C'est le paradigme dominant dans de nombreux domaines scientifiques, par exemple la mécanique des fluides computationnelle. Toutefois ces méthodes sont limitées si la connaissance physique est imparfaite, ce

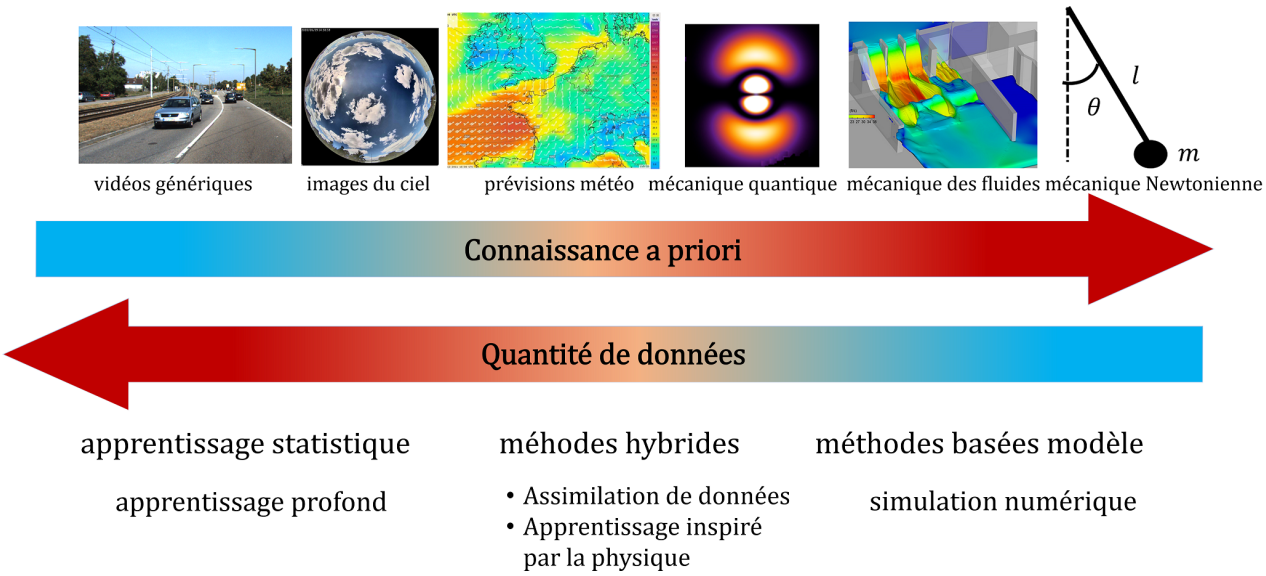


Figure 10.2: **Les différents contextes de prédiction**. A gauche, l'apprentissage statistique et profond peuvent extrapoler des systèmes dynamiques sans a priori après apprentissage sur un grand jeu de données. A droite, les méthodes basées modèle supposent une connaissance physique complète du système et prédisent le futur par simulation numérique depuis des conditions aux limites. Entre les deux, les méthodes hybrides utilisant des données et de la connaissance incomplète sont une voie d'exploration très active et prometteuse.

qui est souvent le cas pour des systèmes physiques complexes comme la modélisation du climat.

De l'autre côté, les méthodes d'apprentissage machine (*Machine Learning, ML*) sont une alternative agnostique à l'information a priori sur le système. L'apprentissage profond a prouvé sa capacité à apprendre automatiquement des relations complexes à partir de grandes bases de données annotées et est devenu état de l'art pour de nombreuses tâches de prédiction. Toutefois, ces méthodes sont toujours limitées pour modéliser des dynamiques physiques complexes. En outre, elles manquent la plausibilité physique pour interpréter les résultats et extrapoler pour de nouvelles conditions.

Entre les deux, les méthodes hybrides *model-based machine learning* (MB/ML) sont une approche attrayante pour combiner de l'information a priori et des données. Historiquement, les méthodes d'assimilation de données exploitent des données pour corriger les prédictions de modèles physiques en présence d'observations bruitées [18, 116]. Elles constituent toujours l'état de l'art pour la prévision météorologique.

Revisiter la coopération MB/ML avec l'apprentissage profond moderne est un sujet émergent qui suscite un intérêt majeur pour de nombreuses communautés scientifiques. La physique peut être incorporée dans l'apprentissage de modèles soit sous la forme de contraintes douces dans la fonction de perte [201, 231], soit comme des contraintes dures dans les architectures des réseaux [57, 174]. Du point de vue apprentissage, ces contraintes physiques permettent de développer des modèles plus interprétables qui se conforment aux lois physiques et qui restent robustes en présence de données bruitées. Cela se traduit typiquement par une plus grande efficacité dans l'utilisation des données et de meilleures performances d'extrapolation au-delà du domaine d'apprentissage.

Dans cett thèse, nous explorons cette catégorie de méthodes hybrides et nos contributions tâchent de répondre à la question générale suivante:

Comment exploiter de la connaissance physique a priori dans des modèles d'apprentissage statistique?

Nous nous concentrons sur deux principales directions: incorporation d'information physique a priori dans la fonction d'entraînement des modèles et développement d'architectures augmentées MB/ML dans le cas de connaissance physique incomplète.

10.4 Critères différentiable de forme et de temps pour la prédiction déterministe et probabiliste

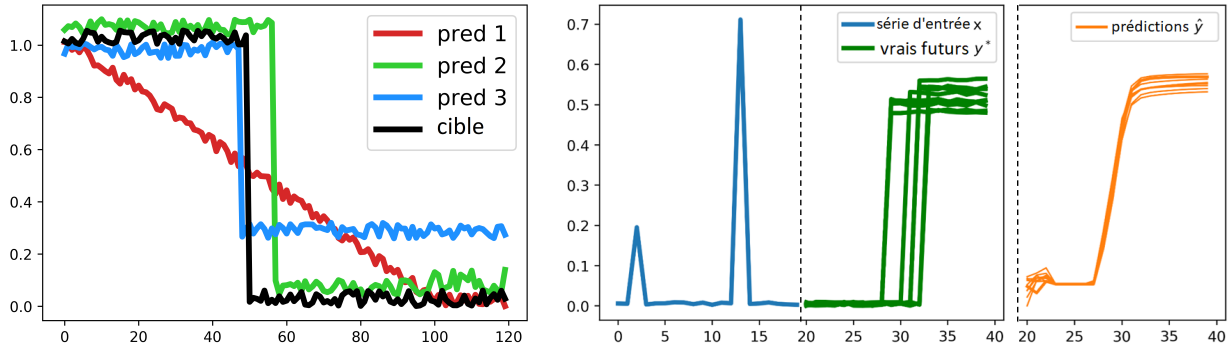
Les méthodes traditionnelles de prévision de séries temporelles sont des méthodes statistiques basées modèle qui décrivent des caractéristiques telles que les tendances et la saisonalité. Elles comprennent les méthodes autorégressives comme les modèles ARIMA (*Auto Regressive Integrated Moving Average*) [21]. Ces méthodes font souvent des hypothèses fortes sur les données, par exemple la stationarité, qui ne sont pas vérifiées en pratique.

Avec l'avènement de l'apprentissage profond, les réseaux de neurones profonds sont devenus la méthode état de l'art pour la prédiction de séries temporelles [133, 219, 183, 300], grâce à leur capacité à modéliser des dépendences temporelles complexes à partir d'un corpus d'apprentissage. La plupart des travaux récents se sont concentrés sur l'amélioration des architectures des réseaux de neurones. Le choix de la fonction de perte d'apprentissage, tout aussi important, est quant à lui peu abordé: la plupart des méthodes optimisent l'erreur quadratique moyenne (EQM) ou ses variantes.

L'erreur quadratique moyenne (EQM) est assez peu adaptée pour comparer des séries temporelles à plusieurs pas de temps, comme nous l'illustrons sur la Figure 10.3. L'EQM ne permet pas de modéliser les erreurs de forme ni les décalages temporels entre séries. Pourtant, des critères de forme et de temps sont utilisés dans les applications pour évaluer les prédictions fournies par des algorithmes, par exemple le ramp score [252] pour la forme et le TDI (Temporal Distortion Index) [83] pour le temps. Mais ils ne sont pas utilisés en pratique pour l'entraînement des réseaux de neurones car ils sont la plupart du temps non différentiables.

Dans cette thèse, nous proposons d'exploiter des critères de forme et de temps pour l'entraînement de réseaux de neurones profonds pour la prédiction de séries temporelles, dans le cas déterministe et probabiliste. Notre objectif est d'aborder des problèmes de prédiction non stationnaires, où les séries temporelles peuvent avoir des variations brutales, comme c'est le cas pour l'irradiance solaire qui chute brutalement lorsqu'un nuage occulte le soleil. Pour cela, nous introduisons des critères différentiables de forme et de temps, que nous formulons à la fois sous la forme de dissimilarités (fonctions de perte) et de similarités (noyaux semi-définis positifs). Les critères de forme sont basés sur une approximation différentiable de l'algorithme du *Dynamic Time Warping (DTW)* [217] et ceux de temps sur le *Temporal Distortion Index* (TDI) [83].

Nous proposons deux implémentations de ces critères, pour la prévision déterministe et probabiliste.



(a) Prévision déterministe **(b) Prévision probabiliste**

Figure 10.3: **Limites d l'erreur quadratique moyenne pour la prévision déterministe et probabiliste.** (a) Pour la prévision déterministe, les trois prédictions (1,2,3) ont la même erreur quadratique moyenne (EQM) par rapport au vrai futur (en noir). Mais on voudrait favoriser la prédiction 2 (bonne forme, léger retard) et 3 (bon positionnement temporel, forme imprécise) sur la prédiction 1 (pas très informative). (b) Pour la prévision probabiliste, les méthodes état de l'art apprises avec l'EQM [292, 203] perdent la capacité à produire des prédictions nettes (en orange) par rapport aux vraies trajectoires futures (en vert).

10.4.1 DILATE

Pour la prévision déterministe de séries temporelles avec des réseaux de neurones profonds, nous introduisons une fonction de perte appelée DILATE (*DIstortion Loss with shApe and TimE*). Conçue comme une alternative à l'EQM, DILATE combine une composante sur la forme des séries temporelles et une composante sur le décalage temporel pour comparer une série prédite $\hat{\mathbf{y}}$ avec le vrai futur $\mathbf{y}^*$:

$$\mathcal{L}_{\text{DILATE}}(\hat{\mathbf{y}}, \mathbf{y}^*) = \alpha\ \mathcal{L}_{forme}(\hat{\mathbf{y}}, \mathbf{y}^*) + (1-\alpha)\ \mathcal{L}_{temporelle}(\hat{\mathbf{y}}, \mathbf{y}^*) \tag{10.3}$$

$$= \alpha\ \text{DTW}_{\gamma}^{\mathbf{\Delta}}(\hat{\mathbf{y}}, \mathbf{y}^*) + (1-\alpha)\ \text{TDI}_{\gamma}^{\mathbf{\Delta},\mathbf{\Omega}_{\mathbf{dissim}}}(\hat{\mathbf{y}}, \mathbf{y}^*). \tag{10.4}$$

Le principe de DILATE est illustré sur la Figure 10.4. La perte sur la forme $\mathcal{L}_{forme}$ correspond à la soft-DTW [55] et la perte temporelle $\mathcal{L}_{temporelle}$ à une relaxation différentiable du TDI [83]. Les deux pertes sont combinées linéairement avec un facteur $\alpha \in [0; 1]$ qui est un hyperparamètre de la méthode.

Nous conduisons des expériences sur plusieurs jeux de données synthétiques et réels pour évaluer les performances de la perte DILATE. Les résultats révèlent que l'entraînement avec DILATE améliore significativement les performances évaluées sur des critères de forme et de temps, tout en maintenant des performances équivalentes évaluées en EQM. DILATE est agnostique à l'architecture du réseau de neurones et fonctionne aussi bien avec des architecture standard que les dernières architectures état de l'art.

10.4.2 STRIPE

La prévision probabiliste consiste à décrire la loi de probabilité conditionnelle des trajectoires futures sachant une trajectoire d'entrée. Dans cette thèse, notre objectif est de décrire cette loi de

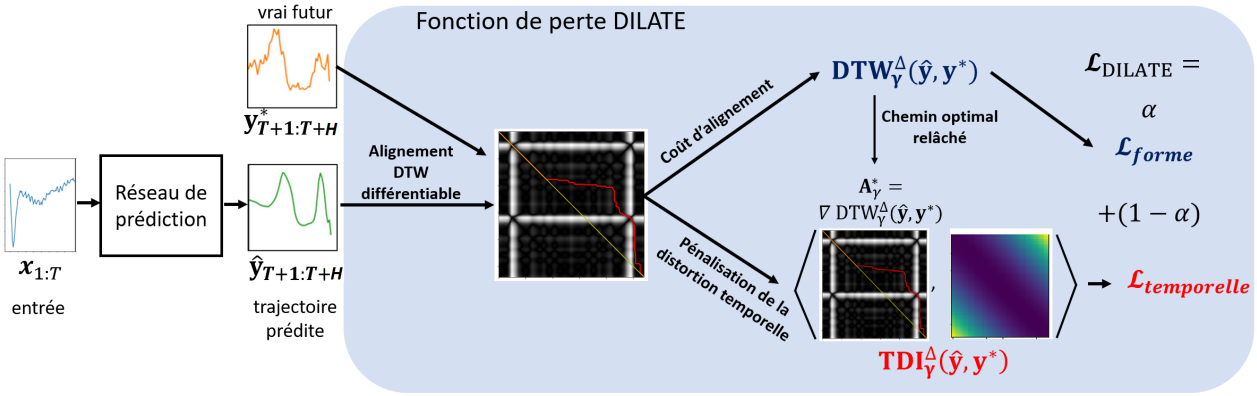


Figure 10.4: Fonction de perte DILATE pour l'entraînement de réseaux de neurones profonds pour la prédiction déterministe de séries temporelles.

probabilité par un petit ensemble (par exemple 10) de trajectoires futures possibles qui représentent bien la variabilité sur l'évolution du phénomène. Ces scénarios doivent être à la fois précis et divers selon des critères de forme et de temps, ce que ne permettent pas les méthodes actuellement état de l'art en prévision probabiliste [219, 202].

Pour cela, nous introduisons un modèle appelé STRIPE (*Shape and Time diverRsIty in Probabilistic forEcasting*). Illustré sur la Figure 10.5, le modèle STRIPE est une architecture de type encodeur-décodeur qui permet de générer des trajectoires futures à plusieurs pas de temps. Il s'agit d'un modèle génératif où les différents futurs possibles sont générés à partir de l'échantillonnage de variables latentes. Plus précisément, le modèle STRIPE est composé d'un encodeur qui prend la série temporelle d'entrée $\mathbf{x}_{1:T}$ et produit une variable descriptive h. On adjoint à cette variable h des variables latentes z_s et z_t qui capturent la variabilité en forme (respectivement en temps). Le décodeur prend en entrée la concaténation (h, z_s, z_t) et produit une trajectoire future $\hat{\mathbf{y}}_{T+1:T+\tau}$.

Pour structurer la diversité des trajectoires prédites, les variables latentes sont générées par des réseaux de neurone appelés STRIPE-forme et STRIPE-temps. La diversité est favorisée par l'ajout d'une fonction de perte de diversité $\mathcal{L}_{diversité}$. Elle est basée sur l'utilisation des processus ponctuels déterminantaux (DPP) [130], qui sont un outil mathématique élégant pour décrire la diversité d'un ensemble d'éléments. La perte de qualité $\mathcal{L}_{qualité}$ est la perte DILATE pour assurer des prédictions avec à la fois la bonne forme et un faible décalage temporel. Pour assurer le maintien de la qualité des prédictions lors de l'étape de diversification, un réseau postérieur permet d'échantillonner les variables latentes lors de l'entraînement, pour qu'elles correspondent à de réelles trajectoires du jeu de données.

Nous menons des expériences sur un jeu de données synthétique où l'on dispose de l'ensemble des futures trajectoires comme supervision, ainsi que sur des jeux de données réels où l'on a qu'un seul futur disponible. Les résultats montrent que STRIPE parvient à des prédictions avec une bien meilleure diversité mesurée avec des critères de forme et de temps que des mécanismes de diversification concurrents de la littérature [63, 244, 71, 292] et que des algorithmes dédiés à la prédiction probabiliste [219]. De plus, STRIPE maintient une bonne qualité des prédictions obtenues et obtient le meilleur compromis entre qualité et diversité.

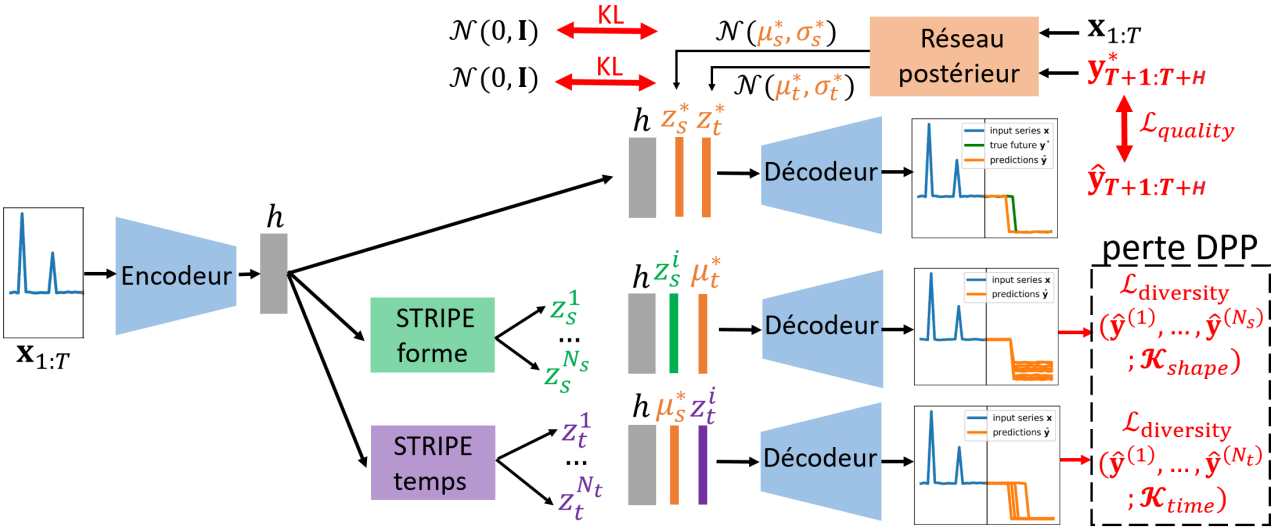


Figure 10.5: Modèle STRIPE pour la prévision probabiliste.

10.5 Prédiction avec incorporation d'information physique incomplète

Dans cette partie de la thèse, nous explorons comment incorporer de l'information physique a priori dans les modèles d'apprentissage statistique. En particulier, nous nous intéressons au cas où la connaissance physique est incomplète, ce qui est une question très peu traitée dans la littérature.

10.5.1 Modèle PhyDNet pour la prédiction de vidéo

Nous proposons un modèle d'apprentissage profond dédié à la prédiction de vidéos, dénommé PhyDNet, qui incorpore de l'information physique sous la forme d'une classe d'équations aux dérivées partielles (EDP) linéaires. Toutefois, pour des vidéos génériques, les équations physiques de la dynamique ne s'appliquent pas directement au niveau des pixels. Par exemple, il est nécessaire au préalable de segmenter les objets et de déterminer leur centre de masse avant d'appliquer les lois de Newton. C'est un cas représentatif d'un a priori non observable dans l'espace d'entrée.

Pour traiter ce problème, nous supposons qu'il existe un espace latent dans lequel le modèle dynamique d'EDP linéaire s'applique. Le modèle PhyDNet est composé d'un encodeur-décodeur pour apprendre automatiquement l'espace latent le plus adapté à partir des données. Dans cet espace latent, nous décomposons la dynamique en deux parties: une partie qui intégre les lois a priori de la physique et une partie qui apprend l'information complémentaire à la physique nécessaire pour avoir une bonne prédiction au niveau des pixels.

Le modèle PhyDNet est un réseau de neurones récurrent, illustré sur la Figure 10.6 dans sa version pliée (à gauche) et dépliée (à droite). Pour modéliser la partie physique, nous introduisons une cellule récurrente appelée PhyCell qui discrétise une équation aux dérivées partielle linéaire par un schéma d'Euler, pour laquelle les dérivées partielles sont calculées avec des convolutions contraintes [155]. La deuxième branche modélise le résidu qui n'est pas expliqué par la physique; pour cela nous utilisons un

réseau de neurones récurrent assez générique, en l'occurence un ConvLSTM [281]. Les deux branches sont sommées dans l'espace latent, avant d'être décodées vers une prédiction de l'image future.

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = \mathbf{h}_t + \Phi(\mathbf{h}_t) & \textbf{Prediction} \quad (10.5) \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \mathbf{K}_t \odot \left(\mathbf{E}(\mathbf{u}_t) - \tilde{\mathbf{h}}_{t+1}\right). & \textbf{Correction} \quad (10.6) \end{cases}$$

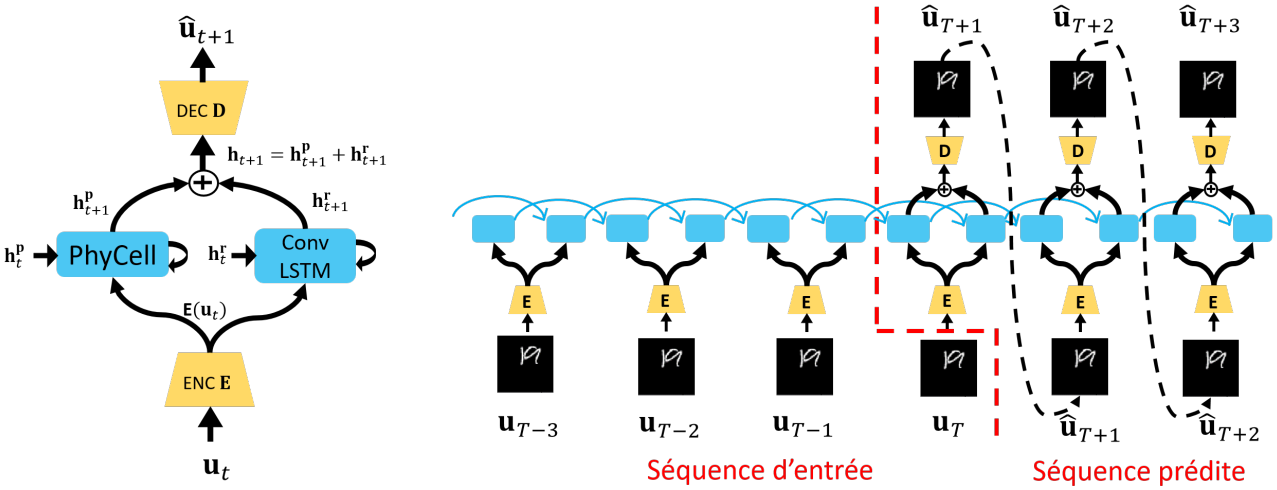


Figure 10.6: Modèle PhyDNet pour la prédiction de vidéo.

Nous menons des expériences sur des jeux de données avec différents niveaux de connaissance a priori: depuis Moving MNist où la dynamique de déplacement des chiffres est parfaitement connue, jusqu'à des vidéos généralistes de mouvements humains, en passant par des cas où l'on a un a priori physique incomplet sur la dynamique, comme pour le traffic routier ou la température de surface des océans. Dans tous ces cas, nous montrons la supériorité de PhyDNet par rapport à des modèles d'apprentissage profond sans a priori physique.

10.5.2 Modèle APHYNITY pour la coopération optimale entre physique et apprentissage profond

La prédiction de systèmes dynamiques pour lesquels on a une connaissance partielle de leur dynamique est un problème très courant dans de nombreux champs scientifiques. Par exemple pour la modélisation climatique, il est très compliqué de mettre en équations précisément tous les phénomènes complexes régissant la dynamique de l'atmosphère.

Nous introduisons ici un schéma d'apprentissage, appelé APHYNITY, pour augmenter des modèles physiques simplifiés décrits par des équations aux dérivées partielles, avec des réseaux de neurones profonds. Nous considérons des systèmes dynamiques sous la forme de l'équation différentielle:

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = F(X_t). \quad (10.7)$$

Le modèle APHYNITY décompose la fonction de dynamique F en une composante F_p pour laquelle nous avons un a priori physique et une composante d'augmentation F_a qui corrige les erreurs du modèle physique: $F = F_p + F_a$.

Le problème d'apprentissage est formulé de manière à ce que le modèle physique explique la dynamique le plus possible, tandis que le modèle d'augmentation ne capture que l'information qui ne peut pas être capturée par la physique. Inspiré par le principe de moindre action, ce schéma d'apprentissage consiste à minimiser la norme du résidu F_a sous la contrainte de prédiction parfaite du modèle augmenté:

$$\min_{F_p \in \mathcal{F}_p, F_a \in \mathcal{F}} \|F_a\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{\mathrm{d}X_t}{\mathrm{d}t} = (F_p + F_a)(X_t). \tag{10.8}$$

Sous de faibles hypothèses qui sont vérifiées dans de nombreux cas expérimentaux, il y a existence et unicité du problème d'optimisation APHYNITY, ce qui favorise l'interprétabilité et la généralisation du modèle.

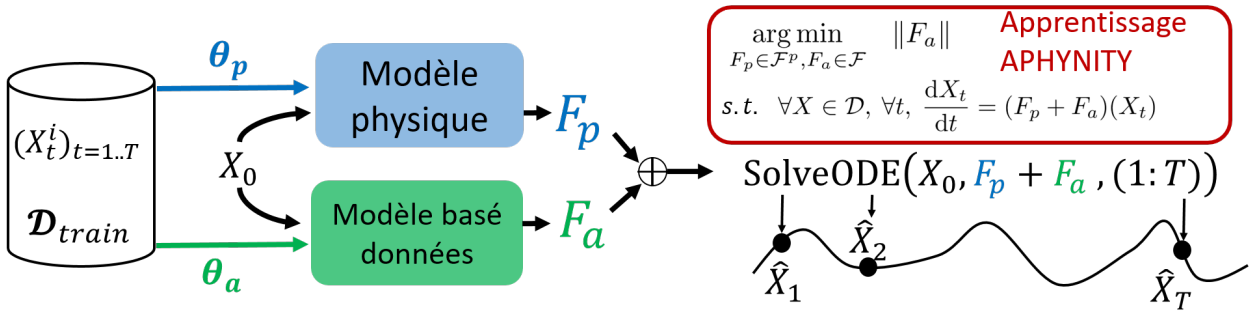


Figure 10.7: Schéma d'apprentissage APHYNITY pour la coopération optimale entre modèles physiques et modèles d'apprentissage.

Nous proposons une approche trajectoire pour implémenter en pratique le schéma APHYNITY, qui est illustré sur la Figure 10.7. A partir d'une condition initiale X_0, un modèle physique paramétré par θ_p donne la dynamique physique F_p, tandis que le modèle d'augmentation basé données paramétrisé par θ_a fournit la dynamique F_a. La dynamique résultante $F = F_p + F_a$ est intégrée dans le temps par un schéma numérique différentiable qui donne les prédictions pour un ensemble de pas de temps futurs. Les paramètres du modèle sont appris par l'optimisation du problème sous contraintes APHYNITY (Eq 10.8). Un algorithme d'optimisation sous contraintes adaptatif est utilisé pour résoudre efficacement le problème de l'Eq 10.8.

Nous menons des expériences sur trois problèmes representatifs de classes de phénomènes physiques: dynamique Newtonienne (pendule amorti), équations de réaction-diffusion et équations d'ondes. Dans chaque cas, nous considérons des modèles physiques simplifiés (par exemple les équations du pendule sans le terme d'amortissement) et augmentons ces modèles avec le schéma APHYNITY.

Les résultats expérimentaux montrent la supériorité d'APHYNITY sur des modèles basés données uniquement, sur des modèles physiques incomplets et sur des méthodes état de l'art qui combinent données et connaissances. Le gain de performances se voit à la fois sur l'erreur de prédiction et sur l'erreur d'identification des paramètres physiques du modèle. De plus, l'approche APHYNITY est

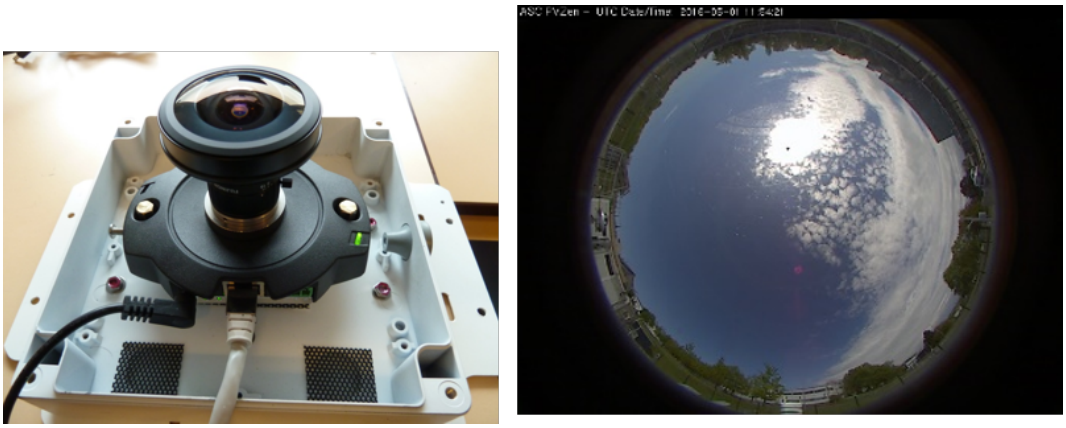


Figure 10.8: Caméra fisheye et exemple d'image fisheye utilisées pour la prévision à court-terme de l'irradiance solaire.

suffisamment flexible pour s'adapter à des niveaux différents de connaissance physique a priori.

10.6 Application à la prédiction d'irradiance solaire

Les énergies renouvelables sont en forte progression dans le monde ces dernières années. Toutefois, leur variabilité spatiale et temporelle reste un défi pour leur intégration à grande échelle dans les réseaux électriques existants, pour lesquels l'équilibre à tout instant entre production et consommation d'électricité est primordial. L'enjeu réside également dans le pilotage indépendant de parcs photovoltaïques ou éoliens qui peuvent être couplés à des moyens de stockage ou de production supplémentaires, notamment dans les systèmes insulaires isolés.

Dans ce contexte, EDF a engagé depuis plusieurs années des travaux sur la prévision de production photovoltaïque, à différents horizons temporels et à l'aide de différentes données d'entrée (modèles météorologiques, images satellites, images au sol, mesures en temps réel). L'amélioration des méthodes de prévision à court terme (de quelques minutes à une heure) est aujourd'hui un enjeu fondamental. La variabilité temporelle à court-terme de la production photovoltaïque est principalement liée à des phénomènes physiques météorologiques, tels que le déplacement des nuages. Les modèles météorologiques et les images satellite ont une résolution spatiale et temporelle insuffisante pour prédire le déplacement des nuages à court-terme au-dessus d'un site de production. Pour cela, l'utilisation de caméras au sol hémisphériques est une piste très prometteuse pour suivre les nuages et anticiper les variations brusques de production à quelques minutes [87, 43, 42, 164, 223]. EDF dispose de plusieurs sites instrumentés de caméras hémisphériques fisheye et de capteurs de rayonnement solaire (pyranomètres), constituant ainsi une base de données annotées de plusieurs millions d'images du ciel au pas de temps 10s (Figure 10.8).

Les méthodes traditionnelles de prévision par images fisheye reposent sur du traitement d'images classique. La chaîne de traitement typique [87, 43, 42, 223] se compose des étapes suivantes: calibration de la caméra fisheye, prétraitement de l'image, segmentation de l'image avec des seuillages, calcul du flot optique et propagation du mouvement pour prévoir la future position des nuages et enfin calcul de l'irradiance future avec des algorithmes de régression.

Depuis quelques années, les méthodes d'apprentissage profond se sont révélées être une alternative intéressante pour estimer et prévoir le rayonnement solaire de bout en bout [195, 295, 235, 239, 179, 186, 299], sans la nécessité de definir des indicateurs sur les images manuellement. Au début de cette thèse, nous avons exploré de premières architectures de réseaux de neurones profonds pour l'estimation et la prévision du rayonnement [136]. Pour l'estimation du rayonnement correspondant à l'image courante, nous avons remarqué un gain de performances très important en utilisant des réseaux convolutionnels par rapport aux méthodes traditionelles, ce qui était attendu sachant les succès de l'apprentissage profond pour les tâches de perception. Par contre, la prévision du rayonnement est une tâche beaucoup plus compliquée: notre architecture préliminaire basée sur un ConvLSTM donne de meilleurs résultats que la méthode traditionnelle mais avec une marge plus faible.

Pour améliorer les prédictions, nous avons appliqué les contributions méthodologiques de cette thèse à ce problème. Nous avons adapté le modèle PhyDNet de prédiction de vidéo à la prédiction jointe des images fisheye et des rayonnements futurs. Illustrée sur la Figure 10.9, cette architecture prend en entrée une séquence d'images fisheye qui est traitée par le réseau de neurones récurrent PhyDNet. Le réseau est ensuite appliqué récursivement pour décoder les images futures et les rayonnements futurs.

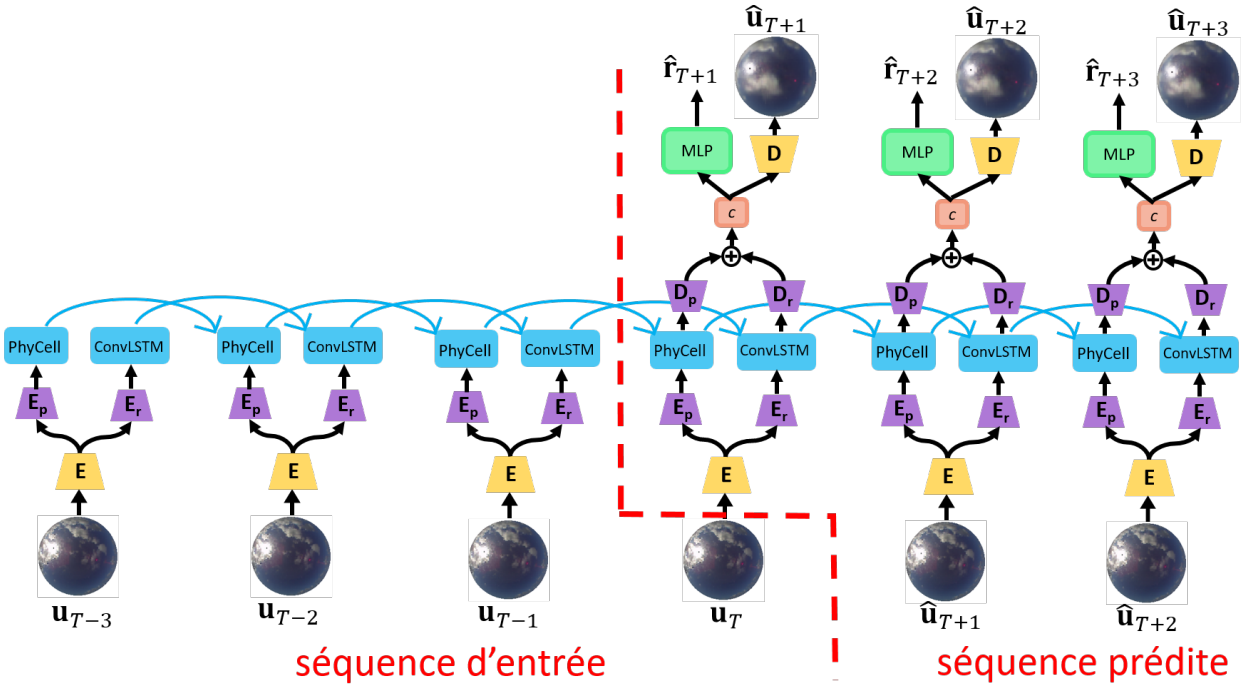


Figure 10.9: Modèle PhyDNet adapté pour la prévision de l'irradiance solaire.

Le modèle PhyDNet a permis un gain de performances important sur les prévisions de l'irradiance solaire à 5min par rapport à notre modèle de base ConvLSTM.

Nous avons également exploré l'application de la fonction de perte DILATE et du schéma d'apprentissage APHYNITY à ce problème. Ces deux mécanismes permettent d'obtenir un nouveau gain de performances, quoique plus faible que celui apporté par l'architecture inspirée par la physique PhyDNet. Nous en avons analysé les raisons et proposé des pistes d'améliorations futures.

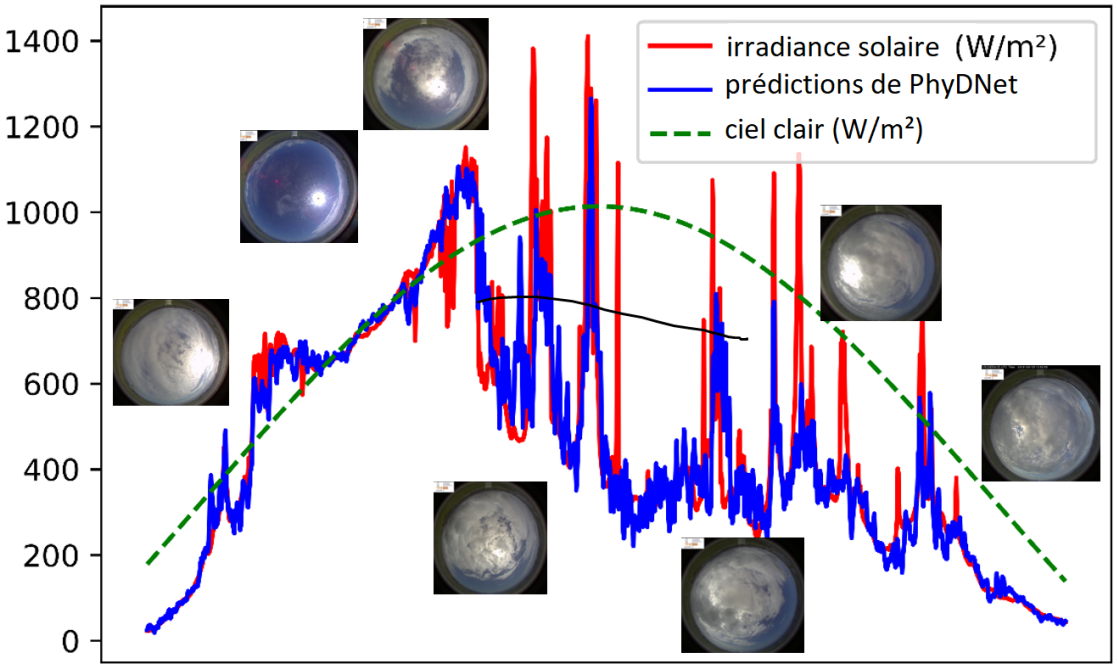


Figure 10.10: Prévisions de l'irradiance à 5min avec des images fisheye. Notre modèle inspiré par la physique prédit correctement les variations brusques de l'irradiance solaire.

10.7 Conclusion et perspectives

Dans cette thèse, nous avons exploré de manière générale comment incorporer de la connaissance physique a priori dans les modèle d'apprentissage statistique pour améliorer la prévision spatio-temporelle. Plus particulièrement, nous avons abordé deux principales directions de recherche.

La première concerne le choix de la fonction de perte pour entraîner les modèles. Au lieu de l'erreur quadratique moyenne très majoritairement utilisée, nous proposons d'utiliser des critères de forme et de décalage temporel sur les trajectoires prédites. Nous nous attaquons au contexte de la prévision déterministe avec notre proposition de fonction de perte DILATE, et au contexte probabiliste, où notre objectif est de décrire la distribution prédictive par un faible nombre de scénarios divers et précis, avec notre modèle STRIPE.

Notre seconde direction de recherche est d'augmenter des modèles physiques incomplets avec des réseaux de neurones profonds basés données. Pour la prédiction de vidéo, nous introduisons le modèle PhyDNet qui sépare une partie de dynamique physique modélisée par des équations aux dérivées partielles, d'une partie résiduelle qui capture l'information complémentaire, comme la texture et les détails, nécessaire à la bonne prédiction. Nous proposons aussi un schéma d'apprentissage, appelé APHYNITY, qui assure une décomposition bien posée et unique entre des modèle physiques incomplets et des réseaux de neurones profonds, sous de faibles hypothèses.

Nous avons validons les contributions de cette thèse sur de nombreux jeux de données synthétiques et réels, et sur l'application de prévision photovoltaïque à EDF.

Les travaux de cette thèse ouvrent de nombreuses perspectives intéressantes à explorer. A court-

terme, les perspectives pour l'améliorations des prédictions d'irradiance comprennent l'utilisation de modèle physiques plus spécifiques à la dynamique de l'atmosphère, l'apprentissage sur des séquences temporelles de plus longue durée, ou encore l'utilisation de réseaux de neurones qui encodent l'invariance par rotation pour le traitement des images fisheye.

A plus long terme, l'étude des modèles physiques augmentés et leur application pour résoudre des problèmes naturels complexes comme la prévision climatique est particulièrement attrayante. Plusieurs applications pourraient directement bénéficier de ces travaux, par exemple l'estimation du flot optique qui est traditionnellement basée sur l'hypothèse simplifiée de la conservation de l'intensité lumineuse, ou l'apprentissage par renforcement basé modèle qui suppose un modèle de dynamique (souvent simplifié) pour prendre des décisions.

Par ailleurs, nous avons étudié dans cette thèse des décompositions linéaires entre modèles physiques simplifiés et leur augmentations, ce qui est une hypothèse assez forte. D'autres schémas de décompositions peuvent être envisagés, par exemple entre des modélisations physiques à des échelles spatiales différentes.

Mots-clés : apprentissage profond, prévision spatio-temporelle, prévision photovoltaïque.

Appendix A

Appendix for differentiable shape and temporal criteria for non-stationary forecasting

A.1 Proof that the temporal kernel is PSD

The DTW score between two time series $\mathbf{y} \in \mathbb{R}^{d\times n}$ and $\mathbf{z} \in \mathbb{R}^{d\times m}$ can be written $S(\pi) = \sum_{i=1}^{|\pi|} \boldsymbol{\Delta}(\mathbf{y}_{\pi_1(i)}, \mathbf{z}_{\pi_2(i)})$ where $\pi = (\pi_1, \pi_2)$ is a valid alignment between both series. Equivalently we can write the DTW score $S(\pi) = S(\mathbf{A}) = \langle \mathbf{A}, \boldsymbol{\Delta}(\mathbf{y}, \mathbf{z})\rangle$, where $\mathbf{A} \subset \{0,1\}^{n\times m}$ is the warping path in matrix form ($\mathbf{A}_{ij} = 1$ if $\mathbf{y}_i$ is associated to $\mathbf{z}_j$ and 0 otherwise).

Let $w : \mathcal{A}_{n,m} \longrightarrow \mathbb{R}_+^*$ be a strictly positive weighting function on alignment paths and let's consider the following kernel:

$$\mathcal{K}_w(\mathbf{y}, \mathbf{z}) = \sum_{\mathbf{A}\in\mathcal{A}_{n,m}} w(\mathbf{A}) \;\; e^{-\frac{S(\mathbf{A})}{\gamma}} \tag{A.1}$$

$$= \sum_{\mathbf{A}\in\mathcal{A}_{n,m}} w(\mathbf{A}) \;\; e^{-\frac{\langle \mathbf{A}, \boldsymbol{\Delta}(\mathbf{y},\mathbf{z})\rangle}{\gamma}} \tag{A.2}$$

$$= \sum_{\pi\in\mathcal{A}_{n,m}} w(\pi) \;\; e^{-\frac{\sum_{j=1}^{|\pi|} \boldsymbol{\Delta}\left(\mathbf{y}_{\pi_1(j)}, \mathbf{z}_{\pi_2(j)}\right)}{\gamma}} \tag{A.3}$$

$$= \sum_{\pi\in\mathcal{A}_{n,m}} w(\pi) \prod_{j=1}^{|\pi|} e^{-\frac{\boldsymbol{\Delta}\left(\mathbf{y}_{\pi_1(j)}, \mathbf{z}_{\pi_2(j)}\right)}{\gamma}} \tag{A.4}$$

$$= \sum_{\pi\in\mathcal{A}_{n,m}} w(\pi) \prod_{j=1}^{|\pi|} k(\mathbf{y}_{\pi_1(j)}, \mathbf{z}_{\pi_2(j)}), \tag{A.5}$$

where we denote $k = e^{-\frac{\boldsymbol{\Delta}}{\gamma}}$. We prove the following result:

Proposition 3 *If k is a PSD kernel such that $\frac{k}{1+k}$ is also PSD, the kernel $\mathcal{K}_w$ defined in Eq. A.5 is also PSD.*

The proof is adapted from [56]. First, for any time series $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_n) \in \mathbb{R}^{d \times n}$ of length n and for any sequence $a \in \mathbb{N}^n$, we introduce the notation:

$$\mathbf{y}_a = (\underbrace{\mathbf{y}_1, \dots, \mathbf{y}_1}_{a_1 \text{ times}}, \dots, \underbrace{\mathbf{y}_n, \dots, \mathbf{y}_n}_{a_n \text{ times}}). \tag{A.6}$$

Let χ be any PSD kernel defined on $\mathbb{R}^d$ with the following condition $|\chi| < 1$, we introduce the kernel κ defined as:

$$\kappa(\mathbf{y}, \mathbf{z}) = \begin{cases} \prod_{i=1}^{|x|} \chi(\mathbf{y}_i, \mathbf{z}_j) & \text{if } |\mathbf{y}| = |\mathbf{z}| \\ 0 \quad \text{otherwise.} \end{cases} \tag{A.7}$$

Then, given a strictly positive weighting function $w(a, b) > 0$, the following kernel $\mathcal{K}_w$ defined in Eq. A.8 is PSD by construction:

$$\mathcal{K}_w(\mathbf{y}, \mathbf{z}) = \sum_{a \in \mathbb{N}^n} \sum_{b \in \mathbb{N}^m} w(a, b)\ \kappa(\mathbf{y}_a, \mathbf{z}_b). \tag{A.8}$$

where we recall that $n = |\mathbf{y}|$ and $m = |\mathbf{z}|$. We denote $\epsilon_a = (\underbrace{1, \dots, 1}_{a_1 \text{ times}}, \dots, \underbrace{p, \dots, p}_{a_p \text{ times}})$ for any $a \in \mathbb{N}^p$. We also write for any sequences u and v of common length p: $u \otimes v = ((u_1, v_1), \dots, (u_p, v_p))$. With these notations, we can rewrite $\mathcal{K}_w$ as:

$$\mathcal{K}_w(\mathbf{y}, \mathbf{z}) = \sum_{\substack{a \in \mathbb{N}^n, b \in \mathbb{N}^m \\ \|a\| = \|b\|}} w(a, b) \prod_{i=1}^{\|a\|} \chi((\mathbf{y}, \mathbf{z})_{\epsilon_a \otimes \epsilon_b(i)}). \tag{A.9}$$

Notice now for each couple (a, b) there exists a unique alignment path π and an integral vector v verifying $\pi_v = \epsilon_a \otimes \epsilon_b$. Conversely, for each couple (π, v) there exists a unique pair (a, b) verifying $\pi_v = \epsilon_a \otimes \epsilon_b$. Therefore the kernel $\mathcal{K}_w$ in Eq. A.9 can be written equivalently with a parameterization on (π, v) for w:

$$\mathcal{K}_w(\mathbf{y}, \mathbf{z}) = \sum_{\pi \in \mathcal{A}_{n,m}} \sum_{v \in \mathbb{N}^{|\pi|}} w(\pi, v) \prod_{j=1}^{|\pi|} \chi((\mathbf{y}, \mathbf{z})_{\pi_v(j)}), \tag{A.10}$$

where $\chi_{\pi(j)}$ is a shortcut for $\chi(\mathbf{y}_{\pi_1(j)}, \mathbf{z}_{\pi_2(j)})$.

Now we assume that the weighting function w depends only on π: $w(\pi, v) = w(\pi)$. Then we have:

$$\begin{aligned}\mathcal{K}_w(\mathbf{y}, \mathbf{z}) &= \sum_{\pi \in \mathcal{A}_{n,m}} w(\pi) \sum_{v \in \mathbb{N}^{|\pi|}} \prod_{j=1}^{|\pi|} \chi_{\pi(j)}^{v_j} \\ &= \sum_{\pi \in \mathcal{A}_{n,m}} w(\pi) \prod_{j=1}^{|\pi|} \left(\chi_{\pi(j)} + \chi_{\pi(j)}^2 + \chi_{\pi(j)}^3 + \dots \right) \\ &= \sum_{\pi \in \mathcal{A}_{n,m}} w(\pi) \prod_{j=1}^{|\pi|} \frac{\chi_{\pi(j)}}{1 - \chi_{\pi(j)}}.\end{aligned}$$

By setting now $\chi = \frac{k}{1+k}$ which is PSD by hypothesis and verifies $|\chi| < 1$ (recall that $k = e^{-\frac{\mathbf{\Delta}}{\gamma}}$), we get:

$$\begin{aligned}\mathcal{K}_w(\mathbf{y}, \mathbf{z}) &= \sum_{\pi \in \mathcal{A}_{n,m}} w(\pi) \prod_{j=1}^{|\pi|} k_{\pi(j)} \\ &= \sum_{\pi \in \mathcal{A}_{n,m}} w(\pi) \prod_{j=1}^{|\pi|} k(\mathbf{y}_{\pi_1(j)}, \mathbf{z}_{\pi_2(j)}),\end{aligned}$$

which corresponds exactly to the kernel $\mathcal{K}_w$ defined in Eq. A.5. This proves that $\mathcal{K}_w$ in Eq. A.5 is a well defined PSD kernel.

With the particular choice $w(\mathbf{A}) = \langle \mathbf{A}, \mathbf{\Omega_{sim}} \rangle$, we recover:

$$\begin{aligned}\mathcal{K}_w(\mathbf{y}, \mathbf{z}) &= \sum_{\mathbf{A} \in \mathcal{A}} \langle \mathbf{A}, \mathbf{\Omega_{sim}} \rangle \quad e^{-\frac{\langle \mathbf{A}, \mathbf{\Delta}(\mathbf{y}, \mathbf{z}) \rangle}{\gamma}} \\ &= Z \times \mathrm{TDI}_\gamma^{\mathbf{\Delta}, \mathbf{\Omega_{sim}}}(\mathbf{y}, \mathbf{z}) \\ &= e^{-\mathrm{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z})/\gamma} \times \mathrm{TDI}_\gamma^{\mathbf{\Delta}, \mathbf{\Omega_{sim}}}(\mathbf{y}, \mathbf{z}) \\ &= \mathcal{K}_{time}(\mathbf{y}, \mathbf{z}),\end{aligned}$$

which finally proves that $\mathcal{K}_{time}$ defined in paper Eq. 9 is a valid PSD kernel.

The particular choice $k(u, v) = \dfrac{\frac{1}{2} e^{-\|u-v\|_2^2}}{1 - \frac{1}{2} e^{-\|u-v\|_2^2}}$ fulfills Proposition 1 requirements: k is indeed PSD as the infinite limit of a sequence of PSD kernels $\sum_{i=1}^{\infty} g^i = \frac{g}{1-g} = k$, where g is a halved Gaussian PSD kernel: $g(u, v) = \frac{1}{2} e^{-\|u-v\|_2^2}$. For this choice of k, the corresponding pairwise cost matrix writes (it is the half-Gaussian cost defined in Section 3.2.1.2):

$$\mathbf{\Delta}(\mathbf{y}_i, \mathbf{z}_j) = \gamma \left[\|\mathbf{y}_i - \mathbf{z}_j\|_2^2 - \log\left(2 - e^{-\|\mathbf{y}_i - \mathbf{z}_j\|_2^2}\right) \right]. \tag{A.11}$$

Appendix B

Appendix for DILATE

B.1 External shape and temporal metrics

We detail here the two external metrics used in our experiments to evaluate the shape and temporal errors.

Ramp score: The notion of *ramping event* is a major issue for intermittent renewable energy production that needs to be anticipated for electricity grid management. For assessing the performance of trained forecasting models in presence of ramps, the Ramp Score was proposed in [252]. This score is based on a piecewise linear approximation on both input and target time series by the Swinging Door algorithm [22, 79]. The Ramp Score described in [252] is computed as the integral between the unsigned difference of derivatives of both linear approximated series. For assessing only the shape error component, we apply in our experiments the ramp score on the target and prediction series after alignment by the optimal DTW path.

Hausdorff distance: Given a set of change points $\mathcal{T}^*$ in the target signal and change points $\hat{\mathcal{T}}$ in the predicted signal, the Hausdorff distance is defined as:

$$\text{Hausdorff}(\mathcal{T}^*, \hat{\mathcal{T}}) := \max(\max_{\hat{t}\in\hat{\mathcal{T}}} \min_{t^*\in\mathcal{T}^*} |\hat{t} - t^*|, \max_{t^*\in\mathcal{T}^*} \min_{\hat{t}\in\hat{\mathcal{T}}} |\hat{t} - t^*|). \tag{B.1}$$

It corresponds to the greatest temporal distance between a change point and its prediction.

We now explain how the change points are computed for each dataset: for Synthetic, we know exactly by construction the positions of the change points in the target signals. For the predictions, we look for a single change point corresponding to the location of the predicted step function. We use the exact segmentation method by dynamic programming described in [248] with the Python toolbox `http://ctruong.perso.math.cnrs.fr/ruptures-docs/build/html/index.html#` .

For ECG5000 and Traffic datasets which present sharp peaks, this change point detection algorithm is not suited (detected change points are often located at the inflexion points of peaks and not at the exact peak location). We thus use a simple peak detection algorithm based on first order finite

differences. We tune the threshold parameter for outputting a detection and the min distance between detections parameter experimentally for each dataset.

B.2 Comparison to DILATE divergence variant

Blondel *et al.* [17] point out two limitations for using $\text{DTW}_\gamma^{\mathbf{\Delta}}$ as a loss function: first, it can take negative values and second, $\text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z})$ does not reach its minimum when $\mathbf{y} = \mathbf{z}$. To address these issues, they propose a proper divergence defined as follows [17]:

$$\text{DTW-div}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z}) = \text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{z}) - \frac{1}{2}(\text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{y}) + \text{DTW}_\gamma^{\mathbf{\Delta}}(\mathbf{z}, \mathbf{z})). \tag{B.2}$$

This divergence is non-negative and satisfies $\text{DTW-div}_\gamma^{\mathbf{\Delta}}(\mathbf{y}, \mathbf{y}) = 0$. However, it is still not a distance function since the triangle inequality is not verified (as for the true DTW).

These limitations also hold for DILATE. Consequently, we use the same normalization trick to define a proper DILATE-divergence. Forecasting results in Table B.1 show that DILATE-div is equivalent to DILATE with the Seq2Seq and N-Beats [183] models, and inferior to DILATE with the Informer model [300]. It confirms the good behaviour of the DILATE loss that does not require this renormalization.

Table B.1: Comparison between DILATE and DILATE-div on the synthetic-det dataset.

Model	MSE	DILATE
Seq2Seq DILATE	**13.1 ± 1.8**	**33.7 ± 3.1**
Seq2Seq DILATE-div	**13.6 ± 0.9**	**33.6 ± 2.1**
N-Beats [183] DILATE	**13.3 ± 0.7**	**37.9 ± 1.6**
N-Beats [183] DILATE-div	**13.8 ± 0.9**	**38.5 ± 1.4**
Informer [300] DILATE	**11.8 ± 0.7**	**30.1 ± 1.3**
Informer [300] DILATE-div	12.9 ± 0.1	31.8 ± 6.5

B.3 DILATE additional visualizations

We provide additional qualitative predictions with DILATE for the `Synthetic-det` in Figure B.1, for `ECG5000` in Figure B.2 and for `Traffic` in Figure B.3.

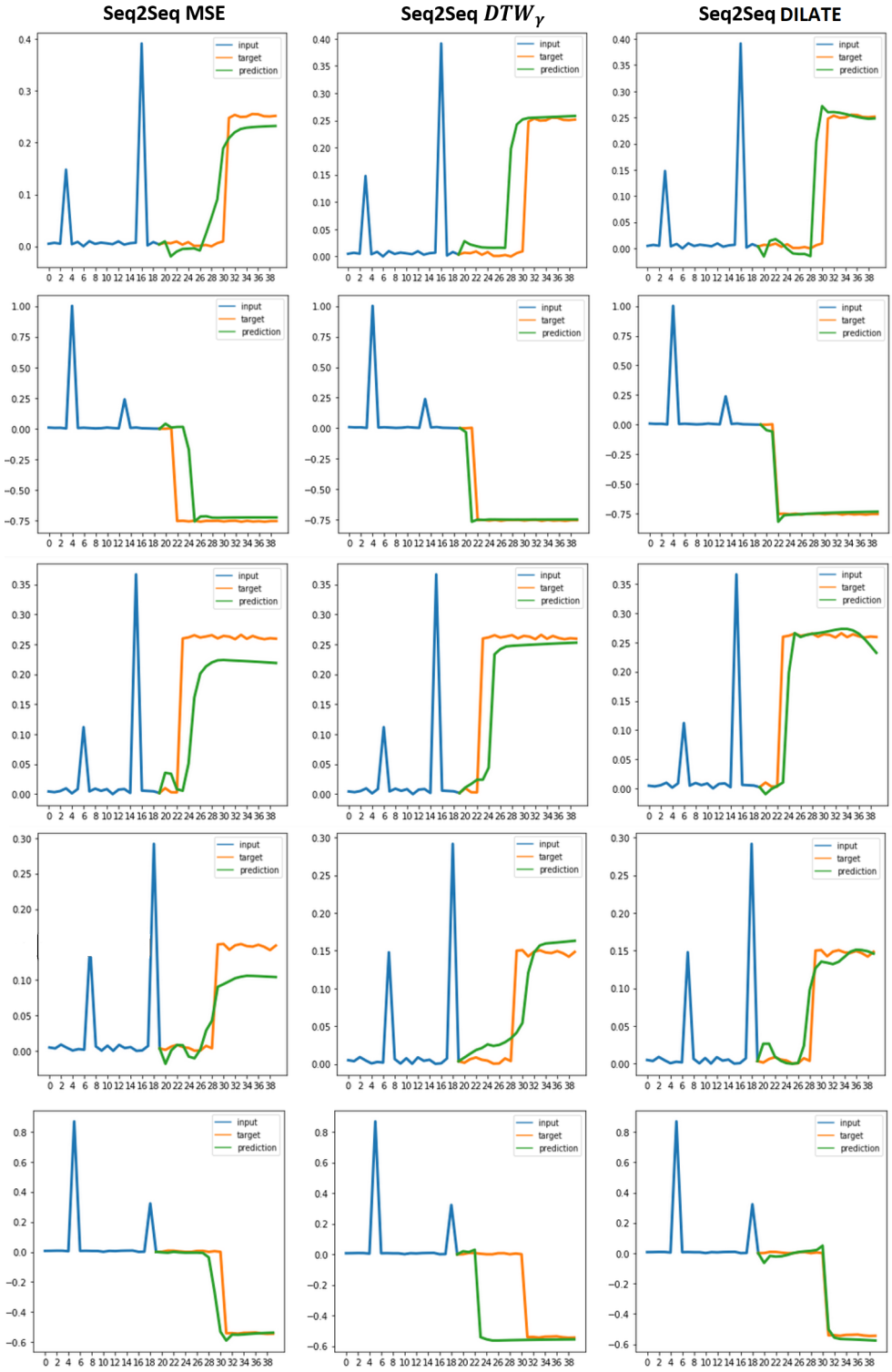


Figure B.1: Qualitative predictions for the `Synthetic-det` dataset.

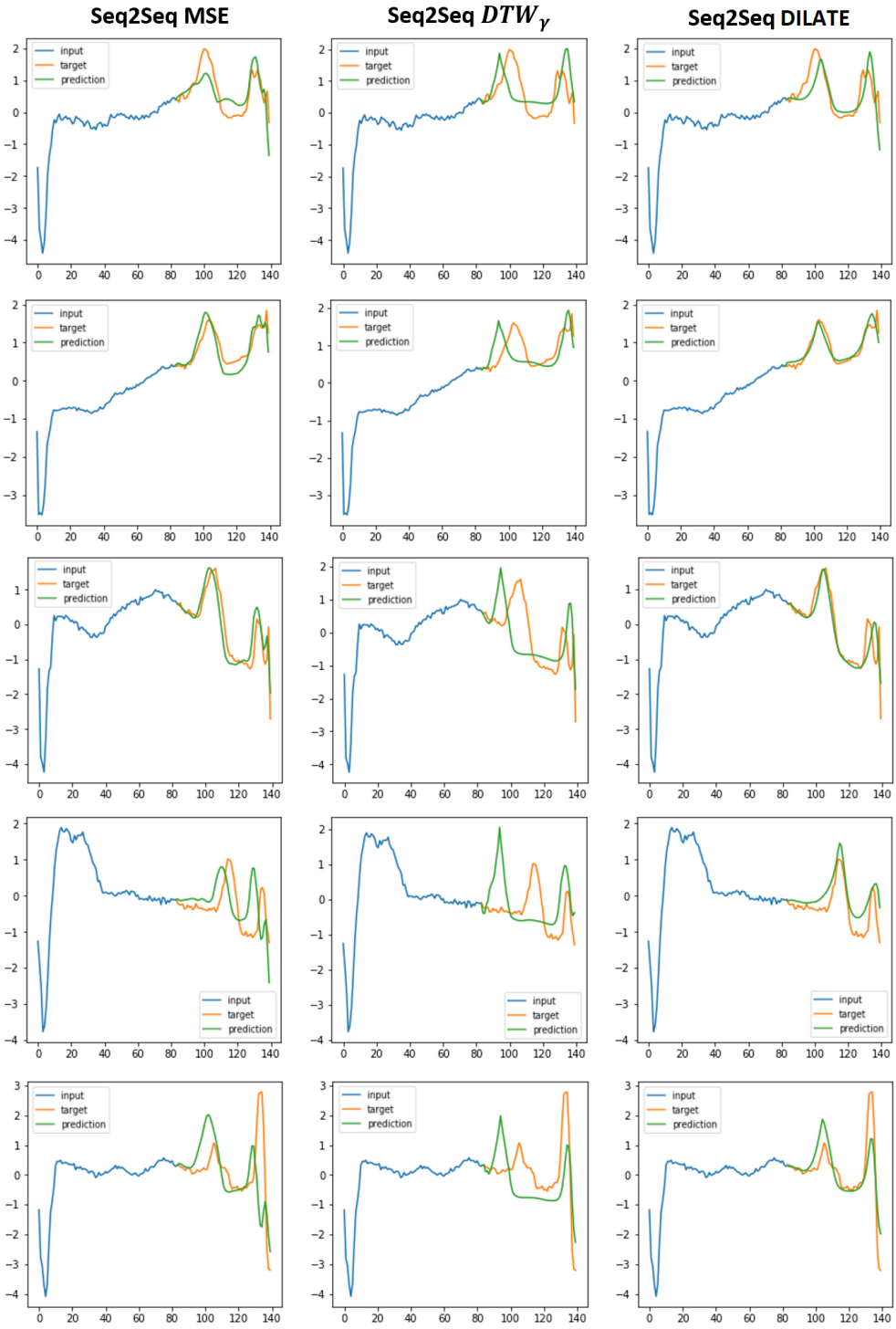


Figure B.2: Qualitative predictions for the ECG5000 dataset.

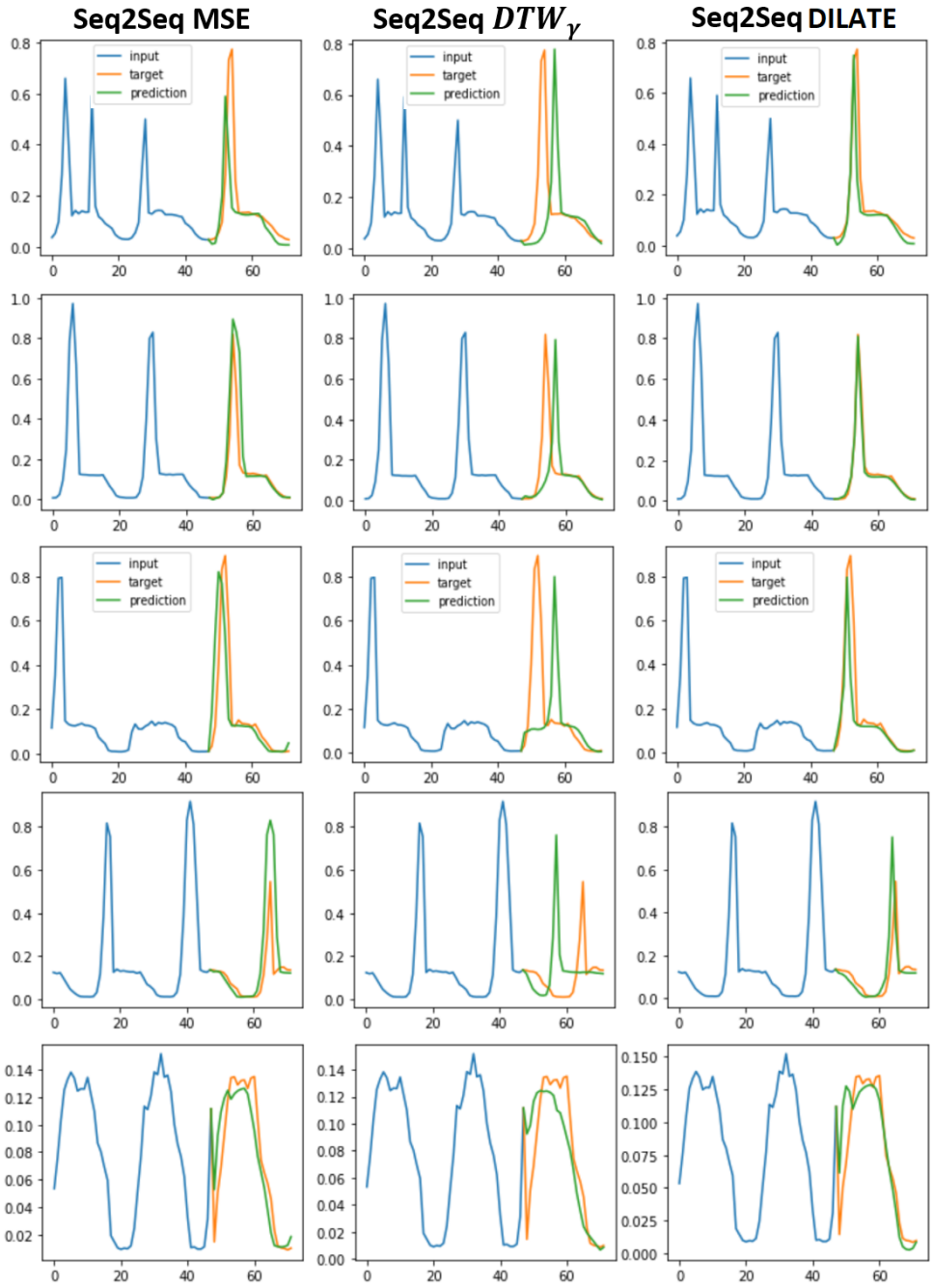


Figure B.3: Qualitative predictions for the `Traffic` dataset.

Appendix C

Appendix for STRIPE

C.1 STRIPE implementation details

Neural network architectures: STRIPE++ is composed of a Sequence To Sequence predictive model. The encoder is a recurrent neural network (RNN) with 1 layer of 128 Gated Recurrent Units (GRU) [39] units, producing a latent state h of dimension 128. We fixed by cross-validation the dimension of each diversifying variable z_s or z_t to be $k = 8$. The decoder is another RNN with $128 + 8 + 8 = 144$ GRU units followed by fully connected layers responsible for producing the future trajectory.

The Posterior network has a similar architecture as the encoder: it is a RNN with 1 layer of 128 GRU units that takes as input the full series $(\mathbf{x}_{1:T}, \mathbf{y}^*_{T+1:T+H})$, followed by two multi-layer perceptrons (MLP) dedicated to output the parameters (μ^*_s, σ^*_s) and (μ^*_t, σ^*_t) of the Gaussian distribution from which to sample the posterior diversifying variables z^*_s and z^*_t.

The $\text{STRIPE}^{++}_{\text{shape}}$ and $\text{STRIPE}^{++}_{\text{time}}$ proposal mechanisms build on top of the encoder (that produces h) with a MLP with 3 layers of 512 neurons (with Batch Normalization and LeakyReLU activations) and a final linear layer to produce $N = 10$ latent codes of dimension $k = 8$ (corresponding to the proposals for z_s or z_t).

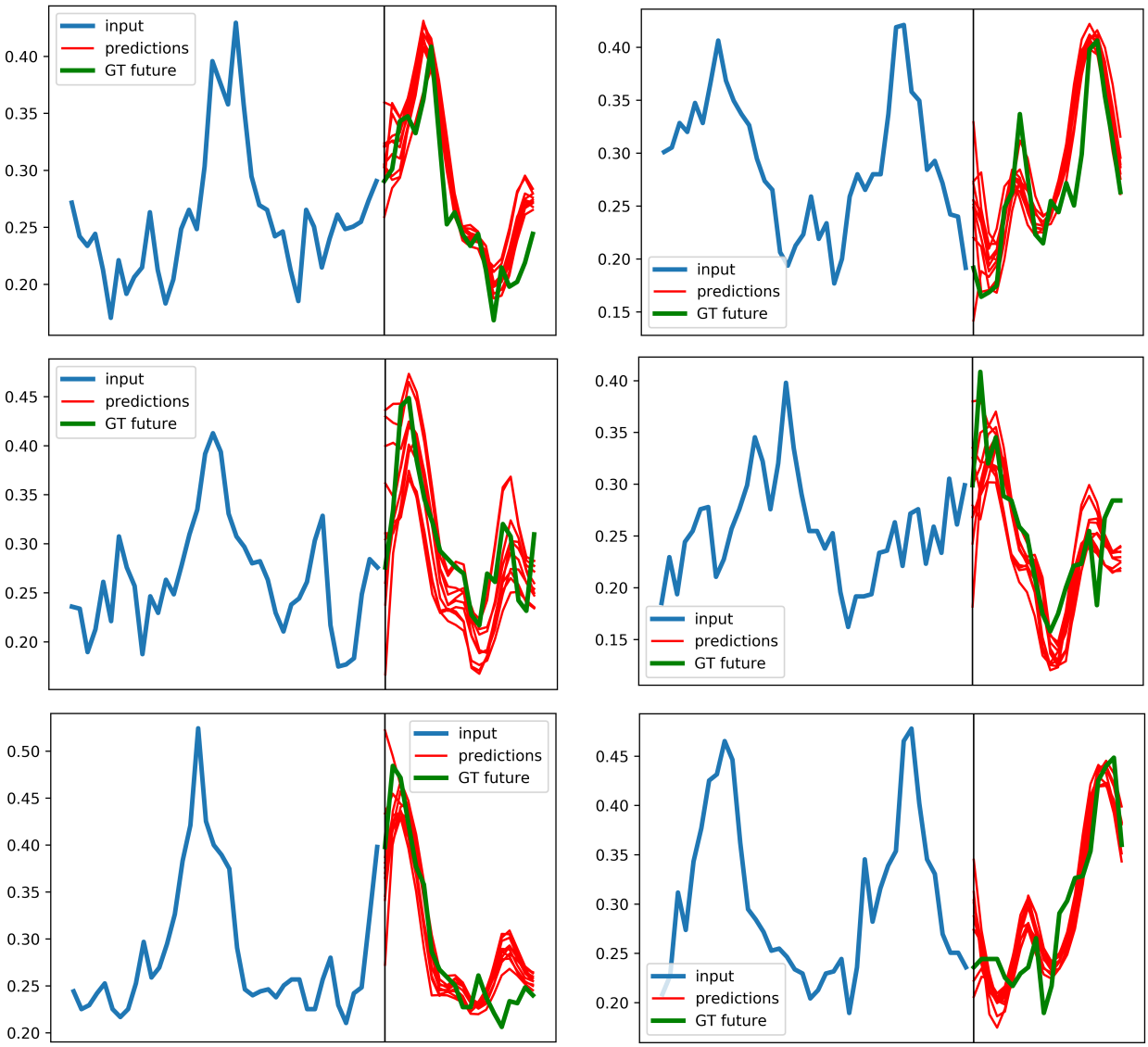
STRIPE hyperparameters: We cross-validated the relevant hyperparameters of STRIPE:

- k: dimension of the diversifying latent variables z. This dimension should be chosen relatively to the hidden size of the RNN encoders and decoders (128 in our experiments). We fixed $k = 8$ in all cases.
- N: the number of future trajectories to sample. We fixed $N = 10$. We performed a sensibility analysis to this parameter in paper Figure 8.
- $\mu = 20$: quality constraint hyperparameter in the DPP kernels.

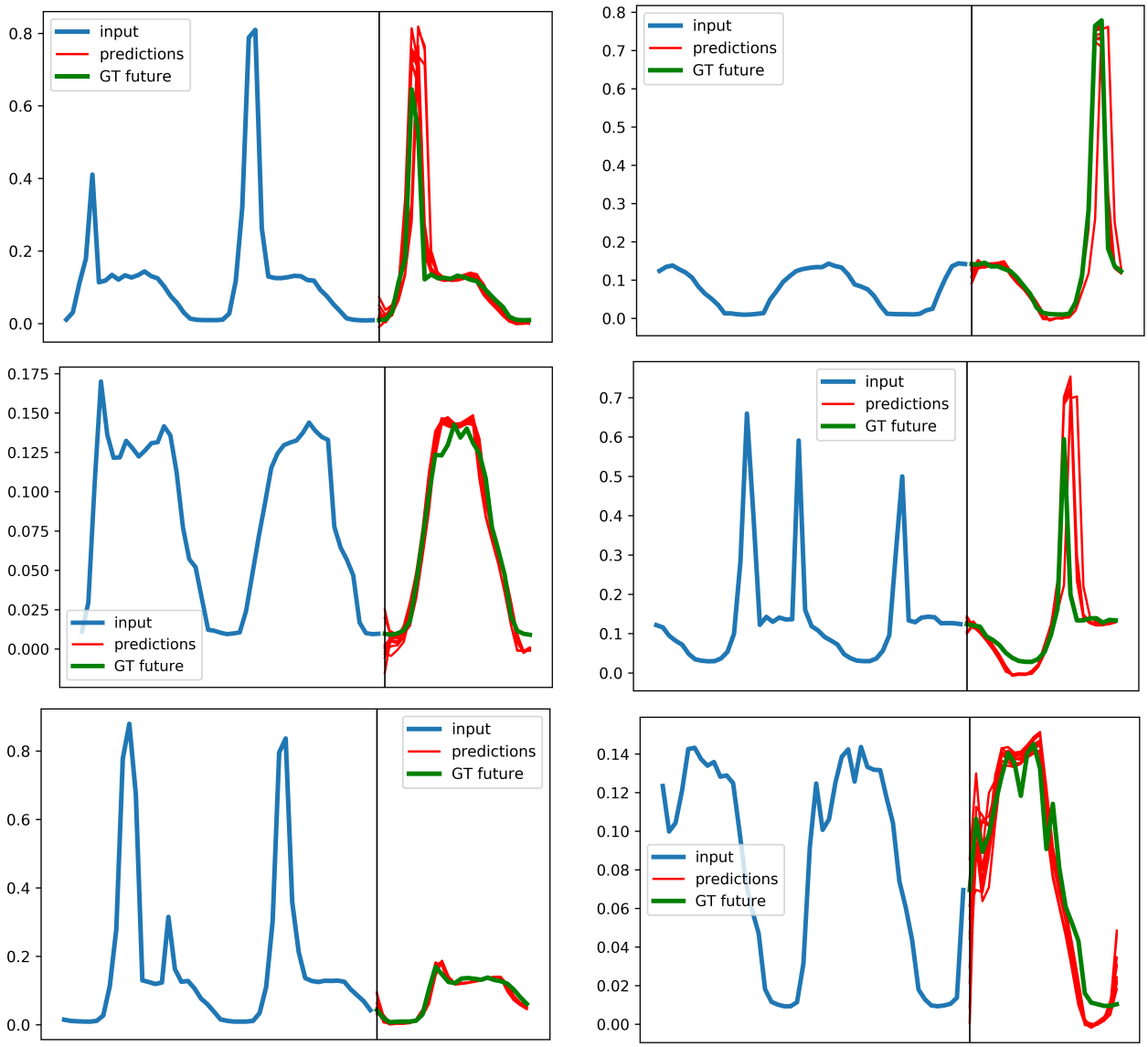
C.2 STRIPE additional visualizations

Wee provide additional visualizations for the Traffic and Electricity datasets that confirm that STRIPE predictions are both diverse and sharp.

C.2.0.1 Electricity



C.2.0.2 Traffic



Appendix D

Appendix for PhyDNet

D.1 PhyDNet model

D.1.1 Discrete PhyCell derivation

PhyCell dynamics is governed by the PDE:

$$\begin{aligned}\frac{\partial \mathbf{h}}{\partial t}(t,\mathbf{x}) &= \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h},\mathbf{u}) \\ &= \Phi(\mathbf{h}(t,\mathbf{x})) + \mathbf{K}(t,\mathbf{x}) \odot (\mathbf{E}(\mathbf{u}(t,\mathbf{x})) - (\mathbf{h}(t,\mathbf{x}) + \Phi(\mathbf{h}(t,\mathbf{x}))).\end{aligned}$$

By Euler discretization $\frac{\partial \mathbf{h}}{\partial t} = \delta \mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_{t-1}$, we get:

$$\begin{aligned}\mathbf{h}_{t+1} - \mathbf{h}_t &= \Phi(\mathbf{h}_t) + \mathbf{K}_t \odot (\mathbf{E}(\mathbf{u}_t) - (\mathbf{h}_t + \Phi(\mathbf{h}_t))) \\ \mathbf{h}_{t+1} &= \mathbf{h}_t + \Phi(\mathbf{h}_t) + \mathbf{K}_t \odot (\mathbf{E}(\mathbf{u}_t) - (\mathbf{h}_t + \Phi(\mathbf{h}_t))) \\ \mathbf{h}_{t+1} &= (1 - \mathbf{K}_t) \odot (\mathbf{h}_t + \Phi(\mathbf{h}_t)) + \mathbf{K}_t \odot \mathbf{E}(\mathbf{u}_t).\end{aligned}$$

D.1.2 Moment matrix

For a filter $\mathbf{w}$ of size $k \times k$, the moment matrix $\mathbf{M}(\mathbf{w})$ is a matrix of size $k \times k$ defined as:

$$\mathbf{M}(\mathbf{w})_{i,j} = \frac{1}{i!j!} \sum_{u=-\frac{k-1}{2}}^{\frac{k-1}{2}} \sum_{v=-\frac{k-1}{2}}^{\frac{k-1}{2}} u^i v^j \mathbf{w}[u,v],$$

for $i,j = 0,...,k-1$.

For any function $h : \mathbb{R}^2 \longrightarrow \mathbb{R}$, we consider the convolution of h with the filter $\mathbf{w}$. Taylor's expansion

gives:

$$\sum_{u=-\frac{k-1}{2}}^{\frac{k-1}{2}}\sum_{v=-\frac{k-1}{2}}^{\frac{k-1}{2}}\mathbf{w}[u,v]h(x+\delta x.u, y+\delta y.v) = \sum_{u=-\frac{k-1}{2}}^{\frac{k-1}{2}}\sum_{v=-\frac{k-1}{2}}^{\frac{k-1}{2}}\mathbf{w}[u,v]\sum_{i,j=1}^{k-1}\frac{\partial^{i+j}h}{\partial x^i\partial y^j}(x,y)\frac{u^i v^j}{i!j!}\delta x^i\delta y^j$$
$$+o(|\delta x|^{k-1}+|\delta y|^{k-1})$$
$$=\sum_{i,j=1}^{k-1}\mathbf{M}(\mathbf{w})_{i,j}\delta x^i\delta y^j\frac{\partial^{i+j}h}{\partial x^i\partial y^j}(x,y)+o(|\delta x|^{k+1}+|\delta y|^{k-1}).$$

This equation shows that we can control the differential order approximated by the filter $\mathbf{w}$ by imposing constraints on its moment matrix $\mathbf{M}(\mathbf{w})$. For example, in order to approximate the differential operator $\frac{\partial^{a+b}}{\partial x^a\partial y^b}(.)$, it suffices to impose $\mathbf{M}(\mathbf{w})_{i,j}=0$ for $i\neq a$ and $j\neq b$. By denoting $\mathbf{\Delta}^k_{i,j}$ the Kronecker matrix of size $k\times k$, which equals 1 at position (i,j) and 0 elsewhere, we thus enforce the moment matrix $\mathbf{M}(\mathbf{w})$ to match the target $\mathbf{\Delta}^k_{a,b}$ with the Frobenius norm. This justifies the choice of our moment loss for enforcing each filter $\mathbf{w}^k_{p,i,j}$ to approximate the corresponding derivative $\frac{\partial^{i+j}}{\partial x^i\partial y^j}(.)$:

$$\mathcal{L}_{\text{moment}}=\sum_{i\leq k}\sum_{j\leq k}||\mathbf{M}(\mathbf{w}^k_{p,i,j})-\mathbf{\Delta}^k_{i,j}||_F.$$

D.1.3 Prediction mode training

We show in section D.1.3.1 that the decomposition $\boldsymbol{\mathcal{M}}_r(\mathbf{h},\mathbf{u})=\Phi(\mathbf{h})+\mathcal{C}(\mathbf{h},\mathbf{u})$ still holds for standard Seq2Seq models (RNN, GRU, LSTM). As mentioned in Chapter 6, the resulting predictor Φ is, however, naive and useless for multi-step prediction, i.e. $\Phi(\mathbf{h})=-\mathbf{h}$ and $\tilde{\mathbf{h}}_{t+1}=0$.

In multi-step prediction, the option followed by standard Seq2seq models is to recursively reinject back predictions as ground truth input for the next time steps. Scheduled Sampling [14] is a solution to mitigate error accumulation and train/test discrepancy, that we use in our ConvLSTM branch. This is, however, inferior to the results obtained with our PhyCell trained in the "prediction-only" mode, as shown in Section 6.4.4.

D.1.3.1 PDE formulation for standard RNNs

Vanilla RNN The equations for the vanilla RNN are:

$$\mathbf{h}_t=\tanh(\mathbf{W}_h\mathbf{h}_{t-1}+\mathbf{W}_u\mathbf{u}_t+\mathbf{b}),$$

with weight matrices $\mathbf{W}_h$, $\mathbf{W}_u$ and bias $\mathbf{b}$. By approximating $\frac{\partial\mathbf{h}}{\partial t}=\delta\mathbf{h}_t=\mathbf{h}_t-\mathbf{h}_{t-1}$, we get the PDE:

$$\frac{\partial\mathbf{h}}{\partial t}(t,\mathbf{x})=\boldsymbol{\mathcal{M}}(\mathbf{h},\mathbf{u})$$
$$=\tanh(\mathbf{W}_h\mathbf{h}(t)+\mathbf{W}_u\mathbf{u}(t)+\mathbf{b})-\mathbf{h}(t).$$

A linear decoupling of this PDE is

$$\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u}),$$

with $\Phi(\mathbf{h}) = -\mathbf{h}(t)$ and $\mathcal{C}(\mathbf{h}, \mathbf{u}) = \tanh(\mathbf{W}_h \mathbf{h}(t) + \mathbf{W}_u \mathbf{u}(t) + \mathbf{b})$ which gives in discrete time the prediction-correction scheme:

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = 0 & \text{(D.1)} \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \tanh\left(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_u \mathbf{u}_t + \mathbf{b}\right). & \text{(D.2)} \end{cases}$$

We see that the prior predictor Φ brings no information and that the correction step drives the whole dynamics.

Gated Recurrent Unit (GRU) The equations of the Gated Recurrent Unit [39] are:

$$\begin{aligned} \mathbf{r}_t &= \sigma(\mathbf{W}_{rh} \mathbf{h}_{t-1} + \mathbf{W}_{ru} \mathbf{u}_t + \mathbf{b}_r) \\ \mathbf{z}_t &= \sigma(\mathbf{W}_{zh} \mathbf{h}_{t-1} + \mathbf{W}_{zu} \mathbf{u}_t + \mathbf{b}_z) \\ \mathbf{g}_t &= \tanh(\mathbf{W}_{gh}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{W}_{gu} \mathbf{u}_t + \mathbf{b}_g) \\ \mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t, \end{aligned}$$

where $\mathbf{r}_t$ is the reset gate, $\mathbf{z}_t$ is the update gate and $\mathbf{g}_t$ is the update vector. By approximating $\frac{\partial \mathbf{h}}{\partial t} = \delta \mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_{t-1}$, we get the PDE:

$$\begin{aligned} \frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) &= \boldsymbol{\mathcal{M}}(\mathbf{h}, \mathbf{u}) \\ &= \mathbf{z}(t) \odot \mathbf{h}(t) + (1 - \mathbf{z}(t)) \odot \mathbf{g}(t) - \mathbf{h}(t). \end{aligned}$$

A linear decoupling of this PDE is

$$\frac{\partial \mathbf{h}}{\partial t}(t, \mathbf{x}) = \Phi(\mathbf{h}) + \mathcal{C}(\mathbf{h}, \mathbf{u}),$$

with $\Phi(\mathbf{h}) = -\mathbf{h}(t)$ and $\mathcal{C}(\mathbf{h}, \mathbf{u}) = \mathbf{z}(t) \odot \mathbf{h}(t) + (1 - \mathbf{z}(t)) \odot \mathbf{g}(t)$ which gives in discrete time the prediction-correction scheme:

$$\begin{cases} \tilde{\mathbf{h}}_{t+1} = 0 & \text{(D.3)} \\ \mathbf{h}_{t+1} = \tilde{\mathbf{h}}_{t+1} + \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t. & \text{(D.4)} \end{cases}$$

We again see that the prior predictor Φ brings no information and that the correction step drives the whole dynamics.

Long Short-Term Memory (LSTM) We give the formulation for the standard LSTM [102] (the ConvLSTM [281] can be immediately deduced by replacing matrix products by convolutions):

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{iu}\mathbf{u}_t + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{W}_{fu}\mathbf{u}_t + \mathbf{b}_f) \\
\mathbf{g}_t &= \tanh(\mathbf{W}_{gh}\mathbf{h}_{t-1} + \mathbf{W}_{gu}\mathbf{u}_t + \mathbf{b}_g) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\
\mathbf{o}_t &= \sigma(\mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{ou}\mathbf{u}_t + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t).
\end{aligned}$$

where $\mathbf{i}_t$ is the input gate, $\mathbf{f}_t$ the forget gate, $\mathbf{g}_t$ the input-modulation gate, $\mathbf{o}_t$ the output gate, $\mathbf{c}_t$ the cell state and $\mathbf{h}_t$ the latent state. We define the LSTM augmented latent state as:

$$\bar{\mathbf{h}} = \begin{pmatrix} \mathbf{g} \\ \mathbf{c} \end{pmatrix}.$$

The augmented state $\bar{\mathbf{h}}$ thus verifies the PDE:

$$\frac{\partial \bar{\mathbf{h}}}{\partial t} = \begin{pmatrix} \frac{\partial \mathbf{h}}{\partial t} \\ \frac{\partial \mathbf{c}}{\partial t} \end{pmatrix} = \begin{pmatrix} \mathbf{o}(t) \odot \tanh(\mathbf{c}(t)) - \mathbf{h}(t)) \\ \mathbf{f}(t) \odot \mathbf{c}(t) + \mathbf{i}(t) \odot \mathbf{g}(t) - \mathbf{c}(t) \end{pmatrix}.$$

A linear decoupling of this PDE is

$$\frac{\partial \bar{\mathbf{h}}}{\partial t}(t, \mathbf{x}) = \Phi(\bar{\mathbf{h}}) + \mathcal{C}(\bar{\mathbf{h}}, \mathbf{u}),$$

with $\Phi(\bar{\mathbf{h}}) = -\bar{\mathbf{h}}(t)$ and

$$\mathcal{C}(\bar{\mathbf{h}}, \mathbf{u}) = \begin{pmatrix} \mathbf{o}(t) \odot \tanh(\mathbf{c}(t)) \\ \mathbf{f}(t) \odot \mathbf{c}(t) + \mathbf{i}(t) \odot \mathbf{g}(t) \end{pmatrix},$$

which gives in discrete time the prediction-correction scheme:

$$\begin{cases} \tilde{\bar{\mathbf{h}}}_{t+1} = 0 & \text{(D.5)} \\ \bar{\mathbf{h}}_{t+1} = \tilde{\bar{\mathbf{h}}}_{t+1} + \begin{pmatrix} \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \\ \mathbf{f}_t \odot \mathbf{c}_t + \mathbf{i}_t \odot \mathbf{g}_t \end{pmatrix}. & \text{(D.6)} \end{cases}$$

We again see that the prior predictor Φ brings no information and that the correction step drives the whole dynamics.

D.2 Experiments

D.2.1 Model architectures and training

Model architectures We give here the architecture of the encoder and decoder for all datasets. They share common building blocs, composed of convolutions, GroupNorm activation functions [280] and LeakyRelu non-linearities. For each of the following architectures, we use skip connections from the encoder to the decoder, as classically done, e.g. in [60]. We define:

- conv-block(input, output, stride) = {Conv2D + GroupNorm + LeakyRelu(0.2)}
- upconv-block(input,output,stride)={TransposedConv2D + GroupNorm + LeakyRelu(0.2) }
- upconv(input,output,stride)=TransposedConv2D(input, output, stride)

Moving MNIST:

Encoder	Decoder
conv-block(1,8,1)	upconv-block(128,64,1)
conv-block(8,16,1)	upconv-block(128,32,2)
conv-block(16,32,2)	upconv-block(64,32,1)
conv-block(32,32,1)	upconv-block(64,16,2)
conv-block(32,64,2)	upconv-block(32,8,1)
conv-block(64,64,1)	upconv(16,1,1)

Traffic:

Encoder	Decoder
conv-block(2,32,1)	upconv-block(256,64,1)
conv-block(32.64,2)	upconv-block(128,32,2)
conv-block(64,128,1)	upconv(64,2,1)

SST:

Encoder	Decoder
conv-block(1,32,1)	upconv-block(256,64,1)
conv-block(32.64,2)	upconv-block(128,32,2)
conv-block(64,128,1)	upconv(64,1,1)

Human 3.6:

Encoder	Decoder
conv-block(3,16,1)	upconv-block(256,128,1)
conv-block(16,32,1)	upconv-block(256,64,2)
conv-block(32,64,2)	upconv-block(128,64,1)
conv-block(64,64,1)	upconv-block(128,32,2)
conv-block(64,128,2)	upconv-block(64,16,1)
conv-block(128,128,1)	upconv(32,3,1)

Influence of λ We show in Figure D.1 the influence of parameter λ balancing $\mathcal{L}_{\text{image}}$ and $\mathcal{L}_{\text{moment}}$ when training PhyDNet for Moving MNIST dataset. When λ decreases towards 0, MSE tends towards the unconstrained case at 29. MSE reaches a minimum around $\lambda = 1$. When λ further increases, physical regularization is too high and MSE increases above 30. In the paper, we fix $\lambda = 1$ for all datasets.

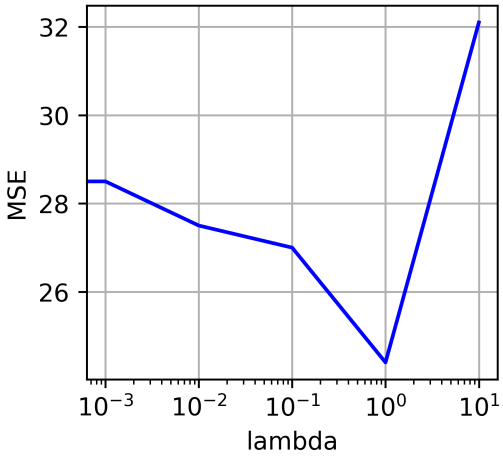


Figure D.1: Influence of hyperparameter λ when training PhyDNet for Moving MNIST dataset.

D.2.2 State-of-the art comparison

We show here that PhyDNet results are equivalent on Human 3.6 to a recent baseline that explicitly uses additional human pose annotations [259]. In the supplementary of their paper [259], the authors evaluate their model with Peak Signal over Noise Ratios (PSNR) curves with respect to the forecasting horizon for all deciles of motion in Human 3.6 videos. Regarding prediction horizon up to $H = 4$, their method obtains a PSNR always below 21 and around 22 for the 1^{st} decile (with the least human motion). In comparison, PhyDNet attains a per-frame MSE of 369, corresponding to a PSNR of 21.2. This shows that PhyDNet performs similarly than [259] for the prediction horizon considered, without requiring additional human pose annotations.

Table D.1: A detailed ablation study shows the impact of the physical regularization $\mathcal{L}_{\text{moment}}$ on the performances of PhyCell and PhyDNet for all datasets.

Method	**Moving MNist**			**Traffic BJ**			**Sea Surface Temperature**			**Human 3.6**		
	MSE	MAE	SSIM	MSE ×100	MAE	SSIM	MSE ×10	MAE	SSIM	MSE /10	MAE /100	SSIM
ConvLSTM	103.3	182.9	0.707	48.5*	17.7*	0.978*	45.6*	63.1*	0.949*	50.4*	18.9*	0.776*
PhyCell	50.8	129.3	0.870	48.9	17.9	0.978	38.2	60.2	0.969	42.5	18.3	0.891
PhyCell without $\mathcal{L}_{\text{moment}}$	43.4	112.8	0.895	43.6	16.89	0.980	35.4	56.0	0.970	39.6	17.4	0.894
PhyDNet	**24.4**	**70.3**	**0.947**	**41.9**	**16.2**	**0.982**	**31.9**	53.3	**0.972**	36.9	16.2	0.901
PhyDNet without $\mathcal{L}_{\text{moment}}$	29.0	81.2	0.934	43.9	16.6	0.981	32.3	**53.1**	0.971	**36.7**	**15.9**	**0.904**

D.2.3 Ablation study

We give in Figure D.6 additional visualisations completing Figure D.6. We qualitatively analyze partial predictions of PhyDNet for the physical branch $\hat{\mathbf{u}}^{\mathbf{p}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{p}}_{t+1})$ and residual branch $\hat{\mathbf{u}}^{\mathbf{r}}_{t+1} = \mathbf{D}(\mathbf{h}^{\mathbf{r}}_{t+1})$. For Moving MNIST (a) and Human 3.6 (d), $\mathbf{h}^{\mathbf{p}}$ captures coarse localisations of objects, while $\mathbf{h}^{\mathbf{r}}$ captures fine-grained details that are not useful for the physical model. For Traffic BJ, $\mathbf{h}^{\mathbf{p}}$ captures the main patterns of the road network, while $\mathbf{h}^{\mathbf{r}}$ models remaining details. Finally for SST, the visual difference between $\mathbf{h}^{\mathbf{p}}$ and $\mathbf{h}^{\mathbf{r}}$ is slighter, but the cooperation between both branches is crucial, as shown by quantitatives results.

D.2.4 Influence of physical regularization

We provide the detailed ablation study for all datasets in Table D.1 that complements Table 6.2. When we disable $\mathcal{L}_{\text{moment}}$ for training PhyCell, performances improve for all datasets (improvement of 7 MSE points for Moving MNIST, 5 points for Traffic BJ, 3 points for SST and Human 3.6). This again shows that physical constraints alone are too restrictive for learning dynamics in a general context, where other factors are required for prediction. When we further include PhyCell in our two-branches disentangling architecture PhyDNet, there is another huge performance gain compared to PhyCell (improvement of 25 MSE points on Moving MNIST, 7 points for Traffic and SST, 5 points for Human 3.6). We also remark that when we disable $\mathcal{L}_{\text{moment}}$ for training PhyDNet, we get worse performances (drop of 5 MSE points for Moving MNIST and 2 points for Traffic) or equivalent performances (difference below 0.5 MSE point for SST and Human 3.6). This again confirms the relevance of physical constraints.

D.2.5 Additional visualisations

We give further qualitative prediction of PhyDNet on Traffic BJ (Figure D.2) with a comparison with Memory in Memory [269] that is state-of-the-art for this dataset. We see that PhyDNet leads to sharper results and a lower absolute error. Interestingly, PhyDNet absolute errors are approximately spatially independent, whereas MIM errors tend to be higher at a few keys locations of Beijing road network.

We also provide additional prediction visualisations for Sea Surface Temperature (Figure D.3) and Human 3.6 (Figure D.4) which confirm the good behaviour of PhyDNet.

We add a detailed qualitative comparison to DDPAE in Figure D.5. DDPAE is a specific disentangling method for Moving MNIST that extracts the positions of the two digits and tracks them

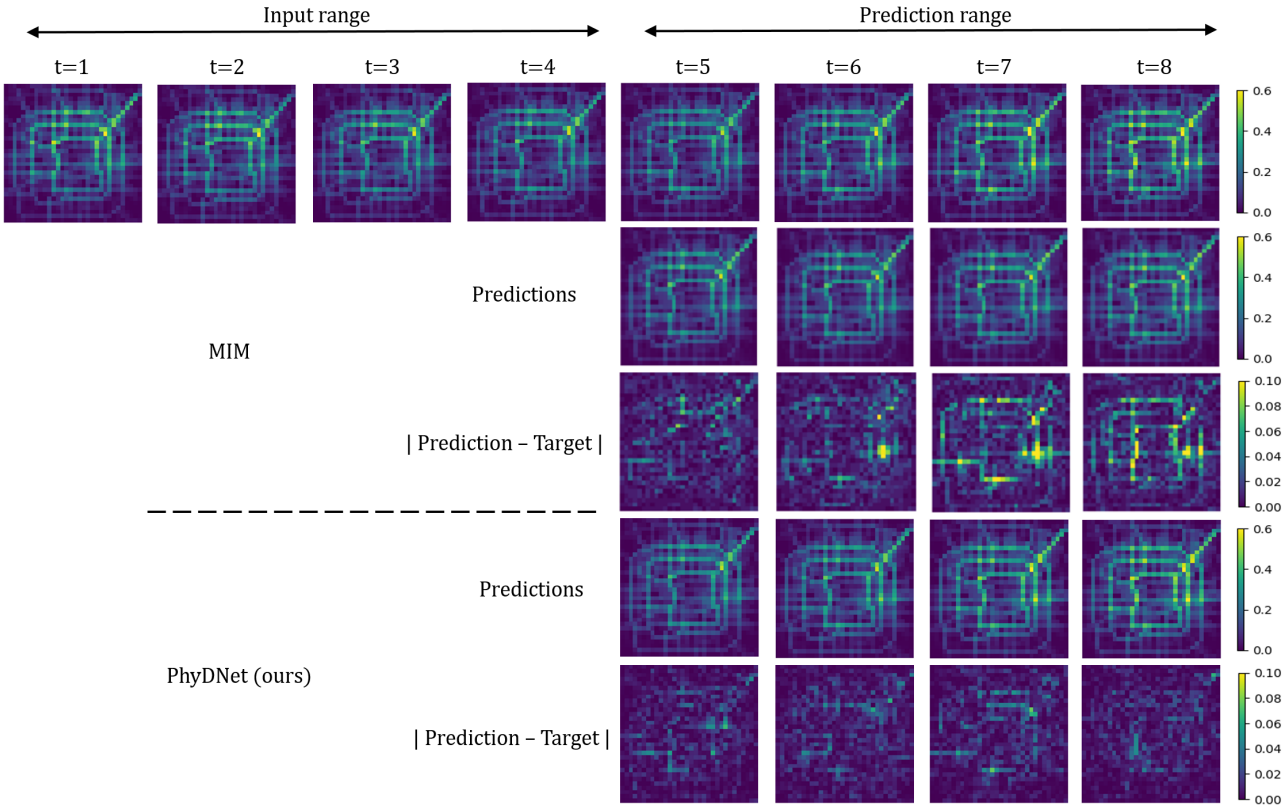


Figure D.2: Additional qualitative results for Traffic BJ and comparison to Memory In Memory [269]. We see that PhyDNet absolute error are smaller than MIM errors, and independent of the spatial structure of the road network.

with a predictive recurrent neural network. In this example, DDPAE fails to disentangle the two digits (components 1 and 2) in Figure D.5 when they overlap in the input sequence, resulting in blurry predictions. In contrast, PhyDNet successfully learns a latent space in which the two digits are disentangled, resulting in far better predictions in terms of sharpness and position of the digits.

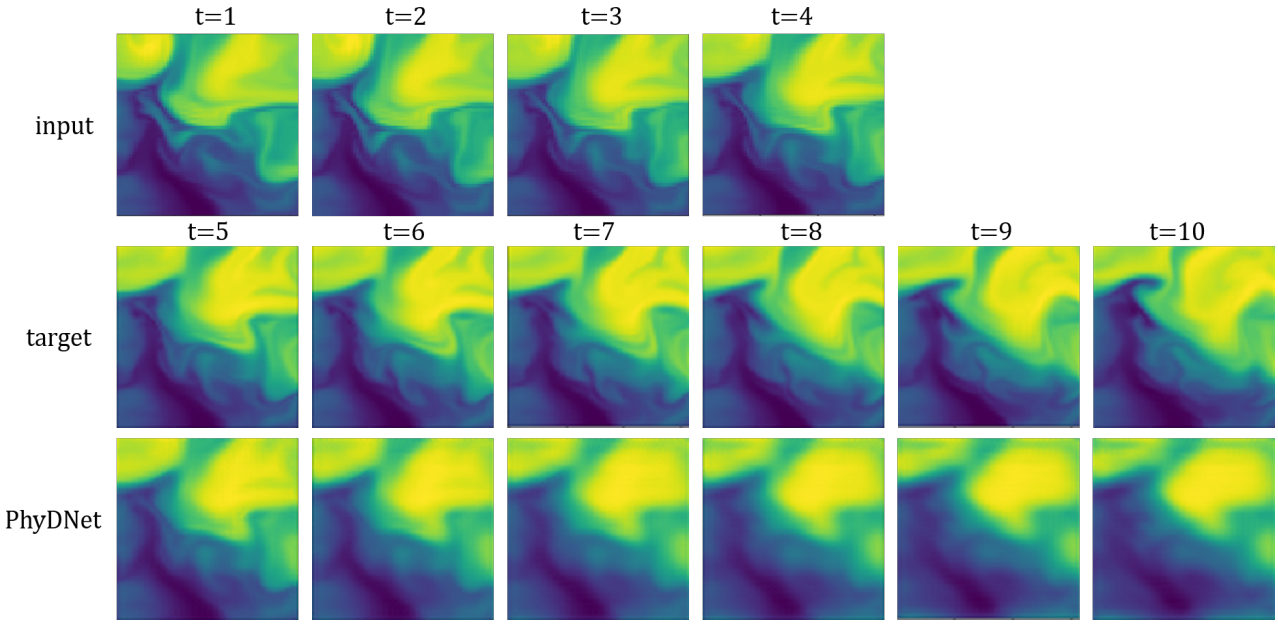


Figure D.3: PhyDNet additional qualitative results for Sea Surface Temperature.

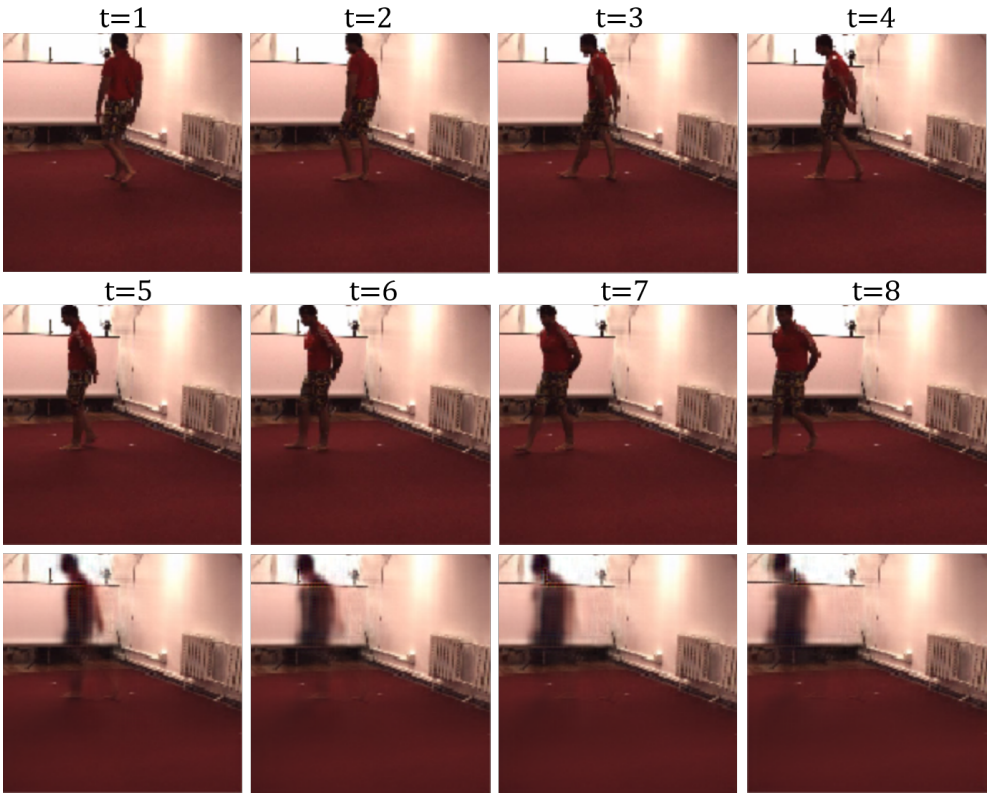


Figure D.4: PhyDNet additional qualitative results for Human 3.6.

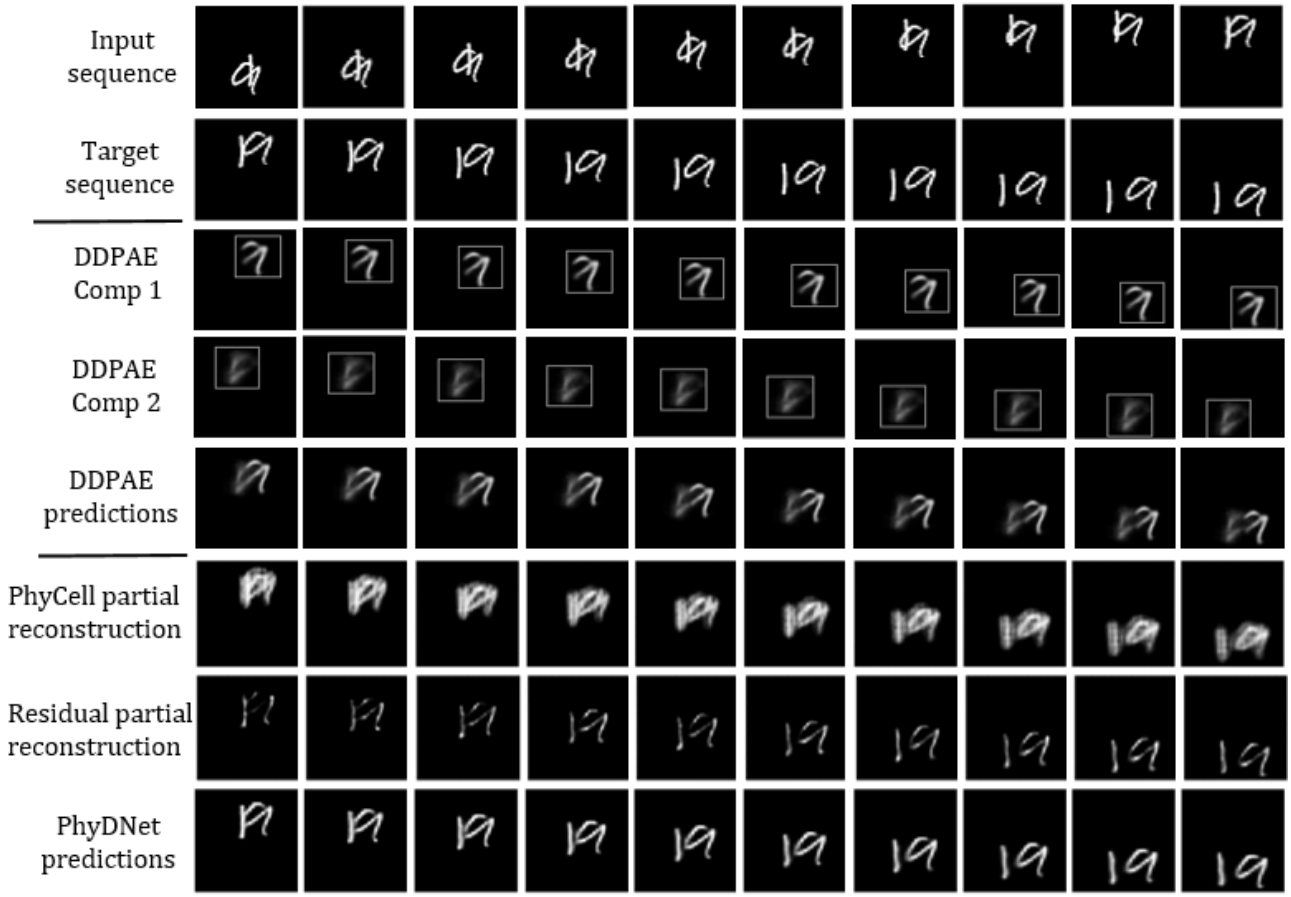
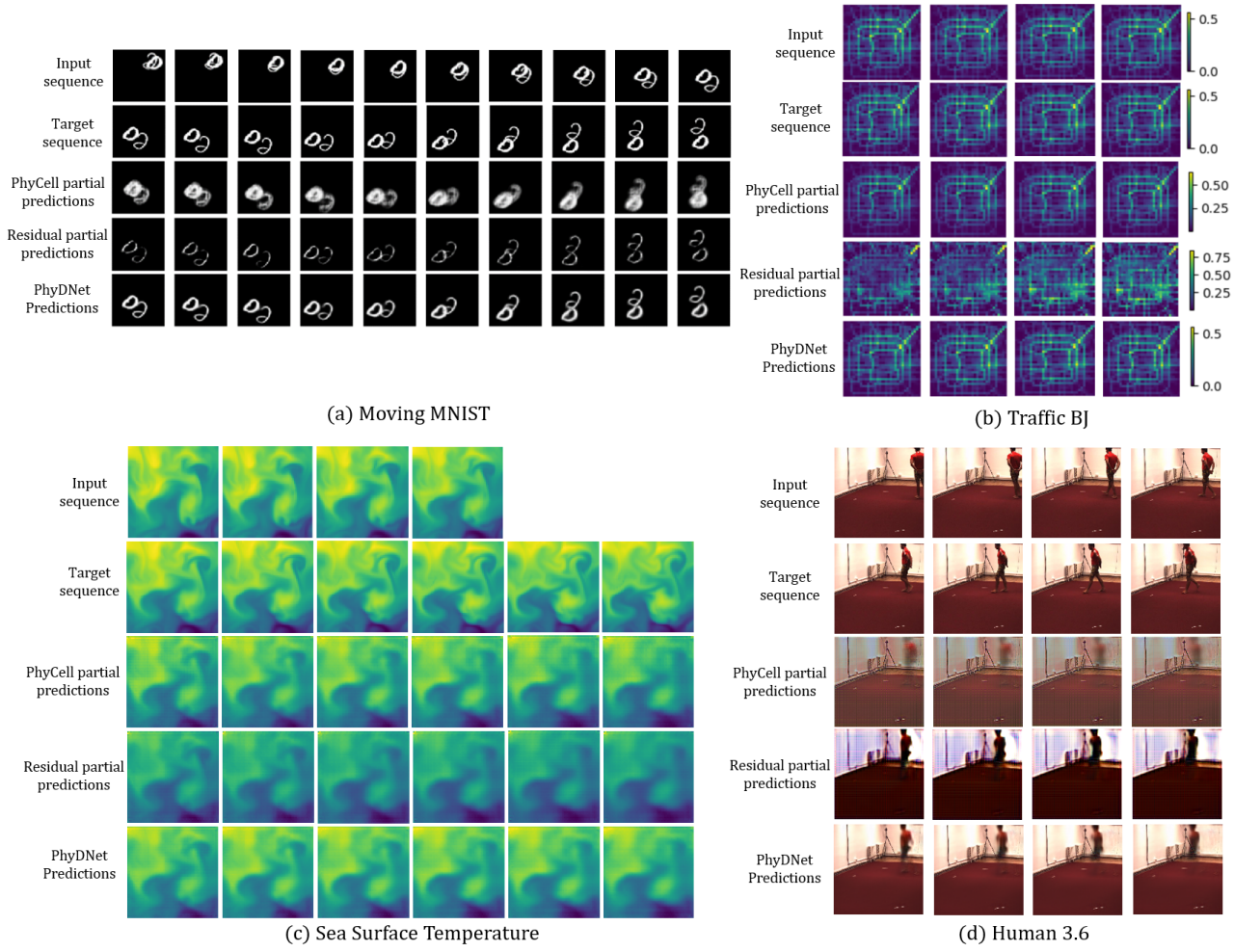


Figure D.5: Detailed qualitative comparison to DDPAE [104] on Moving MNIST dataset.



(a) Moving MNIST

(b) Traffic BJ

(c) Sea Surface Temperature

(d) Human 3.6

Figure D.6: PhyDNet additional ablation visualisations for all datasets.

Appendix E

Appendix for APHYNITY

E.1 Reminder on proximinal and Chebyshev sets

We begin by giving a definition of proximinal and Chebyshev sets, taken from [78]:

Definition 1 *A proximinal set of a normed space $(E, \|\cdot\|)$ is a subset $\mathcal{C} \subset E$ such that every $x \in E$ admits at least a nearest point in $\mathcal{C}$.*

Definition 2 *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 $\mathcal{C}$.*

Proximinality reduces to a compacity condition in finite dimensional spaces. In general, it is a weaker one: boundedly compact sets verify this property for example.

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 [114] 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.

E.2 Proof of Propositions 1 and 2

We prove the following result which implies both propositions in the article:

Proposition 4 *The optimization problem:*

$$\min_{F_p \in \mathcal{F}_p, F_a \in \mathcal{F}} \quad \|F_a\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{\mathrm{d}X_t}{\mathrm{d}t} = (F_p + F_a)(X_t) \tag{E.1}$$

is equivalent a metric projection onto $\mathcal{F}_p$.

If $\mathcal{F}_p$ is proximinal, Eq E.1 admits a minimizing pair.

If $\mathcal{F}_p$ is Chebyshev, Eq E.1 admits a unique minimizing pair which F_p is the metric projection.

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:

$$F^{\mathcal{D}}(a) = \left.\frac{\mathrm{d}X_t}{\mathrm{d}t}\right|_{t=t_0}.$$

For any (F_p, F_a) satisfying the constraint in Eq E.1, we then have that $(F_p + F_a)(a) = {}^{\mathrm{d}X_t}/_{\mathrm{d}t|_{t_0}} = F^{\mathcal{D}}(a)$ for all $a \in \mathcal{A}$. Conversely, any pair such that $(F_p, F_a) \in \mathcal{F}_p \times \mathcal{F}$ and $F_p + F_a = F^{\mathcal{D}}$, verifies the constraint.

Thus we have the equivalence between Eq E.1 and the metric projection formulated as:

$$\underset{F_p \in \mathcal{F}_p}{\text{minimize}} \quad \left\|F^{\mathcal{D}} - F_p\right\|. \tag{E.2}$$

If $\mathcal{F}_p$ is proximinal, the projection problem admits a solution which we denote $F_p^\star$. Taking $F_a^\star = F^{\mathcal{D}} - F_p^\star$, we have that $F_p^\star + F_a^\star = F^{\mathcal{D}}$ so that $(F_p^\star, F_a^\star)$ verifies the constraint of Eq E.1. Moreover, if there is (F_p, F_a) satisfying the constraint of Eq E.1, we have that $F_p + F_a = F^{\mathcal{D}}$ by what was shown above and $\|F_a\| = \|F^{\mathcal{D}} - F_p\| \geq \|F^{\mathcal{D}} - F_p^\star\|$ by definition of $F_p^\star$. This shows that $(F_p^\star, F_a^\star)$ is minimal.

Moreover, if $\mathcal{F}_p$ is a Chebyshev set, by uniqueness of the projection, if $F_p \neq F_p^\star$ then $\|F_a\| > \|F_a^\star\|$. Thus the minimal pair is unique.

E.3 Parameter estimation in incomplete physical models

Classically, when a set $\mathcal{F}_p \subset \mathcal{F}$ summarizing 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:

$$\begin{aligned} &\underset{F_p \in \mathcal{F}_p}{\text{minimize}} \quad \mathbb{E}_{X\sim\mathcal{D}} L(\widetilde{X}^{X_0}, X) \\ &\text{subject to} \quad \forall g \in \mathcal{I},\ \widetilde{X}_0^g = g \text{ and } \forall t,\ \frac{\mathrm{d}\widetilde{X}_t^g}{\mathrm{d}t} = F_p(\widetilde{X}_t^g). \end{aligned} \tag{E.3}$$

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_p 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 $F \notin \mathcal{F}_p$, the loss is strictly positive at the minimum. This means that reducing the space of functions $\mathcal{F}_p$ makes us lose in terms of accuracy.[1]
- The obtained function F_p might not even be the most meaningful function from $\mathcal{F}_p$ as it would try to capture phenomena which are not explainable with functions in $\mathcal{F}_p$, 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 $\mathcal{F}_p$ but not the dampening, the estimated period will account for the dampening and will thus be biased.

This is confirmed in Section 7.4: 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:

$$\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2} + \omega_0^2\theta + \alpha\frac{\mathrm{d}\theta}{\mathrm{d}t} = 0.$$

We take the same notations as in the article and parametrize the simplified physical models as:

$$F_p^a : X \mapsto (\frac{\mathrm{d}\theta}{\mathrm{d}t}, -a\theta),$$

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:

$$F : X \mapsto (\frac{\mathrm{d}\theta}{\mathrm{d}t}, -2(\theta + \frac{\mathrm{d}\theta}{\mathrm{d}t})).$$

[1] This is true in theory, although not necessarily in practice when F overfits a small dataset.

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 will not 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:

$$F_r^a : X \mapsto (0, (a-2)\theta - 2\frac{\mathrm{d}\theta}{\mathrm{d}t}).$$

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.

E.4 Discussion on supervision over derivatives

In order to find the appropriate decomposition (F_p, F_a), we use a trajectory-based error by solving:

$$\begin{aligned} \underset{F_p \in \mathcal{F}_p,\, F_a \in \mathcal{F}}{\text{minimize}} \quad & \|F_a\| \\ \text{subject to} \quad & \forall g \in \mathcal{I},\ \widetilde{X}_0^g = g \text{ and } \forall t,\ \frac{\mathrm{d}\widetilde{X}_t^g}{\mathrm{d}t} = (F_p + F_a)(\widetilde{X}_t^g), \\ & \forall X \in \mathcal{D},\ L(X, \widetilde{X}^{X_0}) = 0. \end{aligned} \tag{E.4}$$

In the continuous setting where the data is available at all times t, this problem is in fact equivalent to the following one:

$$\underset{F_p \in \mathcal{F}_p}{\text{minimize}} \quad \mathbb{E}_{X\sim\mathcal{D}} \int \left\| \frac{\mathrm{d}X_t}{\mathrm{d}t} - F_p(X_t) \right\|. \tag{E.5}$$

where the supervision is done directly over derivatives, obtained through finite-difference schemes. This echoes the proof in Section E.2 of the Appendix 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 E.5 is indeed equivalent to Eq E.4 in the continuous setting, in the practical discrete one, the way error propagates is not anymore: For Eq E.4 it is controlled over integrated trajectories while for Eq E.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 Eq E.4, if F_a 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 Eq E.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 $X \in \mathcal{D}$:

$$\forall t,\ \frac{\mathrm{d}(X - \widetilde{X})_t}{\mathrm{d}t} = F(X_t) - F(\widetilde{X}_t) - \epsilon(\widetilde{X}_t).$$

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.

E.5 Implementation details

We describe here the three use cases studied in the paper for validating APHYNITY. All experiments are implemented with PyTorch and the differentiable ODE solvers with the adjoint method implemented in `torchdiffeq`.[2]

E.5.1 Damped pendulum

We consider the non-linear damped pendulum problem, governed by the ODE

$$\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2} + \omega_0^2 \sin\theta + \alpha\frac{\mathrm{d}\theta}{\mathrm{d}t} = 0,$$

where $\theta(t)$ is the angle, $\omega_0 = \frac{2\pi}{T_0}$ is the proper pulsation (T_0 being the period) and α is the damping coefficient. With the state $X = (\theta, \frac{\mathrm{d}\theta}{\mathrm{d}t})$, the ODE can be written as $\frac{\mathrm{d}X_t}{\mathrm{d}t} = F(X_t)$ with $F : X \mapsto (\frac{\mathrm{d}\theta}{\mathrm{d}t}, -\omega_0^2 \sin\theta - \alpha\frac{\mathrm{d}\theta}{\mathrm{d}t})$.

[2] `https://github.com/rtqichen/torchdiffeq`

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 ($T_0 = 12, \alpha = 0.2$) and varying initial conditions. The simulation integrator is Dormand-Prince Runge-Kutta method of order (4)5 (DOPRI5, [66]). 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 [96].

Neural network architectures We detail in Table E.1 the neural architectures used for the damped pendulum experiments. All data-driven augmentations 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 [96, 246] 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: $F(X_t) = \left(\frac{\partial \mathcal{H}}{\partial(\mathrm{d}\theta/\mathrm{d}t)}, -\frac{\partial \mathcal{H}}{\mathrm{d}\theta}\right)$.

Table E.1: Neural network architectures for the damped pendulum experiments. n/a corresponds to non-applicable cases.

Method	Physical model	Data-driven model
Neural ODE	n/a	MLP(in=2, units=200, layers=3, out=2)
Hamiltonian	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, α)	2 trainable parameters ω_0, λ	n/a
APHYNITY Param ODE (ω_0, α)	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 E.2. All models were trained with a maximum number of 5000 steps with early stopping.

Table E.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

E.5.2 Reaction-diffusion equations

The system is driven by a FitzHugh-Nagumo type PDE [121]

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

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 circular padding around the borders. Δ is systematically estimated with a 3×3 discrete Laplace operator.

Dataset Starting from a randomly sampled initial state $X_{\text{init}} \in [0, 1]^{2\times32\times32}$, 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 Appendix E.7.1.

Neural network architectures Our F_a here is a 3-layer convolution network (ConvNet). 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 E.3. The estimated physical parameters θ_p in F_p are simply a trainable vector $(a, b) \in \mathbb{R}^2_+$ or $(a, b, k) \in \mathbb{R}^3_+$.

Table E.3: ConvNet architecture in reaction-diffusion and wave equation experiments, used as data-driven derivative operator in APHYNITY and Neural ODE [33].

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

Optimization hyperparameters We choose to apply the same hyperparameters for all the reaction-diffusion experiments: $Niter = 1, \lambda_0 = 1, \tau_1 = 1 \times 10^{-3}, \tau_2 = 1 \times 10^3$.

E.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:

$$w(x, y, t = 0) = C \times \exp^{\frac{(x-x_0)^2 + (y-y_0)^2}{\sigma^2}}. \tag{E.6}$$

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 $X_{t_0} = (w, \frac{\mathrm{d}w}{\mathrm{d}t})(t_0)$ and predicts all states from $t_0 + \delta t$ up to $t_0 + 25\delta t$.

Neural network architectures The neural network for F_a is a 3-layer convolution neural network with the same architecture as in Table E.3. For F_p, 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 E.3.

Optimization hyperparameters We use the same hyperparameters for the experiments:
$Niter = 3, \lambda_0 = 1, \tau_1 = 1 \times 10^{-4}, \tau_2 = 1 \times 10^2$.

E.6 Ablation study

We conduct ablation studies to show the effectiveness of APHYNITY's adaptive optimization and trajectory-based learning scheme.

E.6.1 Ablation to vanilla ML/MB cooperation

In Table E.4, we consider the ablation case with the vanilla augmentation scheme found in [139, 264, 168], 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

Table E.4: Ablation study comparing APHYNITY to the vanilla augmentation scheme [264, 168] for the reaction-diffusion equation, wave equation and damped pendulum.

Dataset	Method	log MSE	%Err Param.	$\|F_a\|^2$
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
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

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.

E.6.2 Detailed ablation study

We conduct also two other ablations in Table E.5:

- *derivative supervision*: in which $F_p + F_a$ is trained with supervision over approximated derivatives on ground truth trajectory, as performed in [96, 53]. More precisely, APHYNITY's $\mathcal{L}_{\text{traj}}$ is here replaced with $\mathcal{L}_{\text{deriv}} = \|\frac{\mathrm{d}X_t}{\mathrm{d}t} - F(X_t)\|$ as in Eq. (E.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 $\|F_a\|$ without the adaptive optimization of λ shown in Algorithm 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

Table E.5: Detailed ablation study on supervision and optimization for the reaction-diffusion equation, wave equation and damped pendulum.

Dataset	Method	log MSE	%Err Param.	$\|F_a\|^2$
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
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

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.

E.7 Additional experiments

E.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 E.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.

In Table E.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 E.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 [33]	-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

E.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 E.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 E.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 E.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	$\|F_a\|^2$
Data-driven	Neural ODE [33]	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

Vincent LE GUEN

le cnam

Deep learning for spatio-temporal forecasting - application to solar energy.

Résumé : Cette thèse aborde le problème de la prédiction spatio-temporelle par apprentissage profond, motivée par la prévision à court-terme de la production photovoltaïque à Electricité de France (EDF). Nous explorons dans cette thèse deux principales directions de recherche. La première concerne le choix de la fonction de perte pour entraîner les modèles: nous proposons d'utiliser des critères de forme et de décalage temporel sur les trajectoires prédites. Nous introduisons la fonction de perte DILATE pour la prévision déterministe et le modèle STRIPE pour la prévision probabiliste. Notre seconde direction de recherche est d'augmenter des modèles physiques incomplets avec des réseaux de neurones profonds. Pour la prédiction de vidéo, nous introduisons le modèle PhyDNet qui sépare une partie de dynamique physique, d'une partie résiduelle qui capture l'information complémentaire, comme la texture et les détails, nécessaire à la bonne prédiction. Nous proposons aussi un schéma d'apprentissage, appelé APHYNITY, qui assure une décomposition bien posée et unique entre des modèles physiques incomplets et des réseaux de neurones profonds, sous de faibles hypothèses.

Mots clés : apprentissage profond, apprentissage statistique, prévision spatio-temporelle, prévision photovoltaïque.

Abstract : This thesis tackles the subject of spatio-temporal forecasting with deep learning. The motivating application at Electricity de France (EDF) is short-term solar energy forecasting with fisheye images. We explore two main research directions for improving deep forecasting methods by injecting external physical knowledge. The first direction concerns the role of the training loss function. We show that differentiable shape and temporal criteria can be leveraged to improve the performances of existing models. We address both the deterministic context with the proposed DILATE loss function and the probabilistic context with the STRIPE model. Our second direction is to augment incomplete physical models with deep data-driven networks for accurate forecasting. For video prediction, we introduce the PhyDNet model that disentangles physical dynamics from residual information necessary for prediction, such as texture or details. We further propose a learning framework (APHYNITY) that ensures a principled and unique linear decomposition between physical and data-driven components under mild assumptions, leading to better forecasting performances and parameter identification.

Keywords : deep learning, machine learning, spatio-temporal forecasting, solar energy forecasting.