Modeling physical processes with deep learning : a dynamical systems approach

Emmanuel de Bézenac

SCIENCES SORBONNE UNIVERSITÉ

THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ

Spécialité **Informatique**

École Doctorale Informatique, Télécommunications et Électronique (Paris)

Modeling Physical Processes with Deep Learning: A Dynamical Systems Approach

Modélisation de Processus Physique avec de l'Apprentissage Profond

Présentée par

Emmanuel de Bézenac

Le 21-10-2021

Dirigée par

Patrick Gallinari

Pour obtenir le grade de

DOCTEUR de SORBONNE UNIVERSITÉ

devant le jury composé de :

M. Nicolas Courty	Université Bretagne Sud	Rapporteur
M. Ronan Fablet	IMT Atlantique – Lab-STICC	Rapporteur
M. Yann LeCun	Facebook AI Research – NYU	Examinateur
Mme. Paola Cinella	Sorbonne Université	Examinatrice
M. Gustau Camps-Valls	Universitat de València	Examinateur
M. Patrick Gallinari	Sorbonne Université – LIP6	Directeur de thèse

ABSTRACT

Deep Learning has emerged as a predominant tool for AI, and has already numerous applications in fields where data is abundant and access to prior knowledge is difficult. This is not necessarily the case for natural sciences, and in particular, for physical processes. Indeed, these have been the object of study since centuries: a vast amount of knowledge has been acquired, and elaborate algorithms and methods have been developped. This thesis has two main objectives. The first considers the study of the role that deep learning has to play in this vast ecosystem of knowledge, theory and tools. We will attempt to answer this general question through a concrete problem: the one of modeling complex physical processes, by leveraging deep learning methods in order to make up for lacking prior knowledge. The second objective is somewhat its converse: it focuses on how perspectives, insights and tools from the field of study of physical processes and dynamical systems can be applied in the context of deep learning, in order to gain a better understanding and develop novel algorithms.

RÉSUMÉ

L'apprentissage profond s'impose comme un outil prédominant pour l'IA, avec de nombreuses applications fructueuses pour des tâches où les données sont abondantes et l'accès aux connaissances préalables est difficile. Cependant ce n'est pas encore le cas dans le domaine des sciences naturelles, et encore moins pour l'étude des systèmes dynamiques. En effet, ceux-ci font l'objet d'études depuis des siècles, une quantité considérable de connaissances a ainsi été acquise, et des algorithmes et des méthodes ingénieux ont été développés. Cette thèse a donc deux objectifs principaux. Le premier concerne l'étude du rôle que l'apprentissage profond doit jouer dans ce vaste écosystème de connaissances, de théories et d'outils. Nous tenterons de répondre à cette question générale à travers un problème concret : la modélisation de processus physiques complexes à l'aide de l'apprentissage profond. Le deuxième objectif est en quelque sorte son contraire ; il concerne l'analyse des algorithmes d'apprentissage profond à travers le prisme des systèmes dynamiques et des processus physiques, dans le but d'acquérir une meilleure compréhension et de développer de nouveaux algorithmes pour ce domaine.

CONTENTS

ABSTRACT i

RÉSUMÉ iii

CONTENTS v

LIST OF FIGURES vii

LIST OF TABLES xvii

ACRONYMS xxi

1 INTRODUCTION 1
- 1.1 Context And Motivation . . . 1
- 1.2 Reading Guide . . . 2
- 1.3 Thesis Topics and Contributions . . . 2

2 RELATED WORKS 10
- 2.1 Learning Dynamical Systems with Deep Learning . . . 10
- 2.2 A Dynamical Systems Viewpoint on Deep Learning . . . 17

3 LEARNING DYNAMICAL SYSTEMS WITH DEEP LEARNING 23
- 3.1 Learning Dynamical Systems from Partial Observations . . . 24
- 3.2 Incorporating Partial Knowledge . . . 42
- 3.3 Incorporating Imperfect Knowledge . . . 60
- 3.4 Leveraging the Data From Different Environments . . . 75

4 A DYNAMICAL SYSTEMS VIEWPOINT ON DEEP LEARNING 92
- 4.1 Analysis of CycleGAN . . . 93
- 4.2 Analysis of Classification Networks . . . 111

5 ADDITIONAL WORK 129
- 5.1 Unsupervised Image Reconstruction . . . 130
- 5.2 A Neural Tangent Kernel Perspective of GANs . . . 146
- 5.3 Normalizing Kalman Filters for Multivariate Time Series Forecasting 162

6 CONCLUSION 178
- 6.1 Learning Dynamical Systems using Deep Learning . . . 178
- 6.2 A Dynamical Systems Viewpoint on Deep Learning . . . 180

BIBLIOGRAPHY 181

A APPENDIX CHAPTER 3 214
- A.1 Appendix 3.1: Learning Dynamical Systems from Partial Observations 215
- A.2 Appendix 3.3: Incorporating Imperfect Knowledge . . . 223
- A.3 Appendix 3.4: Leveraging the Data from Different Environments . . . 237

B APPENDIX CHAPTER 4 257
- B.1 Appendix 4.2: Analysis of Classification Networks . . . 258

C APPENDIX CHAPTER 5 271

C.1 Appendix 5.1: Unsupervised Image Reconstruction 272
C.2 Appendix 5.2: A Neural Tangent Kernel Perspective of GANs . . . 282
C.3 Appendix 5.3: Normalizing Kalman Filters for Multivariate Time Series Forecasting 317

LIST OF FIGURES

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: RELATED WORKS 10

Figure S1 The Figure illustrates successive steps of the dynamic transportation of α to β together with the notations used in the text. Each step could for example correspond to a transformation performed by an elementary module of a ResNet. . 18

CHAPTER 3: LEARNING DYNAMICAL SYSTEMS WITH DEEP LEARNING 23

Figure S1 Setting 1 Model 31

Figure S2 Setting 2 Model 31

Figure S3 Forecasting the Navier Stokes equations 10 time steps ahead with different models, starting from a given initial condition. 34

Figure S4 Forecasting the Navier Stokes equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time stted. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequences is represented as 4 consecutive rows. 36

Figure S5 Forecasting the Navier Stokes equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time stted. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequences is represented as 4 consecutive rows. 37

Figure S6 Forecasting the shallow water equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time steps. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequence is represented as 4 consecutive rows. 39

Figure S7 Forecasting the shallow water equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time steps. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequence is represented as 4 consecutive rows. 40
Figure S8 Time interpolations with our approach on the test set. We train our model by regressing to the targets every 3 images (materialized by the red boxes). We then compare the outputs of the model with the unseen ground truth states. 40
Figure S9 Estimation Model 46
Figure S10 Warping Scheme 47
Figure S11 SST Dataset 50
Figure S12 Model Architecture 51
Figure S13 SST Results 53
Figure S14 Flow Direction 55
Figure S15 Evaluation of our model's accuracy in time on data from 2006 to 2010 using data from 2011 to 2017 for training. Regions 17 to 20 were used for both periods. Each day, we produce daily forecasts for 6 days ahead and calculate the associated mean square error. The color of the flow represents the direction of the direction each of the vector as illustrated in Figure S14. 56
Figure S18 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 (T. Q. Chen et al. 2018c) 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). 63
Figure S19 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 ($t = 2.5$). 72
Figure S20 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. 73

Figure S21 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 74

Figure S22 Test error compared with corresponding theoretical bound. The arrows indicate the changes after applying $\Omega(g_e)$ penalty. 83

Figure S23 Left: final states for GS and NS predicted by the two best baselines (*One-Per-Env.* and FT-NODE) and *LEADS* compared with ground truth. Different environment are arranged by row (3 in total). Right: the corresponding MSE error maps; darker is smaller. (See Sup. A.3.4 for full sequences) 86

Figure S24 Test prediction obtained with two baselines (*One-Per-Env.* and FT-ODE) and *LEADS* compared with ground truth for LV in phase space for 4 envs., one per figure from left to right. 86

Figure S25 Test error for LV w.r.t. the number of envs. We apply the models in 1 to 8 envs. 4 groups of curves correspond to models trained with 1 to 8 trajectories per env. All groups highlight the same tendencies: increasing *One-For-All*, stable *One-Per-Env.*, and decreasing *LEADS*. 88

Figure S26 Test error evolution during training on 2 novel environments for LV. 89

CHAPTER 4: A DYNAMICAL SYSTEMS VIEWPOINT ON DEEP LEARNING 92

Figure S1 Pairings between domains obtained with CycleGAN. Both domains correspond to uniform distributions on a 2d-sphere with shifted centers. Small initialization values lead to simple and ordered mappings (Left), whereas larger ones yield complex and disordered ones (Right). Colors highlight original pairing between domains, before shifting. 97

Figure S2 Left: L^2 distance to the Optimal Transport mapping "*Wasserstein 2 Transport*" as a function of the initialization gain (domains are illustrated in Figure S1). "Ours" refers to the model presented subsequently. Right: Transport cost of the CycleGAN mapping as a function of initialization gain. Metrics are averaged across 5 runs, and the standard deviation is plotted. 98

Figure S3 Visualization of the hidden layers of CycleGAN when mapping the yellow gaussian distribution to the green one with different initializations: As shown by the colored points representing samples under the histograms, when σ increases, the mapping goes from a simple translation (Top) to a more complicated mapping (Bottom), thus inducing an increase in transport cost. . . . 103

Figure S4 Male to Female translation (top) and the inverse (bottom). Intermediate images are the interpolations provided by the network's intermediate layers. The reverse mapping is not trained. Additional samples are available in the Appendix, B.1.4. 106

Figure S5 Each column associates one input image to its outputs for different models: CycleGAN and our model with different initial gain parameters. We have ensured convergence of all models to the same fit to the target distribution. 107

Figure S6 Results for Imbalanced CelebA Task. We wish to map faces that have the **Non-Smiling** and **Black Hair** attributes to **Smiling**, **Black-Hair** faces, while only accessing **Smiling**, **Blond Hair** faces for the target domain. 108

Figure S7 Transformed circles test set by a ResNet9 after blocks 1, 5 and 9 after training; first row with good initialization; second row with a $\mathcal{N}(0, 5)$ initialization; third row with a $\mathcal{N}(0, 5)$ initialization and the transport cost added to the loss 119

Figure S8 Test transport against test accuracy of ResNet9 models on MNIST (left) and CIFAR10 (right) with fitted linear regressions, where each color indicates a different initialization (either orthogonal or normal with varying gains) 120

Chapter 5: additional work 129

Figure S1 The Figure illustrates the dependencies between the variables considered for handling the likelihood term when solving (S4). The likelihood term in (S4) can be replaced by the expectation of $\frac{1}{2\sigma^2}\|y - F(G(y);\theta)\|_2^2$ (see Equation S7). To compute this expectation, one first simulates an observation y from a signal x using $F(x;\theta)$, then generates $\tilde{x} = G(y)$ and $\tilde{y} = F(\tilde{x};\theta)$, as in the Figure. This allows us to compute the MSE term in the above expression. 134

Figure S2 The figure illustrates the dependencies of the variables used for dealing with the prior term in (S4). An observation y is sampled, and then transformed by the generative network into a reconstructed signal $\hat{x} = G(y)$. One then simulates a measurement $\hat{y} := F(\hat{x}; \theta)$ from this reconstruction. We then enforce the distributions of observations p_Y and simulated measurements p_Y^G to be similar using an adversarial loss. In order to produce indistinguishable distributions, the generator G has to *remove* the corruption and recover a sample $\hat{x}$ from p_X. 135

Figure S3 General Approach. We wish to train G to recover a plausible signal from lossy measurements. As is shown in Section 5.1.2.2, this requires the reconstructions $\hat{x} := G(y)$ to have high probability under the likelihood and the prior. For simplicity, variable $\mathcal{E}$ has been omitted. *Prior :* we sample a measurement y from the data, produce a reconstruction $\hat{x}$, and sample a perturbation parameter θ. We enforce the simulated measurement $\hat{y} := F(\hat{x}; \theta)$ to be similar to measurements in the data using an adversarial penalty. Intuitively, this requires the network to *remove* the corruption. *Likelihood :* to enforce G to produce reconstructions with high likelihood, it is not possible to add a penalty to constrain the mean square error (MSE) between y and $\hat{y}$ to be small. This is because the underlying perturbation that caused y is unknown, and may be different from θ. Starting from $\hat{y}$ we generate a $\tilde{y}$ (see figure S3) using the same θ as the one used for generating $\hat{y}$. We then constrain $\|\hat{y} - \tilde{y}\|_2^2$ to be small ($\tilde{y} = F(G(\hat{y}))$. 137

Figure S4 Model reconstructions for different corruption processes, on CelebA. Each row corresponds to a specific corruption process, and each column to a particular model. 142

Figure S5 On the top row, randomly sampled test set measurements from CelebA corrupted using Patch-Band($h = 20$), and below, our associated reconstructions. 143

Figure S6 On the top row, randomly sampled test set measurements from CelebA corrupted using Remove-Pixel($p = 0.95$), and below, our associated reconstructions. 143

Figure S7 On the top row, randomly sampled test set measurements from LSUN corrupted using Patch-Band($h = 20$), and below, our associated reconstructions. 143

Figure S8 On the top row, randomly sampled test set measurements from Recipe-1M corrupted using Remove-Pixel ($p = 0.9$), and below, our associated reconstructions. 143
Figure S9 Values of $c_{f^\star}$ for LSGAN and IPM, where $f^\star$ is a 3-layer ReLU MLP with bias and varying width trained on the dataset represented by ▼ (fake) and ▲ (real) markers, initialized at $f_0 = 0$. The infinite-width network is trained for a time $\tau = 1$ and the finite-width networks using 10 gradient descent steps with learning rate $\varepsilon = 0.1$, to make training times correspond. The gradients $\nabla_x c_{f^\star}$ are shown with white arrows on the two-dimensional plots for the fake distribution. 157
Figure S10 Generator (●) and target (×) samples for different methods. In the background, $c_{f^\star}$. 159
Figure S11 Generative model of the *NKF*. States l_t and *pseudo-observations* z_t are produced from an *LGM*, which are then transformed through a normalizing flow f_t, producing observations y_t. 166
Figure S12 Evaluation w/ and w/o NF. The axes correspond to the two components of the 2D time series; the temporal axis is marginalized out. Green points correspond to model samples when predicting 24 days ahead and blue points to future samples from the data-generating process. 172
Figure S13 Forecasting with missing data. 175

APPENDIX A: APPENDIX CHAPTER 3 214
Figure S1 Full-length prediction comparison of Fig. S23 for GS. In each figure, from top to bottom, the trajectory snapshots are output respectively from 3 training environments. The temporal resolution of each sequence is $\Delta t = 40$. 253
Figure S2 Full-length error maps of Fig. S23 for GS. In each figure, from top to bottom, the trajectory snapshots correspond to 3 training environments, one per row. The temporal resolution of each sequence is $\Delta t = 40$. 254
Figure S3 Full-length prediction comparison of Fig. S23 for NS. In each figure, from top to bottom, the trajectory snapshots correspond to 3 training environments. The temporal resolution of each sequence is $\Delta t = 1$. 255
Figure S4 Full-length error maps of Fig. S23 for NS. In each figure, from top to bottom, the trajectory snapshots correspond to from 3 training environments. The temporal resolution of each sequence is $\Delta t = 1$. 256

Appendix B: appendix chapter 4 257
Appendix B: Appendix 4.2: Analysis of Classification Networks 258
Figure S1 The Figure illustrates successive steps of the dynamic transportation of α to β together with the notations used in the text. Each step could for example correspond to a transformation performed by an elementary module of a ResNet. . 258
Figure S2 Male to Female, and Back. 262
Figure S3 Decodings of the internal representations (the outputs of the residual blocks) after training a ResNet9 on MNIST (og: original image, ae: encoding, b1: output of block 1...) . . . 265
Figure S4 Decodings of the internal representations (the outputs of the residual blocks) after training a LAP-ResNet9 on MNIST (og: original image, ae: encoding, b1: output of block 1...) . 266
Figure S5 Test accuracy during training of ResNeXt50 models on CIFAR100 268

Appendix C: appendix chapter 5 271
Appendix C: Appendix 5.1: Unsupervised Image Reconstruction 272
Figure S1 Unpaired Variant of our model. As opposed to our model, this baseline has access to samples of the signal distribution p_X. This baseline is similar to our model, however, instead of discriminating between a measurement from the data y and a simulated measurement $\hat{y}$, we directly discriminate between samples from the signal distribution and the output of the reconstruction network $\hat{x}$. 274
Figure S2 Paired Variant of our model. As opposed to our model, this baseline not only has access to samples of the signal distribution p_X, but to signal measurement pairs (y, x) from the joint distribution $p_{Y,X}$. Given input measurement y, the reconstruction is obtained by regressing y to the associated signal x. In order to avoid blurry samples, we add add a adversarial term in the objective in order to enforce G to produce realistic samples, as in Isola et al. 2016b. The model is trained using the same architectures as the ones from our model. 275
Figure S3 Baseline comparison for the CelebA dataset. Corruption is Remove-Pixel ($p = 0.95$). 276
Figure S4 Baseline comparison for the CelebA dataset. Corruption is Remove-Pixel-Channel ($p = 0.95$). 276
Figure S5 Baseline comparison for the CelebA dataset. Corruption is Convnoise($\sigma_C = 0.2$, $l = 3$). 277

Figure S6 Baseline comparison for the CelebA dataset. Corruption is Patch-Band($h = 20$). 277

Figure S7 On the top row, randomly sampled test set images from LSUN. Below, associated couples of corrupted observations and subsequent reconstructions from our model. From top to bottom, corruptions are Remove-Pixel-Channel($p = 0.95$), Remove-Pixel($p = 0.90$), Patch-Band($h = 20$), Convnoise($\sigma_C = 0.15$, $l = 3$). 278

Figure S8 On the top row, randomly sampled test set images from Recipe. Below, associated couples of corrupted observations and subsequent reconstructions from our model. From top to bottom, corruptions are Remove-Pixel-Channel($p = 0.95$), Remove-Pixel($p = 0.90$), Patch-Band($h = 20$), Convnoise($\sigma_C = 0.15$, $l = 3$). 279

Figure S9 Additional samples from our model of the LSUN Bedrooms dataset. From top to bottom, corruptions are Convnoise($\sigma_C = 0.15$, $l = 3$), Patch-Band($h = 20$), Remove-Pixel-Channel($p = 0.90$) and Remove-Pixel($p = 0.95$). 280

Figure S10 Additional sample from our model, on the Recipe dataset. From top to bottom, corruptions are Convnoise($\sigma_C = 0.3$, $l = 5$), Patch-Band($h = 20$), Remove-Pixel-Channel($p = 0.90$) and Remove-Pixel($p = 0.90$). 281

APPENDIX C: Appendix 5.2: A Neural Tangent Kernel Perspective of GANs 282

Figure S11 Generator (●) and target (×) samples for different methods applied to the Density problem. In the background, $c_{f^\star}$. . 308

Figure S12 Initial generator (●) and target (×) samples for the AB problem. 308

Figure S13 Uncurated samples from the results of the descent of a set of 1024 particules over a subset of 1024 elements of MNIST, starting from a standard Gaussian. Training is done using the IPM loss in the infinite-width kernel setting. 310

Figure S14 Gradient field $\nabla c_{f^\star_{\hat{\alpha}_x}}(x)$ received by a generated sample $x \in \mathcal{R}^2$ (i.e. $\hat{\alpha} = \hat{\alpha}_x = \delta_x$) initialized to x_0 with respect its coordinates in $\mathrm{Span}\{x_0, y\}$ where y, marked by a $\times$, is the target distribution (i.e. $\hat{\beta} = \delta_y$), with $\|y\| = 1$. Arrows correspond to the movement of x in $\mathrm{Span}\{x_0, y\}$ following $\nabla c_{f^\star_{\hat{\alpha}_x}}(x)$, for different losses and networks; scales are specific for each pair of loss and network. The ideal case is the convergence of x along this gradient field towards the target y. Note that in the chosen orthonormal coordinate system, without loss of generality, y has coordinate $(1, 0)$; moreover, the gradient field is symmetrical with respect to the horizontal axis. 311

Figure S15 Same plot as Figure S14 but with underlying points $x, y \in \mathcal{R}^{512}$. 312

APPENDIX C: Appendix 5.3: Normalizing Kalman Filters for Multivariate Time Series Forecasting 317

Figure S16 Forecast results with NF (*NKF* model) for both time-series of dataset S1 (left) and S2 (right). For each forecast: In blue, the values of the input (left of red line) and target (right of red line) time series. In dark green, the forecasts mean, and quantiles $[0.1, 0.9]$ are filled light green. 326

LIST OF TABLES

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: RELATED WORKS 10

CHAPTER 3: LEARNING DYNAMICAL SYSTEMS WITH DEEP LEARNING 23

Table S1 Relative MSE and cosine similarity scores for our models and different baselines, at different temporal horizons on the Navier Stokes equations. As the PredRNN does not explicitly model the hidden state, we replace the cosine similarity scores for this baseline with XX. 35

Table S2 Relative MSE and cosine similarity scores for our model, at different temporal horizons on the Shallow Water equations 38

Table S3 Ablation study for our model, at different temporal horizons on the Navier Stokes equations 41

Table S4 Average score and average time on test data. Average score is calculated using the *mean square error* metric (MSE), time is in seconds. 54

Table S5 Evaluation of our model's spatial generalization ability. We train our model on two distinct regions and calculate the MSE on both regions for each trained model. 56

Table S6 Forecasting and identification results on the (a) reaction-diffusion, (b) wave equation, and (c) damped pendulum datasets. We set for (a) $a = 1 \times 10^{-3}, b = 5 \times 10^{-3}, k = 5 \times 10^{-3}$, for (b) $c = 330$, $k = 50$ and for (c) $T_0 = 6$, $\alpha = 0.2$ as true parameters. $\log$ MSEs are computed respectively over 25, 25, and 40 predicted time-steps. %Err param. averages the results when several physical parameters are present. For each level of incorporated physical knowledge, equivalent best results according to a Student t-test are shown in bold. n/a corresponds to non-applicable cases. 70

Table S7 Results for LV, GS, and NS datasets, trained on m envs. with n data points per env. 87

CHAPTER 4: A DYNAMICAL SYSTEMS VIEWPOINT ON DEEP LEARNING 92

Table S1 Average highest test accuracy and 95% confidence interval of ResNet9 over 50 instances on MNIST with training sets of different sizes (in %) . . . 125
Table S2 Average highest test accuracy and 95% confidence interval of ResNet9 over 20 instances on CIFAR10 with training sets of different sizes (in %) . . . 126
Table S3 Average highest test accuracy and 95% confidence interval of ResNeXt50 over 10 instances on CIFAR100 with training sets of different sizes (in %) . . . 126

Chapter 5: additional work 129
Table S1 Average Mean Squared Error (MSE) on CelebA . . . 141
Table S2 Sinkhorn divergence Feydy et al. 2019, lower is better, similar to $\mathcal{W}_2$ averaged over three runs between the final generated distribution and the target dataset for the 8 Gaussians problem. . . . 161
Table S3 Comparative summary of competing approaches on various parameters. . . . 174
Table S4 CRPS-Sum-N (lower is better), averaged over 3 runs. The case $f_t = id$ is *DeepState* Rangapuram et al. 2018 and *VES* can be seen as part of ablation where normalizing flow and RNN are removed from *NKF*. . . . 174

Appendix A: appendix chapter 3 214
Table S1 Neural network architectures for the damped pendulum experiments. n/a corresponds to non-applicable cases. . . 228
Table S2 Hyperparameters of the damped pendulum experiments. . 229
Table S3 ConvNet architecture in reaction-diffusion and wave equation experiments, used as data-driven derivative operator in APHYNITY and Neural ODE T. Q. Chen et al. 2018c. . . 230
Table S4 Ablation study for the damped pendulum. . . . 232
Table S5 Ablation study for the reaction-diffusion equation. . . . 232
Table S6 Ablation study for the wave equation. . . . 233
Table S7 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. . . . 234
Table S8 Results of the dataset of reaction-diffusion with varying (a, b). $k = 5 \times 10^{-3}$ is shared across the dataset. . . . 235

Table S9 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 $k = 50$ 236
Table S10 Capacity definitions of different sets by covering number with associated metric or pseudo-metric. 239
Table S11 Details for the results of evaluation error in test in Fig. S22. 247
Table S12 Detailed results of evaluation error in test for Fig. S25. The case of $m = 1$ is ignored in the table, as three methods are reduced to model *One-Per-Env.* when applied to the groups containing 1 env. each. The vertical and horizontal arrows indicate that the table cells share the same value. 247
Table S13 Test MSE of experiments on LV ($m = 4, n = 1 \cdot K$) with different empirical norms. 248

Appendix B: appendix chapter 4 257
Appendix B: Appendix 4.2: Analysis of Classification Networks 258
Table S1 Average highest test accuracy and 95% confidence interval of ResNet9 over 20 instances on CIFAR10 with training sets of different sizes (in %) 267
Table S2 Average highest test accuracy and 95% confidence interval of ResNet9 over 10 instances on CIFAR100 with training sets of different sizes (in %) 267
Table S3 Average highest test accuracy and 95% confidence interval of ResNeXt50 over 10 instances on CIFAR100 with training sets of different sizes (in %) 267
Table S4 Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 1000 points and 1 block (in %) 269
Table S5 Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 1000 points and 100 blocks (in %) 269
Table S6 Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with a $\mathcal{N}(0, 5)$ initialization (in %) 269
Table S7 Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 50 points and 9 blocks (in %) 270

Appendix C: appendix chapter 5 271
Appendix C: Appendix 5.1: Unsupervised Image Reconstruction 272

Appendix C: Appendix 5.2: A Neural Tangent Kernel Perspective of GANs 282
Table S1 Sinkhorn divergence averaged over three runs between the final generated distribution and the target dataset for the Density problem. 307
Table S2 Sinkhorn divergence averaged over three runs between the final generated distribution and the target dataset for the AB problem. 308
Table S3 Sinkhorn divergence averaged over three runs between the final generated distribution and the target dataset for the 8 Gaussians problem. 309
Appendix C: Appendix 5.3: Normalizing Kalman Filters for Multivariate Time Series Forecasting 317
Table S4 Summary of the datasets used to test the models. Number of steps forecasted, data domain $\mathcal{D}$, frequency of observations, dimension of series N, and number of time steps T. 327
Table S5 CRPS-Sum-N (lower is better), averaged over 3 runs. 328

ACRONYMS

STN	Spatial Transformer Network
ConvLSTM	Convolutional Long-Short Term Memory
GP	Gaussian Process
DL	Deep Learning
GAN	Generative Adversarial Network
LSTM	Long-Short Term Memory
ML	Machine Learning
MSE	Mean Squared Error
PDE	Partial Differential Equation
ODE	Ordinary Differential Equation
NN	Neural Network
PKN	Prior Knowledge Network
SST	Sea Surface Temperature
BCCE	Brightness Constancy Constraint Equation

CHAPTER 1

INTRODUCTION

1.1 Context And Motivation

Deep Learning has emerged as a predominant tool for AI, and has already abundant applications in industry as well as in the scientific community (Lecun et al. 2015). This was made possible by recent advances in computer hardware and software technology allowing larger and larger networks to be trained in a feasible timeframes, and data to be acquired and stored in a large-scale fashion. The effectiveness of deep learning stems from the fact that–given enough data– they are able to learn meaningful representations from the data automatically. From these, previously inconceivable complex tasks are able to be solved, requiring less and less assistance and expert knowledge.

Despite impressive results in domains such as natural language, speech processing and recognition, computed vision, and even health sciences, etc... it is not yet clear if –and how– these results transfer to other fields such as the natural sciences and the study of physical processes and systems (Reichstein et al. 2019). These fields have been the object of a study since centuries, and have led to the developpement of a wide range of tools and technologies ranging from high resolution measurement instruments such as satellites, to mathematical frameworks and abstractions, e.g. dynamical systems, numerical schemes, simulations and carefully handcrafted models of real world systems. The question of the role that deep learning has to play in this complex eco-system of tools and frameworks still remains to be answered; it is the one of the objects of this thesis, and has been the subject of much attention in the last years. More specifically, we tackle the question of modelling physical phenomena –evolving in time and/or in space– from the data, using deep learning.

Concurrently, the inner workings and principles behind the effectiveness of deep learning algorithms still remain elusive to this day, and an appropriate mathematical framework is lacking. Recently, (E 2017) has brought to the attention that deep learning models are analoguous to dynamical systems evolving in time. This offers a new perspective on these models and allows one to leverage the vast litterature and apply the many tools that have been developped over the

years to grasp and deal with evolving physical systems. In particular, the principle of least action, central in many natural sciences e.g. relativistic mechanics, thermodynamics, quantum mechanics, etc..., offers an elegant way to characterize the system from an energetic perspective, based on the fact that nature tends to follow the path of least effort. In this thesis, we attempt to understand and yield a perspective on deep learning from the lens of the principle of least action. More specifically, we analyze the gradual displacements of the data samples induced by the successive neural network layers using the mathematical framework and tools of the field of Optimal Transport.

The thesis is focused on these two main topics: modelling dynamical systems with deep learning, and analyzing deep learning through the lens of dynamical systems.

1.2 Reading Guide

Sections 1 and 2 are devoted to introducing and discussing the related work around the core work of the thesis, centered around the two main aspects: deep learning for physics, and physics for deep learning. Sections 3 and 4 describe these two aspects; each of these is based on several publications that are introduced and may be read separately. This forms a coherent whole of approximately 120 pages. Section 5 is related to auxiliary work conducted during the thesis, which can be put in relation with the core of the work but has been set aside for reasons of clarity and readability. Reading this section is not essential in order to understand the main aspects of the work. Notations throughout the sections have been unified wherever possible, and each chapter can be read independently with notations being reintroduced at the beginning of each section. Appendices associated with some of the chapters provide additional demonstrations and experiences.

1.3 Thesis Topics and Contributions

1.3.1 Learning Dynamical Systems using Deep Learning

In this chapter, we study the modelling of physical processes using deep learning. We will assume that the evolution of the process can be written as a dynamical system, more specifically, a differential equation, whose exact form is unknown. The evolution will then be estimated from the data, once the learning problem is posed. In all the sections, the specific learning problem will vary, but often take the form of a constrained optimization problem (apart from Section 3.2), and can

be seen as an adaptation of the problem presented in the related work, which is often used in the data assimilation or optimal control community.

First, adapting this learning framework, we will see how learning can be achieved in the context where the state is not fully observed, which arises frequently as some quantities are difficult to measure. More specifically, we suppose there exists an unknown dynamical system governing the underlying process, whose state is observed through a lossy but known measurement mechanism, as is the case for many realistic applications. Taking inspiration from the theory of optimal control for dynamical systems, we formulate the learning problem as a constrained optimization problem and solve it using a gradient based approach, computing the gradient with the adjoint state method. This learning framework is generic and can be applied for a wide range of dynamical phenomena. In this same section we also see how this learning framework can be trivially adapted for processes that involve space, using convolutional networks. In this work, as well as subsequent work on modelling dynamical systems, we choose to assess the usefulness of our approach by evaluating the model's capacity to forecast the future. We report good results w.r.t. the baselines for forecasting observations for spatio-temporal processes governed by shallow water and Navier-Stokes equations.

Next, we will study the generalization capabilities of these machines; as in any machine learning algorithm, generalization is key. A central question in our work is how can we develop evolution models from data driven methods that are capable of generalizing, e.g. to unseen initial conditions, unseen times (past or future), or even unseen dynamical systems. This problem will be tackled in two different ways: (1) injecting prior knowledge (Section 3.2 and 3.3), and (2) finding ways to increase the quantity of data (Section 3.4).

For the first aspect (1) we will inject both (i) partial but accurate knowledge (Section 3.2) and (ii) imperfect knowledge (Section 3.3), and analyze the usefulness of these approaches for different complex physical processes.

Section 3.2 focuses on incorporating partial knowledge into a deep learning system, using sea surface temperature (SST) forecasting as an example application. In physics, partial knowledge usually takes the form of some set of evolution equations acting on the dynamics, or locally in space; hence these are the ones we focus on. The use case of SST is particularly interesting as the exact dynamics are very complex and hard to describe through a set of equations. Nonetheless, in this case we have partial knowledge of the behavior of the system: given an unknown velocity vector, the dynamics can be described (approximately) using an advection-diffusion equation. In this work we highlight the need for incorporating prior knowledge in deep learning systems to model natural phenomena in order for them to generalize and learn the laws of physics. We then demonstrate how

this partial knowledge can be incorporated, and exhibit a higher generalization performance w.r.t. to pure deep learning approaches on the problem of forecasting future SST.

Section 3.3 is devoted to incorporating imperfect knowledge, i.e. slightly inaccurate scientific knowledge taking the form of a physical evolution equation. This physical model is simplistic and does not explain the underlying phenomena entirely. The approach consists in decomposing the dynamics into two components: the simplistic physical model, 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 phenomena, i.e. reaction-diffusion equations, wave equations and the nonlinear damped pendulum, show that the approach can efficiently leverage approximate physical models to accurately forecast the evolution of the system and correctly identify relevant physical parameters.

Finally, in Section 3.4, we tackle the problem of generalization when prior knowledge is inaccessible. This work aims to leverage data from slightly similar dynamical systems. It is not uncommon that different physical processes bear similarities, e.g. as first principles may be the same or there may be common driving factors and mechanisms. Intuitively, disregarding the data from these processes does not exploit the data in an optimal way and is prone to scarcity problems and generalization errors. On the other hand, considering the data i.i.d. as in the classical learning setting and learning a single model to cover all situations disregards the discrepancies between environments leading to biased solutions. We propose LEADS, a novel framework that leverages the commonalities and discrepancies among known environments to improve model generalization. This is achieved with a tailored training formulation aiming at capturing common dynamics within a shared model while additional terms capture environment-specific dynamics. We ground our approach in theory, exhibiting a decrease in sample complexity with our approach and corroborate these results empirically, instantiating it for linear dynamics. Moreover, we concretize this framework for neural networks and evaluate it experimentally on representative families of nonlinear dynamics. We show that this new setting can exploit knowledge extracted from environment-dependent data and improves generalization for both known and novel environments.

The work in this section has led to the following publications:

- Ibrahim Ayed, Emmanuel de Bézenac, Arthur Pajot, and Patrick Gallinari (2020). "Learning the Spatio-Temporal Dynamics of Physical Processes from Partial Observations". In: *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, pp. 3232–3236. URL: https://doi.org/10.1109/ICASSP40776.2020.9053035

- Emmanuel de Bezenac, Arthur Pajot, and Patrick Gallinari (Feb. 2018). "Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge". In: URL: https://openreview.net/forum?id=By4HsfWAZ

- Yuan Yin, Vincent LE GUEN, Jérémie DONA, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas THOME, and patrick gallinari (2021b). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=kmG8vRXTFv

- Yuan Yin, Ibrahim Ayed, Emmanuel de Bézenac, Nicolas Baskiotis, and Patrick Gallinari (2021a). *LEADS: Learning Dynamical Systems that Generalize Across Environments*. arXiv: 2106.04546 [cs.LG]

1.3.2 A Dynamical Systems Viewpoint on Deep Learning

The previously introduced chapter treats of the use of deep learning in the context of dynamical systems and physical processes. The problem this chapter treats is actually its converse, i.e. how can one make use of the theory of dynamical systems and the multitude of tools developed in this community in the context of deep learning? Indeed, (E 2017) has brought to the attention that deep learning models, more specifically Resnets, can be linked with a dynamical system evolving in time. This dynamical system in fact corresponds to a transport equation, as it gradually displaces the input samples throughout its layers. This transportation, or forward pass, thus has an associated cost, an action. The principle of least action (or stationary action), central in many natural sciences e.g. relativistic mechanics, thermodynamics, quantum mechanics, etc... offers an elegant way to characterize the system from an energetic perspective, based on the fact that nature tends to follow the path of least effort. In this thesis, we attempt to understand and yield a perspective on deep learning from the lens of this principle of least action, and thus, the theory of optimal transport (OT).

In Section 4.1, we study deep learning models by making the connection with OT for the first time. We focus on the problem of Unsupervised Domain Translation (UDT) where the aim is to transform elements from one domain to another: for example, transforming photographs into their associated stylized paintings. The most widely used approach is CycleGAN (J. Zhu et al. 2017), which uses residual networks and generative adversarial network (GAN) type losses to achieve the desired goal. This approach works very well in practice for a wide range of tasks, but is not theoretically grounded. In fact, the optimization problem for the training phase is ill-posed and there is no reason one should expect good solutions. We find that the reason this algorithm is so effective is that the solution map is highly biased to yielding low kinetic energies. This has brought us to consider the optimal case where the transport equation yields the lowest energy possible; which is in itself an entire mathematical field, the field of Optimal Transport (OT). Reformulating the learning problem in order to make this implicit energetic bias explicit, we are able to link the problem of UDT with OT. This allows one to study the solutions mappings from a theoretical perspective making use of OT theory, e.g. we prove existence and uniqueness of the learning problem, regularity of the mappings, etc... We also propose a simple algorithm to solve this optimal transport problem with neural networks using the dynamical formulation of OT.

In Section 4.2, we use this same analogy to study deep residual models of the problem of classification. Making the low energy bias explicit makes the network and training algorithm amenable to theoretical analysis, and offers practical improvements w.r.t. classical residual models. Indeed, we improve classification results by reducing the generalization gap even in low data regimes. Moreover, we found that the training is stabilized in an adaptive fashion without being slowed down. More generally, we believe this perspective offers a promising alternative insight leading to other potential applications.

The work in this section has led to the following publications:

- Emmanuel de Bézenac, Ibrahim Ayed, and Patrick Gallinari (2019). "Optimal Unsupervised Domain Translation". In: *CoRR* abs/1906.01292. arXiv: 1906.01292. URL: http://arxiv.org/abs/1906.01292

- Skander Karkar, Ibrahim Ayed, Emmanuel de Bézenac, and Patrick Gallinari (2020). "A Principle of Least Action for the Training of Neural Networks". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part II*. ed. by Frank Hutter, Kristian Kersting, Jefrey Lijffijt, and Isabel Valera.

Vol. 12458. Lecture Notes in Computer Science. Springer, pp. 101–117. URL: https://doi.org/10.1007/978-3-030-67661-2%5C_7

1.3.3 Additional Work

Additional work on various deep learning topics have also been conducted during this thesis, available in Chapter 5. They are regrouped in three sections, introduced in the following sections.

The work in this section has led to the following publications:

- Arthur Pajot, Emmanuel de Bezenac, and Patrick Gallinari (Sept. 2018). "Unsupervised Adversarial Image Reconstruction". In: URL: ICLR%202019%20:%20https://openreview.net/forum?id=BJg4Z3RqF7
- Jean-Yves Franceschi, Emmanuel de Bézenac, Ibrahim Ayed, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari (2021). "A Neural Tangent Kernel Perspective of GANs". In: *CoRR* abs/2106.05566. arXiv: 2106.05566. URL: https://arxiv.org/abs/2106.05566
- Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski (2020). "Normalizing Kalman Filters for Multivariate Time Series Analysis". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin. URL: https://proceedings.neurips.cc/paper/2020/hash/1f47cef5e38c952f94c5d61726027439-Abstract.html

1.3.3.1 Unsupervised Image Reconstruction

The first, Section 5.1 treats the problem of reconstructing signals from incomplete and inaccurate observations and measurements without ever having access to the true signal. This is a common scenario arising in many scientific problems, e.g. recovering a dynamical system's state from measurements. A novel approach using deep learning has been proposed. It relies on the fact that we have access to samples from the measurement distribution along with the measurement function, i.e. given a sample from the true signal distribution, it can be deteriorated in such a way that it resembles measured incomplete and inaccurate measurements. The approach, in a nutshell: a network takes as input a measurement of the true signal, and is trained in such a way that its output has a high score under the

log-posterior, i.e. the probability of the true signal given the input measurement. This is achieved by decomposing the log-posterior into a prior and a likelihood term. As we have access to the measurement process, the likelihood term can be estimated from the data using a Monte-Carlo approximation. However, this is not the case for the prior term; it is difficultly estimated as we do not even have access to samples from the true signal distribution. To ensure that the samples have high probability under the prior, we a special form of adversarial training, similar to the one presented in (Bora et al. 2018a). It consists in generating a sample using a neural network, such that the deterioration caused by the measurement process cannot be distinguished from a sample from the measurement distribution, using a discriminator network trained in an adversarial fashion. Our method yields attractive results even when the data is heavily corrupted.

1.3.3.2 A Neural Tangent Kernel Perspective of GANs

The second auxiliary work, Section 5.2 is motivated by the previous work on GANs, based on the fact that these methods to this day remain poorly understood. Its goal is to develop a finer-grained theoretical understanding of the inner-workings of GANs, by letting the discriminator's width grow unbounded, entering the NTK regime (Jacot et al. 2018a). In related work studying GANs, it is common to study adversarial losses used in the GAN regime considering the discriminator is arbitrarily powerful. We show that this framework of analysis is too simplistic to properly analyze GAN training. To tackle this issue, we leverage the theory of infinite-width neural networks to model neural discriminator training for a wide range of adversarial losses via its Neural Tangent Kernel (NTK). Our analytical results show that GAN trainability primarily depends on the discriminator's architecture. We further study the discriminator for specific architectures and losses, and highlight properties providing a new understanding of GAN training. For example, we find that GANs trained with the integral probability metric loss minimize the maximum mean discrepancy with the NTK as kernel. Our conclusions demonstrate the analysis opportunities provided by the proposed framework, which paves the way for better and more principled GAN models. We release a generic GAN analysis toolkit based on our framework that supports the empirical part of our study.

1.3.3.3 Normalizing Kalman Filters for Multivariate Time Series Forecasting

Finally, in Section 5.3.2, treats of forecasting time-series in a probabilistic setting. More specifically, it tackles the modelling of large, complex and multivariate time series panels in a probabilistic setting. To this extent, we present a novel approach reconciling classical state space models with deep learning methods. By

augmenting state space models with normalizing flows, we mitigate imprecisions stemming from idealized assumptions in state space models. The resulting model is highly flexible while still retaining many of the attractive properties of state space models, e.g., uncertainty and observation errors are properly accounted for, inference is tractable, sampling is efficient and good generalization performance is observed, even in low data regimes. We demonstrate competitiveness against state-of-the-art deep learning methods on the tasks of forecasting real world data and handling varying levels of missing data.

CHAPTER 2

RELATED WORKS

Contents

2.1 Learning Dynamical Systems with Deep Learning 10
2.1.1 Dynamical Systems Background 10
2.1.2 Parameter Estimation for Dynamical Systems 12
2.1.3 Machine Learning for Dynamical Systems 14
2.2 A Dynamical Systems Viewpoint on Deep Learning 17

This section is devoted to discussing work in relation with the two main aspects of the thesis. Related work in the core sections is also made available and may be recalled for the sake of clarity.

2.1 Learning Dynamical Systems with Deep Learning

Here we examine related work of Section 3, which treats the problem of modeling physical processes evolving in time and/or in space with deep learning approaches. This will bring us to consider work from the field of dynamical systems, optimal control and parameter estimation, data assimilation and applied machine learning. This section also can be useful to get familiar with the notations used throughout Section 3, even if these will be reintroduced later for simplicity.

2.1.1 Dynamical Systems Background

The classical approach to modelling natural processes evolving in time and/or in space is first to study the system by carefully observing it to uncover the hidden mechanisms and forces that drive the system's evolution. From the latter, the goal is then to transcribe the acquired knowledge into the mathematical world, through a set of equations. Dynamical systems are a tool of choice to model the evolution of phenomena occurring in nature; they describe the evolution of the system by

modelling the evolution of a set of variables X_t describing the system at a given time t, called the state, along with a transition function $T(X_t) = X_t + \Delta t$ linking consecutive states in time. Generally, the continuous limit proves to be more tractable, powerful and convenient for calculations, so that one usually considers an evolution equation of the form:

$$\frac{dX_t}{dt} = F(X_t, t) \tag{S1}$$

Let us first consider the case where the state is a d-dimensional vector. In this case, we have an *ordinary differential equation* (ODE). Together with an initial condition X_0, it is known as the initial value problem (IVP), and the Picard-Lindelöf theorem guarantees a unique solution on some time interval if F is continuous on a region containing t_0 and X_0, and satisfies the Lipschitz condition on the variable X_t. Often, the system is *autonomous*, i.e. the law governing the evolution of the system depends solely on the system's current state and not the time variable t, which may be omitted in the following work. It is usually impossible to write down the explicit formulas for solutions of the IVP, but these may be approximated using *numerical schemes*, also called ODE solvers.

The typical introductory numerical scheme is the explicit Euler method: and some chosen temporal interval Δt, an approximate solution of Equation S1 can be computed iteratively, starting from some initial condition X_0: $X_{(k+1)\Delta t} = X_{k\Delta t} + \Delta t F(X_{k\Delta t}, k\Delta t)$. A plethora of numerical methods exist, the choice of the method usually depending on the problem at hand. However, different criteria for a given method exist, such as numerical stability, convergence, consistency and order, etc... refer to (Hamming 1973) for a general introduction. For instance, the Euler is consistent, and is of order $p = 1$, meaning that the difference between the result given by the method and the solution is $O(\Delta t^{p+1})$ as $\Delta t \to 0$. Furthermore, it may exhibit instable behavior as it is unconditionally unstable, and for this reason, one usually resorts to using higher order methods such as Runge-Kutta 4.

It is also possible to include the dependance on other variables such as space. We consider spatial coordinates $x \in \Omega \subset \mathcal{R}^{d_s}$, considered compact and bounded. In this case, the state is now both a function of time and space, and Equation S1 becomes a *partial differential equation* (PDE), as it usually involves differential operators w.r.t. the spatial variable x. These equations are ubiquitous in the natural sciences, as they are the basis for modelling. In order to clearly define the problem, one usually has to specify boundary conditions, i.e. sets of constraints characterizing the behavior of the system on the boundaries of Ω. All these equations together constitude the boundary value problem (BVP).

As is the case for ODEs, solutions can rarely be obtained analytically but numerical methods for PDEs also exist. The three most widely used are the finite

element method (FEM), finite volume methods (FVM) and finite difference methods (FDM). For example, the FDM consists in approximating the differential terms as difference quotients. All these methods require discretizing the space on a finite number of points called a *mesh*. These are usually irregular as more points are required where the solution and/or its derivatives change rapidly. A necessary condition for the convergence of the explicit time integration methods is given by the Courant-Friedrichs-Lewy (CFL) condition, involving the time-step Δt and length interval Δx. Note that mesh-free methods also exist, most of which are based on the Galerkin method.

2.1.2 Parameter Estimation for Dynamical Systems

In the following work, we assume that the physical process can be accurately modelled by a differential equation of the form S1, however its exact form is unknown (at least partially). The goal will be to leverage the data to help characterise the system's dynamics. This problem has been previously tackled in the data assimilation and optimal control community in the context where the dynamics are largely known, up to some unknown parameters which can be estimated from the data, by solving an constrained optimization problem. Consider a temporal sequence of data states X, and a parameterized model F_θ generating the state X^θ. Estimating parameters θ from the data can then be done by considering the following constrained optimization problem:

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \mathcal{J}(X^\theta) \\ \text{subject to} \quad & \forall t, \frac{\mathrm{d}X_t^\theta}{\mathrm{d}t} = F_\theta(X_t^\theta, t), \\ & X_0^\theta = X_0 \end{aligned} \tag{S2}$$

where $\mathcal{J}$ is an energy functional, usually quantifying the discrepancy between trajectories from the data X_t and the model trajectory X^θ. In general this optimization problem is hard to resolve analytically, especially in the case where the dynamics are non-linear. However, it can be solved approximately using the following iterative procedure, assuming $\mathcal{J}$ is differentiable:

1. Find X^θ solving the constraints solving the IVP using a DE solver,
2. Compute the gradient $\frac{\partial \mathcal{J}}{\partial \theta}(X^\theta)$ using the *adjoint state method*,
3. Apply gradient based optimization method step, and reiterate until convergence.

This technique has been used in (DIMET et al. 1986) to estimate the initial condition in the context of numerical weather prediction, and has been the basis

for a number of algorithms for physical parameter estimation, e.g. 4d-Var (Carrassi et al. 2018), (Bérèziat et al. 2015).

Computing the gradient of $\mathcal{J}$ w.r.t. the parameters θ usually requires calculating $\frac{\partial X^\theta}{\partial \theta}$ which is often very computationally demanding as it requires solving $\dim(\theta)$ forward equations. Instead, the Adjoint state method (Gunzburger 2002) is an elegant method that circumvents this issue by considering the Lagrangian formulation of the constrained optimization problem. As example, consider the case where $J(X^\theta) = \int_0^T \left\| X_t^\theta - X_t \right\| \mathrm{d}t$. In this case, the Lagrangian writes out as:

$$\mathcal{L}(X^\theta, \lambda, \mu, \theta) = J(X^\theta) + \int_0^T \left\langle \lambda_t, \frac{\mathrm{d}X_t^\theta}{\mathrm{d}t} - F_\theta(X_t^\theta, t) \right\rangle \mathrm{d}t + \left\langle \mu, X_0 - X_0^\theta \right\rangle \tag{S3}$$

The scalar product $\langle \cdot, \cdot \rangle$ can be freely chosen here and is problem dependent, e.g. it can for instance be the euclidean scalar product of $\mathcal{R}^d$ in the ODE case, or a scalar product of $\mathcal{L}^2(\Omega)$ in the case where the state is a function of time and space.

Any θ, X^θ satisfies the constraints by definition, which means that:

$$\forall \lambda, \mu, \ \frac{\partial}{\partial \theta} \mathcal{L}(X^\theta, \lambda, \mu, \theta) = \frac{\partial}{\partial \theta} \mathcal{J}(X^\theta) \tag{S4}$$

The gradient can then be obtained by differentiating the Lagrangian using standard calculus of variations:

$$\frac{\partial}{\partial \theta} \mathcal{J}(X^\theta) = -\int_0^T \left\langle \lambda_t, \partial_\theta F_\theta(X_t^\theta, t) \right\rangle \mathrm{d}t - \left\langle \lambda_0, \partial_\theta X_0^\theta \right\rangle \tag{S5}$$

where λ is a solution of:

$$\partial_t \lambda_t = A_t \lambda_t + B_t \tag{S6}$$

solved backwards, starting with $\lambda_T = 0$, and where:

$$A_t = -(\partial_X F_\theta(X^\theta, t))^\star \tag{S7}$$

and

$$B_t = -2(X_t - X_t^\theta) \tag{S8}$$

where $M^\star$ denotes the adjoint operator of the linear operator M.

The Equation S6, known as the *backward* equation, is linear but involves computing the sequence of states X_t^θ with a *forward* pass first.

There are essentially two different ways of solving the backward equation: the *differentiate-then-discretize* approach, and the *discretize-then-differentiate* approach [1].

In a *differentiate-then-discretize* approach, one directly approximates the forward and backward equations using numerical schemes. Here, the approximation error to the gradient comes from the discretization error made in the solver for both the forward and backward equations. This method is used in the black box solvers presented in (T. Q. Chen et al. 2018a). It has the advantage of allowing the use of non-differentiable steps in the solver, and can avoid high memory consumptions in the case where the system is reversible, as past states can be recomputed on the fly from the forward equations solved in reverse. However, this method can yield inconsistent gradients of cost functional $\mathcal{J}$, the discretization of the adjoint equations depends highly on the studied problem and therefore must be carefully selected and tuned (Carrassi et al. 2018).

In a *discretize-then-differentiate* approach, a differentiable solver for the forward equations is used, *e.g.* using an explicit Euler scheme $X^\theta_{t+\Delta t} \approx X^\theta_t + \Delta t F_\theta(X^\theta_t, t)$. Based on the solver's sequence of operations for the forward equations, the backward equations and the gradient can be directly obtained using automatic differentiation software (Paszke et al. 2017). This algorithm is actually equivalent to backpropagation (LeCun et al. 1988) which can be derived as a special case of it. As the step-size approaches zero, the forward and backward equations are recovered.

It is important to note that, while the two methods are consistent and both converge to the continuous-time backward equation, they do not always yield the same results as the two approaches proceed differently.

2.1.3 Machine Learning for Dynamical Systems

Machine Learning in the Geosciences As mentionned in the introduction, machine learning has a useful tool for geophysics modeling for the past decades. Most machine learning methodologies have been applied to geophysics and remote sensing. We will focus here on recent developments in the field. The last few years have seen an exponentially increasing number of deep learning applications to geophysics through the use of earth observation data. We then highlight a few representative applications. (Kalinicheva et al. 2019) perform change detection for satellite image time series using autoencoders. One of the first papers for extreme weather event detection is considered in (Racah et al. 2017). Convolutional LSTMs were introduced in (Shi et al. 2015b) for nowcasting. (Karpatne et al.

1. The differentiate-then-discretize method is often referred to as the *continuous adjoint method*, and the *discretize-then-differentiate* approach as the *discrete adjoint method* (Sirkes et al. 1997).

2017) is one of the first papers constraining neural networks to be consistent with physics and using prior physical knowledge for a prediction task, the application is lake temperature modeling. (Vandal et al. 2018) makes use of a super resolution convolutional neural network with multi-scale input channels for statistical downscaling of climate variables. (Fablet et al. 2018) also tackle the forecasting and assimilation of geophysical fields and consider sea surface temperature as an application. Refer to (Reichstein et al. 2019) for a comprehensive survey.

Related to our work, there is the field of data assimilation (Lorenc 1986; Carrassi et al. 2018), where one is interested in using (partial) observations, in conjunction with the evolution model, supposed known, in order to retrieve the canonical state. Typically, our constrained optimisation problem is similar to the one posed in classical 4D-Var (Carrassi et al. 2018), where the constraint is the evolution equation of the state. Although there have been work in data assimilation community where the evolution equation is only partially known and some unknown forcing terms are estimated from the data (Bérèziat et al. 2015), our work takes a more data-driven approach, where we make no assumptions and use no prior knowledge of the underlying evolution equation.

Differential Equations Discovery In the past, several works have already attempted to learn differential equations from data, such as e.g., (Crutchfield et al. 1987; Alvarez et al. 2013). More recently, (Rudy et al. 2017) uses sparse regression on a dictionary of different terms to recover the underlying Partial Differential Equation (PDE). In (Raissi et al. 2017), they propose recovering the coefficients of the differential terms by deriving a Gaussian Process (GP) kernel from a linearized form of the PDE. In (R. Chen et al. 2018), a 1d non-linear system is learned with neural networks, estimating the temporal evolution term in an ODE from the data. (Z. Long et al. 2018b) carefully tailor the neural network architecture, based on the discretization of the different terms of the underlying PDE. (Raissi 2018) develops a Neural Network (NN) framework for learning PDEs from data. (Fablet et al. 2017) constructs a bilinear network and uses an architecture similar to finite difference schemes to learn fully observed dynamic systems. In those approaches, we often see that either the form of the PDE or the variable dependency are supposed to be known and that the context is the unrealistic setting where the state is fully observed.

Correction in data assimilation Prediction under approximate physical models has been tackled by traditional statistical calibration techniques, which often rely on Bayesian methods (Pernot et al. 2017). Data assimilation techniques, *e.g.* the Kalman filter (Kalman 1960; Becker et al. 2019), 4D-var (Courtier et al. 1994), prediction errors are modeled probabilistically and a correction using observed

data is applied after each prediction step. Similar residual correction procedures are commonly used in robotics and optimal control (W.-H. Chen 2004; S. Li et al. 2014). 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 (Janner et al. 2019) or by model predictive control (Nagabandi et al. 2018). The originality of our approach is to leverage model-based prior knowledge by augmenting it with neurally parametrized dynamics. It does so while ensuring optimal cooperation between the prior model and the augmentation.

Interplay between Machine Learning and Dynamical Systems Combining physical models with machine learning (*gray-box or hybrid* modeling) was first explored from the 1990's: (Psichogios et al. 1992; Thompson et al. 1994; Rico-Martinez et al. 1994) 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 (Z. Long et al. 2018c), as discussed in the introduction. Recently, some approaches have specifically targeted the MB/ML cooperation. HybridNet (Y. Long et al. 2018) and PhICNet (Saha et al. 2020) 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. (Q. Wang et al. 2019; Mehta et al. 2020) 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 our proposed learning scheme in terms of forecasting and parameter identification performances (see experiments in Section 3.3.4.2).

Leveraging Data from Different Environments Recent approaches linking invariances to Out-of-Distribution (OoD) Generalization, such as (Arjovsky et al. 2020; Krueger et al. 2020; Teney et al. 2020), aim at finding a single classifier that predicts well invariantly across environments with power of extrapolating outside the known distributions. However, in our dynamical systems context, the optimal regression function should be different in each environment, and modeling environment bias is as important as modeling the invariant information, as both are indispensable for prediction. Thus such invariant learners are incompatible

with our setting. Meta-learning methods have recently been considered for dynamical systems as in (Finn et al. 2017; S. Lee et al. 2021). Their objective is to train a single model that can be quickly adapted to a novel environment with a few data-points in limited training steps. However, in general they result in a single model and there is no special focus on the generalization in training environments, which is the core of our setting. Multi-task learning (Yu Zhang et al. 2017) seek for learning shared representations of inputs that exploit the domain information. Current multi-task methods are not well motivated for dynamical systems due to the lack of consideration of their specificity. (Spieckermann et al. 2015) try to directly apply the existing multi-task learning ideas on interactive physical environments. Nonetheless, the approach disregards the specificity of the dynamical systems, and does not guarantee that the common dynamics are totally exploited from data. Other approaches like (Yıldız et al. 2019; Norcliffe et al. 2021) integrate some probabilistic methods into a Neural ODE, to learn a distribution of the underlying physical processes. Their focus is principally on the uncertainty of a single model learned with observations from the same dynamics, rather than learning for different ones.

2.2 A Dynamical Systems Viewpoint on Deep Learning

In this section we discuss the related work around Section 4, which focuses on developing deep learning methods, insights and algorithms by making analogies with physics and dynamical systems. More specifically, we relate the displacement of the neural network's inputs along its layers as a dynamical system, and consider an associated least action principle, which leads us to consider and make use of the field of Optimal Transport (OT). This section may also be helpful to get familiar with the notations used throughout Section 4, even if these will be reintroduced later for simplicity.

The principle of least action is central to many fields in physics, mathematics and economics. It is found in classical and relativistic mechanics, thermodynamics, quantum mechanics (Feynman 2005; Garcia-Morales et al. 2008; Gray 2009), etc.. It broadly states that the dynamical trajectory of a system between an initial and final configuration is one that makes a certain action associated with the system locally stationary (Gray 2009). One mathematical theory which can be associated with this general idea is the theory of Optimal Transport which was initially introduced as a way of finding a transportation map minimizing the cost of displacing mass from one configuration to another (Santambrogio 2015).

Formally, let α and β be absolutely continuous distributions compactly supported in $\mathcal{R}^d$, and $c : \mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}$ a cost function. Consider a transportation map $T : \mathcal{R}^d \to \mathcal{R}^d$ that satisfies $T_\sharp \alpha = \beta$, *i.e.* that pushes[2] α to β. The total cost of the transportation then depends on all the individual contributions of costs of transporting (infinitesimal) mass from each point x to $T(x)$, and finding the optimal transportation map amounts to solving:

$$\begin{aligned} \underset{T}{\text{minimize}} \quad & \mathcal{C}^{\text{stat}}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{subject to} \quad & T_\sharp \alpha = \beta \end{aligned} \tag{S9}$$

A standard choice for c is the p-th power of a norm of $\mathcal{R}^d$, *i.e.* $c(x, y) = \|x - y\|^p$, but other costs can be used, defining different variants of the problem. This cost induces, through the p-th root of the minimal value of eq. S9, a distance W_p between any two distributions α and β of finite $p-$th moment, called the p-Wasserstein distance (Peyre et al. 2018).

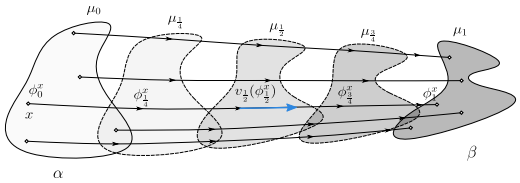


Figure S1. – The Figure illustrates successive steps of the dynamic transportation of α to β together with the notations used in the text. Each step could for example correspond to a transformation performed by an elementary module of a ResNet.

Instead of directly pushing α to β in $\mathcal{R}^d$, it is possible to view α and β as points in a space of measures, and consider trajectories from α to β in this abstract space. Thus, a way to transport the probability mass from α to β is a curve between two points in this space. The curve corresponding to the optimal mapping is then the *shortest* one, in other words it is the *geodesic curve* between the two points.

More formally, let us introduce the *Wasserstein metric space* $\mathbb{W}_p(\mathcal{R}^d)$, *i.e.* the space of absolutely continuous measures of $\mathcal{R}^d$ with finite p-th moment endowed with the Wasserstein distance:

$$W_p(\mu, \nu) = \min_{T_\sharp \mu = \nu} \mathcal{C}(T)^{\frac{1}{p}}$$

when costs of the form $c(x, y) = \|x - y\|_p^p$ are considered, for some integer $p \geq 2$. As $\mathbb{W}_p(\mathcal{R}^d)$ is a space of measures, α and β are seen as points of this space of measures, and thus, any continuous path linking both distributions defines a

2. $T_\sharp \alpha$ is the *push-forward measure*: $T_\sharp \alpha(B) = \alpha(T^{-1}(B))$ for any measurable set B.

gradual transformation from α to β and a mapping transporting α to β: this is the basis of the dynamical formulation of OT.

The following result (from Theorem 5.27 of (Santambrogio 2015)) theoretically motivates the dynamical formulation of OT:

Proposition S1. *$\mathbb{W}_p$ is a geodesic space, meaning that, for any measures $\mu, \nu \in \mathbb{W}_p$, there exists a geodesic curve $(\mu_t)_{t\in[0,1]}$ between μ and ν.*

Thus, according to this result, finding the optimal mapping between two distributions amounts to finding a curve of minimal length in a certain abstract measure space. However, it still does not provide much in the way of a practically useful algorithm. The following theorem makes a formal link with fluid dynamics and basically states that moving probability masses from one distribution to another is the same as moving fluid densities from one configuration to another under a certain velocity field (Santambrogio 2015):

Theorem S1. *Given α and β absolutely continuous w.r.t. the Lebesgue measure and $(\mu_t)_{t\in[0,1]}$ the geodesic curve with $\mu_0 = \alpha$ and $\mu_1 = \beta$, we can associate a vector field $v_t \in L^p(\mu_t)$ that solves the continuity equation*[3]*:*

$$\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$$

with:

$$W_p^p(\alpha, \beta) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{d}t$$

In other words, the geodesic curve $(\mu_t)_{t\in[0,1]}$ between both distributions, together with the minimal energy velocity vector field v solve the continuity equation. Moreover, its energy along this path is precisely equal to the Wasserstein distance $W_p^p(\alpha, \beta)$. If this vector field of minimal energy v could be obtained, probability mass could be displaced according to the flow defined by the continuity equation, and the geodesic curve could be retrieved. Thus, we can reformulate the problem as a problem of optimal control, where v is the control variate:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \mathcal{C}^{\text{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{d}t \\ \text{subject to} \quad & \partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0, \mu_0 = \alpha, \mu_1 = \beta \end{aligned} \tag{S10}$$

It is worth noting that this approach not only gives a mapping between the two distributions but it also gives the entire geodesic curve so that smooth interpolations in $\mathbb{W}_p(\mathcal{R}^d)$ can be recovered.

3. ∂_t is the partial derivative operator *w.r.t.* variable t, and $\nabla\cdot$ the divergence operator *w.r.t.* space.

In this section, we recall some classical and more recent results of regularity for Optimal Transport mappings. This is an intricate subject and the problem had been open for some time after OT theory had been established. The most important results have been established through the study of the Monge-Ampère equation by Caffarelli then De Philippis and Figalli. Extensions for larger families of costs were developed by Ma, Trudinger and Wang (X. Ma et al. 2005a) but this is out of the scope of this work.

In particular, Theorem 6.27 of (Ambrosio et al. 2005) gives a classical almost-everywhere regularity result:

Theorem S2. *If $c(x, y) = \|x - y\|^p$ for $p > 1$, and α and β have compact supports with $d(supp(\alpha), supp(\beta)) > 0$, then the Optimal Transport map T between α and β is $\alpha - a.e.$ differentiable and its Jacobian $\nabla T(x)$ has non-negative eigenvalues $\alpha - a.s.$*

More recently, results summarized below, which correspond to Theorems 4.23, 4.24 and Remark 4.25 of (A. Figalli 2017), state that the OT map has one degree of regularity more than the initial transported density:

Theorem S3. *Suppose there are X, Y, bounded open sets, such that the densities of α and β are null in their respective complements and bounded away from zero and infinity over them respectively.*

Then, if Y is convex, there exists $\eta > 0$ such that the OT map T between α and β is $C^{0,\eta}$ over X.

If Y isn't convex, there exists two relatively closed sets A, B in X, Y respectively such that $T \in C^{0,\eta}(X \setminus A, Y \setminus B)$, where A and B are of null Lebesgue measure.

Moreover, if the densities are in $C^{k,\eta}$, then $C^{0,\eta}$ can be replaced by $C^{k+1,\eta}$ in the conclusions above. In particular, if the densities are smooth, then the transport map is a diffeomorphism (between the reduced input and target domains if the target support is not convex).

Studying Neural Networks using Analogies with Physics The relation between neural networks and physical systems has a long lasting history that can be traced back to its very origin, as biological networks in animal and human brains were the very inspiration and motivation behind the creation of the artificial ones (Mcculloch et al. 1943), (Rosenblatt 1958). Later, variants such as the Hopfield network (Hopfield 1982) and the Boltzmann machine (Hinton et al. 1983) were introduced. These networks are heavily based on physics and bear resemblances with models for spin glass systems. This has led to the study of neural networks by applying theoretical analysis of the Ising model from statistical mechanics, linking the network's states with the states of magnetic systems (Amit 1989). This analogy has also been used to study feedforward networks (Choromanska

et al. 2014). More recently, another line of research considers the infinite width of neural networks during initialization in the in the neural tangent kernel regime (NTK)(Jacot et al. 2018b). This heavily simplifies the dynamics of gradient descent, making the dynamics of training amenable to theoretical study (J. Lee et al. 2019a). This infinite width limit has been applied in Section 5.2 to study the dynamics generative adversarial networks (I. J. Goodfellow et al. 2014a).

Analysis of CycleGAN Our work in Section 4.1 is motivated by the observations of works such as (Galanti et al. 2018; Benaim et al. 2018) which have linked well-behaved UDT models with a notion of simplicity which we tried here to frame in a more rigorous and more useful formulation, making it task dependant. Moreover, similarly to us, (Galanti et al. 2018; Benaim et al. 2017) show that learning a one-sided mapping is possible but do not directly obtain the inverse mapping as we do. Others have tried a hybrid approach between paired and unpaired translation (Tripathy et al. 2018), which still doesn't solve the problem of ill-posedness as there generally still are infinitely many possible mappings. Also similar to us, (Gong et al. n.d.) uses a progressive interpolation. In the domain adaptation field, using Optimal Transport to help a classifier extrapolate has been around for some years, e.g. (Courty et al. 2015; Damodaran et al. 2018) use a transport cost to align two distributions. The task, although related, is clearly different and so are the methods they develop. Finally, (G. Lu et al. 2018) also try to regularize CycleGAN through OT but use barycenters from the optimal plan obtained in the discrete, static setting in order to guide the mapping instead of seeing it directly as an OT map (or as biased towards it), thus not explaining why CycleGAN works in practice.

Analysis of Classification Networks That ResNets (K. He et al. 2016; Kaiming He et al. 2016a) are naturally biased towards minimally transforming their input, especially for later blocks and deeper networks, is already shown in (Jastrzebski et al. 2018), which found that earlier blocks learn new representations while later blocks only slowly refine those representations. (Hauser 2019) found that the deeper the network the more its blocks minimally move their input. Both were inspirations for this work. The ODE point of view of ResNets has inspired new architectures (Chang et al. 2018; Haber et al. 2019; Y. Lu et al. 2018; Ruthotto et al. 2020). Others were inspired by numerical schemes to improve stability, e.g. (Chang et al. 2018) add a penalty term that encourages the weights to vary smoothly from layer to layer and (J. Zhang et al. 2019) replicate an Euler scheme and study the effect of diminishing the discretization step-size. More recently, (Yan et al. 2020) accelerate the training of (R. Chen et al. 2018)'s model for generative tasks using the link with dynamical transport. But most often, regularization is achieved by

penalization of the weights (e.g. spectral norm regularization (Yoshida et al. 2017), smoothly varying weights (Chang et al. 2018)).

OT theory was used in (Sonoda et al. 2019) to analyse deep gaussian denoising autoencoders (not necessarily implemented through residual networks) as transport systems. In the continuous limit, they are shown to transport the data distribution so as to decrease its entropy. Closer to this work, the dynamical formulation of OT is used in (Bézenac et al. 2019) for the problem of unsupervised domain translation.

CHAPTER 3

LEARNING DYNAMICAL SYSTEMS WITH DEEP LEARNING

Contents

3.1 Learning Dynamical Systems from Partial Observations **24**
3.1.1 Introduction 24
3.1.2 Methodology 25
3.1.3 Experiments 30
3.1.4 Conclusion 38
3.2 Incorporating Partial Knowledge **42**
3.2.1 Introduction 42
3.2.2 Physical Motivation 44
3.2.3 Model 46
3.2.4 Experiments 49
3.2.5 Conclusion 57
3.3 Incorporating Imperfect Knowledge **60**
3.3.1 Introduction 60
3.3.2 Related Work 62
3.3.3 The APHYNITY Model 63
3.3.4 Experimental validation 67
3.3.5 Conclusion 73
3.4 Leveraging the Data From Different Environments **75**
3.4.1 Introduction 75
3.4.2 Approach 77
3.4.3 Improving generalization with *LEADS* 79
3.4.4 Experiments 84
3.4.5 Related work 90
3.4.6 Discussions 91

3.1 Learning Dynamical Systems from Partial Observations

abstract

We consider the problem of learning the dynamics of physical processes evolving in space and in time, given only partial observations of the state. We propose a natural data-driven framework, where the system's dynamics are modeled by an unknown time-varying differential equation, and the evolution term is estimated from the data, using a neural network. Given an initial state, an ODE solver can then be used to compute any future state. We qualitatively and quantitatively study the results of our method over simulations of fluid equations. We show that our method not only successfully forecasts future observations, consistently outperforming classical baselines, but it also learns to closely reproduce the unobserved dynamics of the state without direct supervision on the latter when the true initial state is given as input. We also show that our method can still be successfully applied when the initial state is not available and that it produces an interpretable hidden state even in this case.

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

Ibrahim Ayed, Emmanuel de Bézenac, Arthur Pajot, and Patrick Gallinari (2020). "Learning the Spatio-Temporal Dynamics of Physical Processes from Partial Observations". In: *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, pp. 3232–3236. URL: https://doi.org/10.1109/ICASSP40776.2020.9053035

3.1.1 Introduction

Let us consider a dynamical system to describe a real-world physical process. The *state* X_t ($t \in \mathbb{R}_+$ stands for time) of the dynamical system can then be defined as a vector-valued function on a space $x \in \Omega$:

$$\forall t, \ \frac{dX_t}{dt} = F(X_t). \tag{S1}$$

In this equation, X_t is sufficient to describe the temporal dynamics of the underlying process. For example, in the incompressible Navier Stokes equations a state can be the concatenation of the density and velocity fields of the fluid as

those are enough for describing the evolution of the system, while pressure, an additional variable of interest, can be computed from those variables.

With the availability of very large amounts of data captured via diverse sensors and recent advances of statistical methods, a new data-driven paradigm for modeling dynamical systems is emerging, where relations between the states are no longer handcrafted, but automatically discovered based on the available observations. This problem can be approached by considering some class of admissible functions $\{F_\theta\}$, and looking for a θ such that the solution X^θ of:

$$\frac{dX_t}{dt} = F_\theta(X_t) \tag{S2}$$

fits the measured data.

In this chapter, we consider the problem of learning complex spatiotemporal dynamical systems with neural networks from observation, which are only partially informative with respect to the full state.

First, we formulate the problem as a continuous-time optimization problem under the constraints of a Partial Differential Equation (PDE), where the parameters of the neural network are viewed as optimized variables. From this, we then present a natural algorithm for solving the resulting optimization problem, placing the neural network in an Ordinary Differential Equation solver in order to produce future predictions. Finally, we successfully apply our method to three increasingly challenging datasets and show promising results, comparing our approach to standard baselines.

- We propose a general model, parametrized with neural networks, which learns a state representation and its dynamics given partial observations.

- We present experiments on the Navier-Stokes equations exploring the properties of our model in each setting.

3.1.2 Methodology

In this section, we present the optimization problem defining our model to learn partially observed dynamics as well as the training algorithm we used.

3.1.2.1 Our Approach

Continuous State Space Models

We consider space-time dynamics for which X can be written as a function of $(t, x) \in \mathbb{R}_+ \times \Omega$ where t and x are respectively the time and space variables, $\Omega \subset \mathbb{R}^d$ is the domain over which we study the system. The spatial vector-valued function X_t contains the quantities of interest describing a studied physical system at time t.

In a realistic setting, the state is generally only partially observed *e.g.*, when studying the ocean's circulation, variables contained in the system's state such as temperature or salinity are observable, while others such as velocity or pressure are not. In other words, the measured data is only a projection of the complete state X_t. This measurement process is modeled here with a fixed operator linking the system's state X_t to the corresponding observation Y_t:

$$Y_t = \mathcal{H}(X_t) \tag{S3}$$

In the following, $\mathcal{H}$ is supposed to be known, fixed and differentiable. In most practical cases, this hypothesis is verified as $\mathcal{H}$ can usually be represented as a smooth operator. Let us note that, generally, the measurement process represents a considerable loss of information compared to the case where X is available, as the measurements may be sparse and low-dimensional.

Moreover, we assume that X obeys a differential equation of the general form of eq. S1, with an initial condition X_0. This leads us to the following continuous state space model:

$$\begin{cases} X_0 \\ \dfrac{dX_t}{dt} = F(X_t) \\ Y_t = \mathcal{H}(X_t) \end{cases} \tag{S4}$$

Optimization Problem

Our goal is to learn the differential equation driving the dynamics of a smooth state function X for which we only have supervision over observations Y through a fixed operator $\mathcal{H}$. In order to ensure that our dynamical system at least explains the observations, we define a cost functional of the form:

$$\mathcal{J}(Y, \widetilde{Y}) = \int_0^T \|Y_t - \widetilde{Y}_t\|^2 \mathrm{dt} \tag{S5}$$

Here, Y is a spatiotemporal field representing observations of the studied system, $\widetilde{Y}$ is the output of the system, and $\|\cdot\|$ the L2 norm.

The state X_t is constrained to follow the dynamics described by equation S2, starting from an initial condition X_0. The optimization problem is now formulated as :

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \mathbb{E}_{Y \in \text{Dataset}} \left[\mathcal{J}(Y, \mathcal{H}(X))\right] \\ \text{subject to} \quad & \frac{dX_t}{dt} = F_\theta(X_t), \\ & X_0 = g_\theta(Y_{-k}, \breve{X}_0) \end{aligned} \tag{S6}$$

where F_θ is a smooth vector valued function defining the trajectory of X, and g_θ gives us the initial condition X_0. In other words, θ parameterizes both the dynamics through F and the initialization through g. In particular, if a full initial state is given as input to the system, g_θ can be taken as independent from any parameter and does not need to be learned.

For any θ, we assume that F and g are such that there always exists a solution to the equation :

$$\begin{cases} X_0 = g_\theta(Y_{-k}, \breve{X}_0) \\ \dfrac{dX_t}{dt} = F_\theta(X_t) \end{cases} \tag{S7}$$

In the following, we will call such a solution X^θ.

Adjoint State Method

In order to construct a gradient descent algorithm to solve the problem S6, we need to find the gradient of the cost functional under the given constraints, *i.e.* the differential of $\theta \to \mathbb{E}_Y \mathcal{J}(Y, \mathcal{H}(X^\theta))$ (Plessix 2006). However, this implies calculating $\dfrac{\partial X^\theta}{\partial \theta}$, which is often very computationally demanding, as it implies solving $\dim(\theta)$ forward equations. The adjoint state method avoids those costly computations by considering the Lagrangian formulation of the constrained optimization problem. Adjoint State Method is a technique allowing to solve a constraint optimization problem well adapted when the number of parameters to optimize is large. We

introduce below a method for the case of continuous systems, constrained by an Ordinary Differential Equation (ODE).

Theorem S1 (Adjoint State Equation)**.** *The gradient of $\mathcal{J}$ w.r.t parameters θ, for a solution X^θ to the initial value problem described in S6 can calculated as follows.*

$$\frac{\partial}{\partial\theta}\mathcal{J}(Y,\mathcal{H}(X^\theta)) = -\int_0^T \left\langle \lambda_t, \partial_\theta F_\theta(X_t^\theta)\right\rangle \mathrm{dt} - \langle \lambda_0, \partial_\theta g_\theta\rangle \tag{S8}$$

where λ is the solution of:

$$\partial_t \lambda_t = A_t\lambda_t + B_t \tag{S9}$$

solved backwards, starting with $\lambda_T = 0$, and where :

$$A_t = -(\partial_X F_\theta(X_t^\theta))^\star$$

and

$$B_t = 2(\partial_X \mathcal{H}(X_t^\theta))^\star(\mathcal{H}(X_t^\theta) - Y_t)$$

where $M^\star$ denotes the adjoint operator of linear operator M.

Here A is the adjoint of F The proof can be found in Section A.1.3. When training, with this result, for a given value of θ, we can solve the forward eq. S2 to find X^θ. Then, λ can be solved backwards as its equation only depends on X^θ which gives us all necessary elements to calculate the gradient of $\mathcal{J}$. This gives us the following iterative algorithm to solve the optimization problem, starting from a random initialization of θ :

1. Solve the forward state eq. S7 to find X^θ ;
2. Solve the backward adjoint eq. S9 to find the corresponding λ (see Theorem S1);
3. Update θ in the steepest descent direction using eq. S8.

From these steps (and taking into account the estimation of the initial state, further explained in Section 3.1.3), we now have a general algorithm for training. *At inference*, we use the learned $g_{\theta^\star}$ to compute an initial state then simply solve the forward equation with the learned [1]

There are many possible choices regarding the way the different equations are solved in practice: Those are discussed in 3.1.2.1.

1. In particular, it is important to note that no further updates or corrections are made and no additional observations are needed.

Algorithm 1 Training Procedure

Input: Training samples $\{(Y_{-k}, \breve{X}_0,), Y_{+l}\}$.
Guess initial parameters θ
while not converged **do**
  Randomly select sample sequence $\{(Y_{-k}, \breve{X}_0,), Y_{+l}\}$
  if Initial State is Fully Observed **then**
    $X_0 \leftarrow \breve{X}_0$
  else
    $X_0 \leftarrow g_\theta(Y_{-k}, \breve{X}_0)$
  end if
  Solve Forward $\frac{dX_t}{dt} = F_\theta(X_t),\ X(0) = X_0,\ t \in [0, l]$
  Solve Backward $\dfrac{d\lambda_t}{dt} = A_t\lambda_t + B_t,\ \lambda_l = 0,\ t \in [0, l]$
  Compute gradient $\frac{\partial \mathcal{J}}{\partial \theta}(X^\theta)$
  Update θ in the steepest descent direction
end while
Output: Learned parameters θ.

Approximate Solutions

While our algorithm seems straightforward, solving the forward and backward equations (S2, S9) generally is not. Typically, they do not yield a closed-form solution and we must content ourselves with approximate solutions. There are essentially two different ways to tackle this problem: the *differentiate-then-discretize* approach, and the *discretize-then-differentiate* approach [2].

In a *differentiate-then-discretize* approach, one directly approximates the forward and backward equations using numerical schemes. Here, the approximation error to the gradient comes from the discretization error made in the solver for both the forward and backward equations. This method is used in the black box solvers presented in (T. Q. Chen et al. 2018a). It has the advantage of allowing the use of non-differentiable steps in the solver. However, this method can yield inconsistent gradients of cost functional $\mathcal{J}$, the discretization of the adjoint equations depends highly on the studied problem and therefore must be carefully selected and tuned (Carrassi et al. 2018).

In a *discretize-then-differentiate* approach, a differentiable solver for the forward equations is used, *e.g.* using an explicit Euler scheme $X^\theta_{t+\delta t} \approx X^\theta_t + \delta t F_\theta(X^\theta_t)$. Based on the solver's sequence of operations for the forward equations, the backward equations and the gradient can be directly obtained using automatic differentiation software (Paszke et al. 2017). This algorithm is actually equivalent to

2. The differentiate-then-discretize method is often referred to as the *continuous adjoint method*, and the *discretize-then-differentiate* approach as the *discrete adjoint method* (Sirkes et al. 1997).

backpropagation (LeCun et al. 1988) which can be derived as a special case of it. As the step-size approaches zero, the forward and backward equations are recovered.

It is important to note that, while the two methods are consistent and both converge to the equations derived in Theorem S1, they do not always yield the same results as the two approaches proceed differently. In our experiment, the second one proved more stable and the fact that we were limited to differentiable solvers experimentally was not an obstacle. This might not always be the case so the choice must be made after some exploration.

3.1.2.2 Models

We propose two variants of our models:

Setting 1: Jointly Trained (JT) States In this setting, we choose to fix the architectures of g_θ and F_θ and optimize without additional information. The dataset used here is thus only composed of observations and is of the form $\{(Y_{-k+1}^{(i)}, ..., Y_0^{(i)}, ..., Y_T^{(i)})\}$. In the following, the states learned in this setting will be referred to as **Jointly Trained** states.

Setting 2: Feeding in a Canonical Initial Condition A weak way to impose some structure over the learnt states is to remove g and prescribe an initial state with canonical structure[3] Thus, in this setting, the dataset used here is of the form $\{X_0^{(i)}, Y_1^{(i)}, ..., Y_T^{(i)})\}$ and the number of needed states is T **times less** than the number of observations.

There still are infinitely many possible state representations which produce accurate forecasts for observations, even when X_0 is fed as an input to the model. However, by correctly parametrising F, we can hope to conserve the structure of X_0 throughout the forecasts.

3.1.3 Experiments

In this section we evaluate our model, both quantitatively and qualitatively. We consider two different datasets corresponding to two dynamical systems. We evaluate our method with respect to its ability to predict observations and to reproduce the dynamics of the hidden state.

3. This comes at a cost: The algorithm now has to take a full state as input for each sequence of observations.

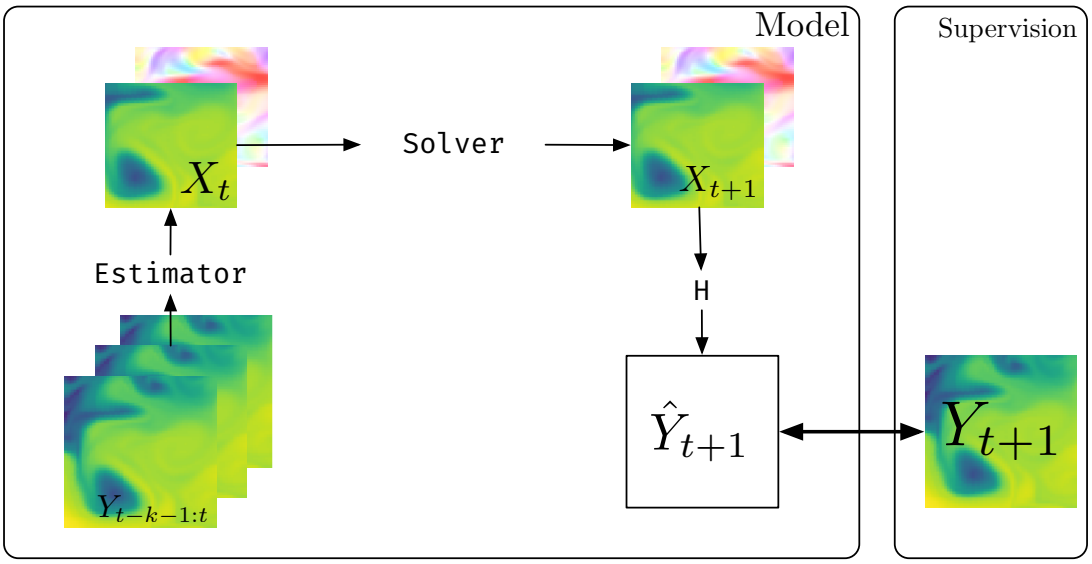


Figure S1. – Setting 1 : initial state is estimated from the input observation $Y_{t-k-1:t}$ and future states are estimated through the forecasting module. The partial observation Y is computed through the operator $\mathcal{H}$. The error signal is calculated as the Euclidean distance between $\hat{Y}_{t+1}$ and the real Y_{t+1}, and is backprogated through the operator scheme to correct the forecast and estimation module. To produce multiple time-step forecasts, the predicted observation is fed back in the model in an autoregressive manner.

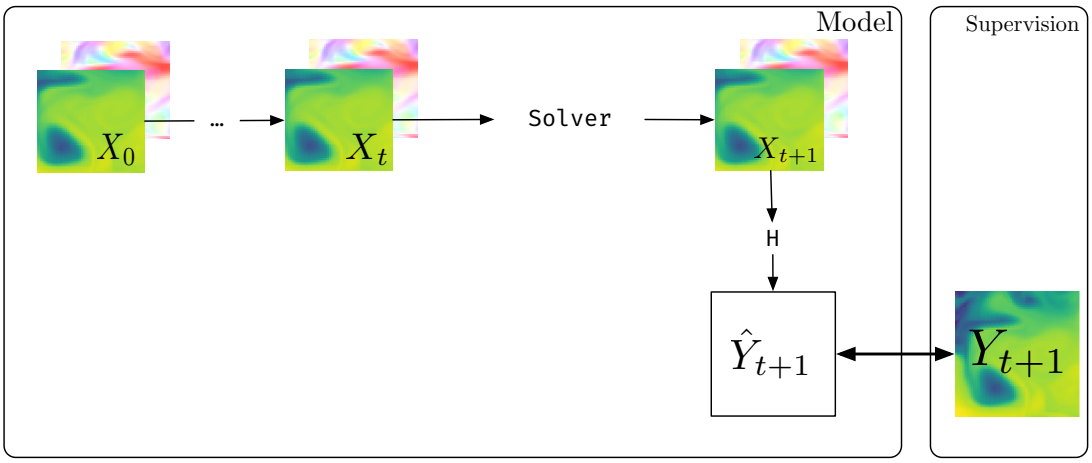


Figure S2. – Setting 2 : future state are estimated from the input state (X_0). The partial observation is computed through the operator $\mathcal{H}$. The error signal is calculated as the Euclidean distance between $\hat{Y}_{t+1}$ and the real Y_{t+1},, and is backprogated through the operator scheme to correct the forecast module. To produce multiple time-step forecasts, one simply apply the forecast F module on the estimated state.

3.1.3.1 Datasets.

Our two datasets are the following:

- **The Shallow Water equations** are derived from the Navier Stokes equations when integrating over the depth of the fluid (see supplementary material, A.1.1). These equations are discretized on a spatial 80×80 grid. We decompose the simulation into train-validation and test subsets of 600 and 1000 acquisitions images respectively.

- **The Navier-Stokes equations** (see A.1.1) are discretized on a spatial 64×64 grid. We use 15000 observations images for the train set and 10000 for the test.

3.1.3.2 Experimental setting.

In practice, the cost functional $\mathcal{J}$ is estimated on a minibatch of sequences from the dataset and optimized using stochastic gradient descent. For both datasets, the split between train, validation and test sets is made so that each split only includes sequences generated by *different, independently sampled* initial conditions. The two datasets are completely simulated: we then have access to the true full state to initialize our algorithm X_0 in eq. S6.

Implementations

Throughout *all* the experiments, F_θ is a standard convolutional residual network (Kaiming He et al. 2015),, with 2 downsampling layers, 6 residual blocks, and bilinear up-convolutions instead of transposed convolutions. To discretize the forward eq. S2 in time, we use a simple three steps Euler scheme. For the spatial discretization, we use the standard gridlike discretization induced by the dataset. The weights of the residual network θ are initialized using an orthogonal initialization. Our model is trained using an exponential scheduled sampling scheme with exponential decay, using the Adam optimizer, with a learning rate set to 1×10^{-5}. We use the Pytorch deep learning library (Paszke et al. 2017).

Metrics.

To evaluate our model's performance we consider the quality of the predictions, using the renormalized mean-squared error between generated and ground-truth observations, averaged over the time sequence, and the spatial coordinates.

$$\frac{1}{K}\frac{1}{|\Omega|}\sum_{k=1}^{K}\sum_{x\in\Omega}\frac{\|\mathcal{H}(X_k(x))-Y_k(x)\|_2}{\|Y_k(x)\|_2} \tag{S10}$$

To evaluate the quality of the hidden states, we use cosine similarity between the model's hidden state u and the truth hidden state of the system v [4]:

$$\frac{1}{K}\sum_{k=1}^{K}\frac{1}{|\Omega|}\sum_{x\in\Omega}\frac{\langle u(x),v(x)\rangle}{\|u(x)\|\,\|v(x)\|} \tag{S11}$$

For the velocity vector field representation, color represents the angle, and the intensity the magnitude of the associated vectors (see Figure S14).

Models and Baselines

We use two different baselines:

- **PKN** This is the physics-informed deep learning model developed in Section 3.2, where prior physical knowledge is integrated: it uses an advection-diffusion equation to link the velocity with the observed temperatures, and uses a neural network to estimate the velocities. The difference with the *Setting 1* model is that there is no forecast in the state space, but an explicit relation between the state X_t and Y_{t+1}. It does not model the full dynamical system but makes use of an autoregressive formulation.

- **PredRNN (Yunbo Wang et al. 2018a)** This is a heavyweight, state-of-the-art model used for video prediction tasks. It is based on a Spatiotemporal Long-Short Term Memory (LSTM) that models spatial deformations and temporal variations simultaneously. As for PKN, it is an auto-regressive model.

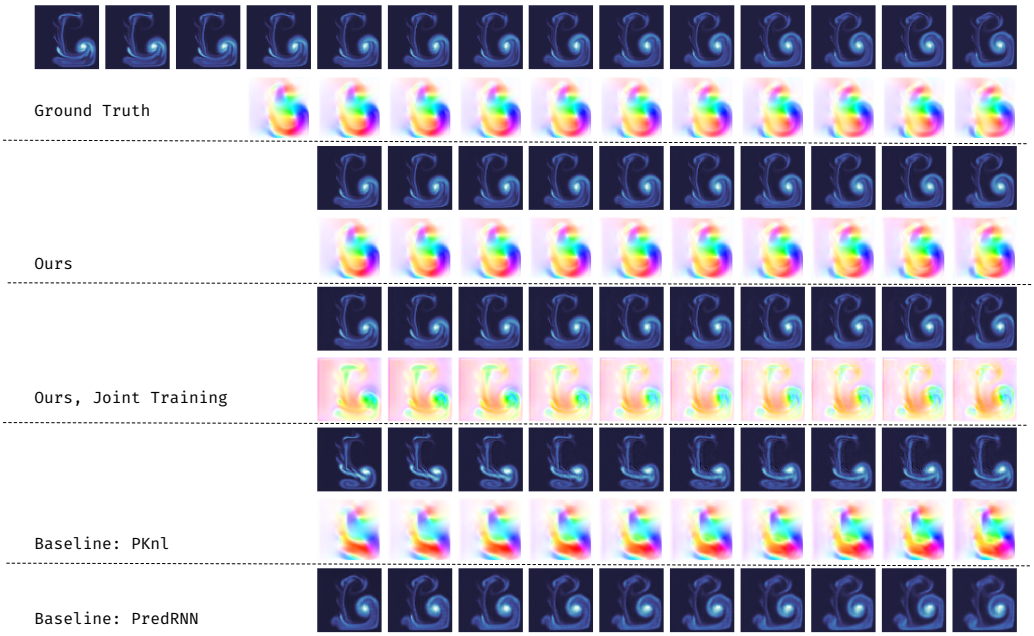


Figure S3. – Forecasting the Navier Stokes equations 10 time steps ahead with different models, starting from a given initial condition.

3.1.3.3 Results.

Experiments with Navier Stokes equations

Forecasting Observations. Figure S3 shows a sample of the predictions of our system over the test set for the Navier Stokes equations. The good results it shows are confirmed by the quantitative results in Table S1. Our model is able to predict observations up to a long forecasting horizon (42 time steps), which means that it generalizes to horizon it has never seen (at is has been train on sequences of size 6), and thus has managed to learn the dynamical system. Note that the initial state used at test time has never been seen at training time which means that the optimization problem was solved correctly without overfitting. The cost function and supervision are only computed with observation, has described in our experiments setting. An interesting remark is to observe that the jointly trained model is slightly less accurate than the one that makes use of the initial state X_0.

Hidden State Discovery. Both our model forecasts a full state X_t and not only the observations Y_t. In order to predict the observations correctly, our model has to learn to predict future hidden states that contain information of the true state.

4. We use the cosine similarity in order to compare with Prior Knowledge Network (PKN): the norm of its hidden state may not correspond to the ground truth norm.

Table S1. – Relative MSE and cosine similarity scores for our models and different baselines, at different temporal horizons on the Navier Stokes equations. As the PredRNN does not explicitly model the hidden state, we replace the cosine similarity scores for this baseline with XX.

Model	h=5		h=10		h=50	
	MSE	cosine	MSE	cosine	MSE	cosine
Setting 2	**0.118**	**0.798**	**0.180**	**0.679**	**0.628**	**0.483**
Setting 1	0.152	0.201	0.243	0.192	0.650	0.183
PKN	0.194	0.243	0.221	0.207	0.752	0.098
PredRNN (Yunbo Wang et al. 2018a)	0.170	XX	0.227	XX	0.719	XX

By feeding the true initial conditions to our model, we find that our method is able to learn the true dynamics of the hidden state with a good accuracy, while never directly enforcing a penalty on the latter. Note that the only access our method has to full states is through the initial state provided as input. This result is intriguing: the model should theoretically be able to use a state *encoding* that is different from the one given by the initial condition. We hypothesize that our network's architecture is biased towards preservation of the input code.

Comparison with baselines. Visually, as can be seen in Figure S3 by looking at the small features of the observations, our model manages to capture many details which are important for robust long-term forecasts while the PredRNN model, which proves to be a strong baseline at the level of observations[5] for the first few steps, produces less sharp predictions which explains its worse performance when evaluated on long-term predictions. Other samples for long-term forecasts of our model can be seen in the appendix for the Navier Stokes as well as for the Shallow Water equations.

Figure S4 and Figure S5 show some examples of forecasts obtained with our model and confirm its accuracy over long-term predictions

In order to explore the properties of our model, we conduct an ablation study, whose results are reported in 3.1.3.3.

5. It does not produce meaningful hidden states.

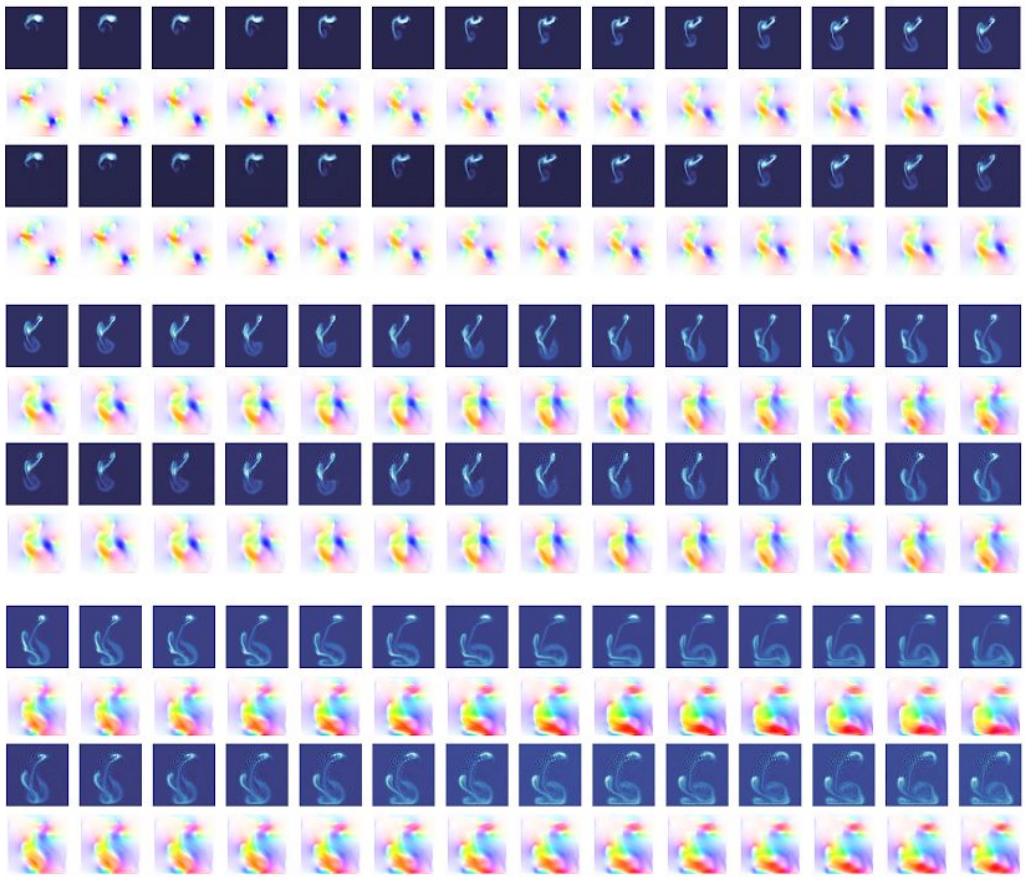


Figure S4. – Forecasting the Navier Stokes equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time stted. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequences is represented as 4 consecutive rows.

Experiments with Shallow Water equations

Table S2 shows the numerical evaluations for our model. Figures S6 and S7 show some examples of forecasts obtained with our model and confirm its accuracy over predictions, both for observations and velocities. Remember that the supervision is provided at the level of observations only.

Interpolation between data points. Our framework allows us to forecast for arbitrary times t, which means that if we only have access to samples at times $t, t+k, t+2k, \ldots$ we can still predict observations at time $t, t+1, t+2, \ldots$. This demonstrates the ability of our model to interpolate. Figure Figure S8 shows a sample of this interpolation mechanism. In this example, the model has been

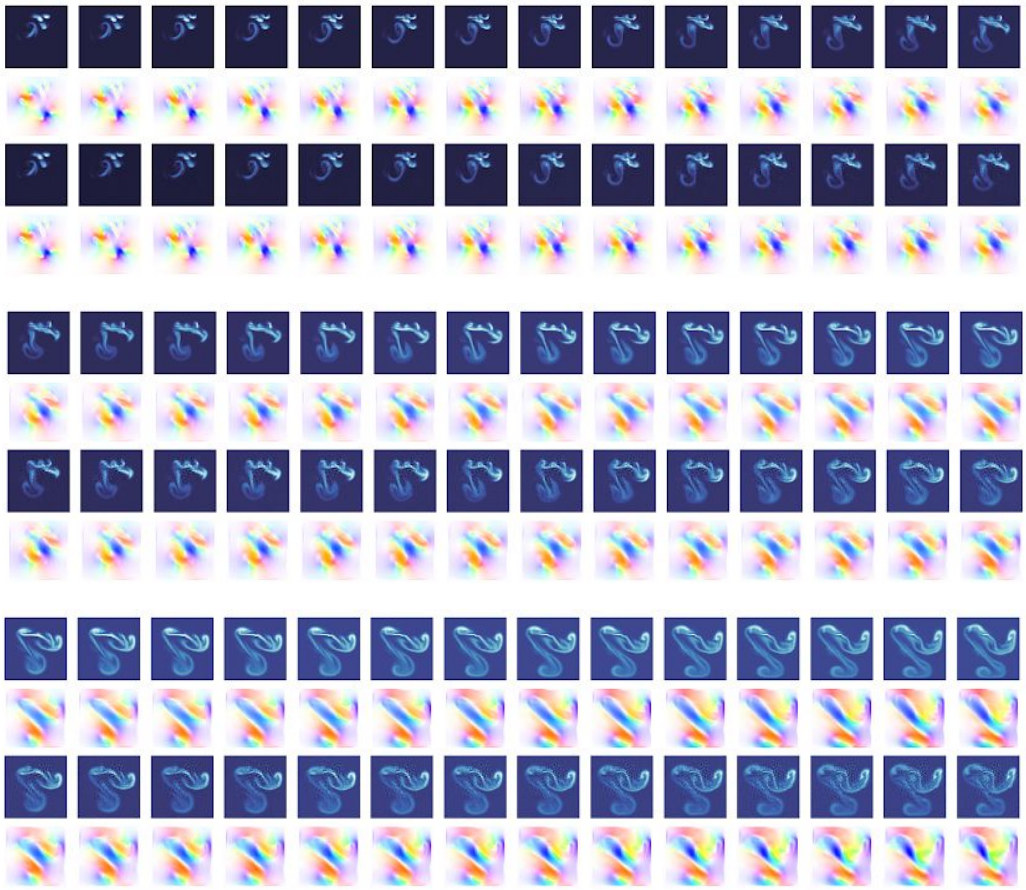


Figure S5. – Forecasting the Navier Stokes equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time stted. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequences is represented as 4 consecutive rows.

trained by regressing to the targets every 3 images (materialized on the figure by the red boxes). The outputs of the model are then compared with the unseen ground truth states. This shows that our approach allows us to learn the true evolution of the state. This is an important feature of our method, similar in this aspect to the claims of (T. Q. Chen et al. 2018a). It is applied here to a high-dimensional, highly non-linear and partially observed learned dynamical systems, for which we can interpolate the observations as well as the inferred hidden state.

Ablation Study

In order to better understand how our model works, we test different slightly modified versions of it on the Navier-Stokes dataset :

Table S2. – Relative MSE and cosine similarity scores for our model, at different temporal horizons on the Shallow Water equations

Model	h=5		h=10	
	MSE	cosine	MSE	cosine
Setting 2	0.1	0.995	0.12	0.992

Ours, Projection. Here we change the operator $\mathcal{H}$ and make it project to one dimension of the velocity field instead of on the pressure. We still give X_0 as input. The results, while slightly less good, are quite robust to this change, considering that we have not changed the hyperparameters of the model.

ResNet. Here we simply use a residual network, with *the exact same architecture* as the one used to parameterize our model. In other words, there are exactly the same number of parameters, layers,... as the F_θ which we learn and put into the solver. The results are notably less accurate for observations but, more importantly, this model turns out to be completely unable to forecast hidden states corresponding to the true ones. This shows that the way our model is structured around a solver which takes into account the differential structure of the studied problem is a strong regularizer.

ResNet no skip. This last argument may remind us that a residual network closely resembles the non-uniform discretization of an ODE. Thus, this should help it to perform well and explains the relatively good results on observations for the ResNet and, by getting rid of the skip connections while keeping all layers untouched, the performance should worsen. This is indeed what happens in our tests.

Unet. We tried using as F_θ this other classical architecture (Ronneberger et al. 2015), which is often used for regression problems, with roughly the same number of parameters as in our parameterization. It proved to be weak against our model.

3.1.4 Conclusion

We have introduced a general data-driven framework to predict the evolution of space-time processes, when the system is highly complex and non-linear and the state is not fully observed. Assuming the underlying system follows a time-

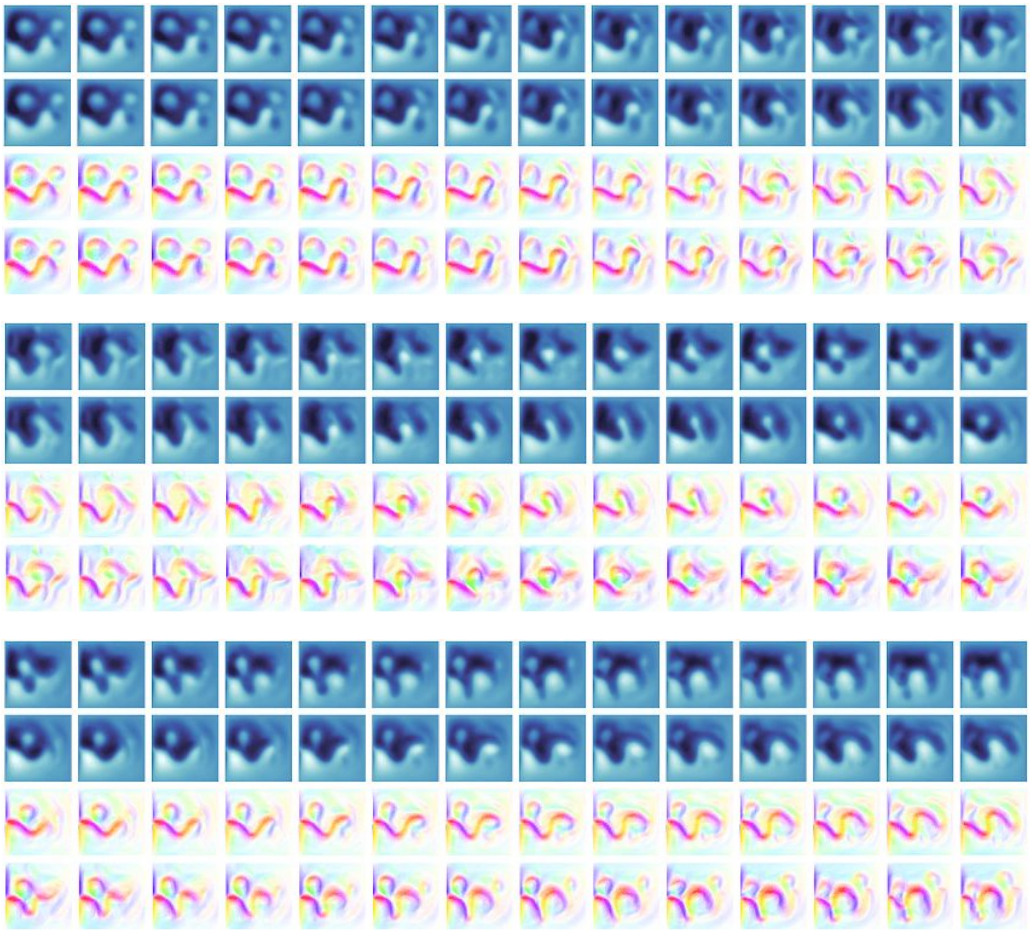


Figure S6. – Forecasting the shallow water equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time steps. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequence is represented as 4 consecutive rows.

dependent differential equation, we estimate the unknown evolution term with a neural network. This is in a natural way to model continuous-time systems. We propose a learning algorithm for this model. Experiments performed on two simulated datasets from fluid dynamics show that the proposed method not only is able to produce high quality forecasts at different horizons, but also learns with a good accuracy the underlying state space dynamics.

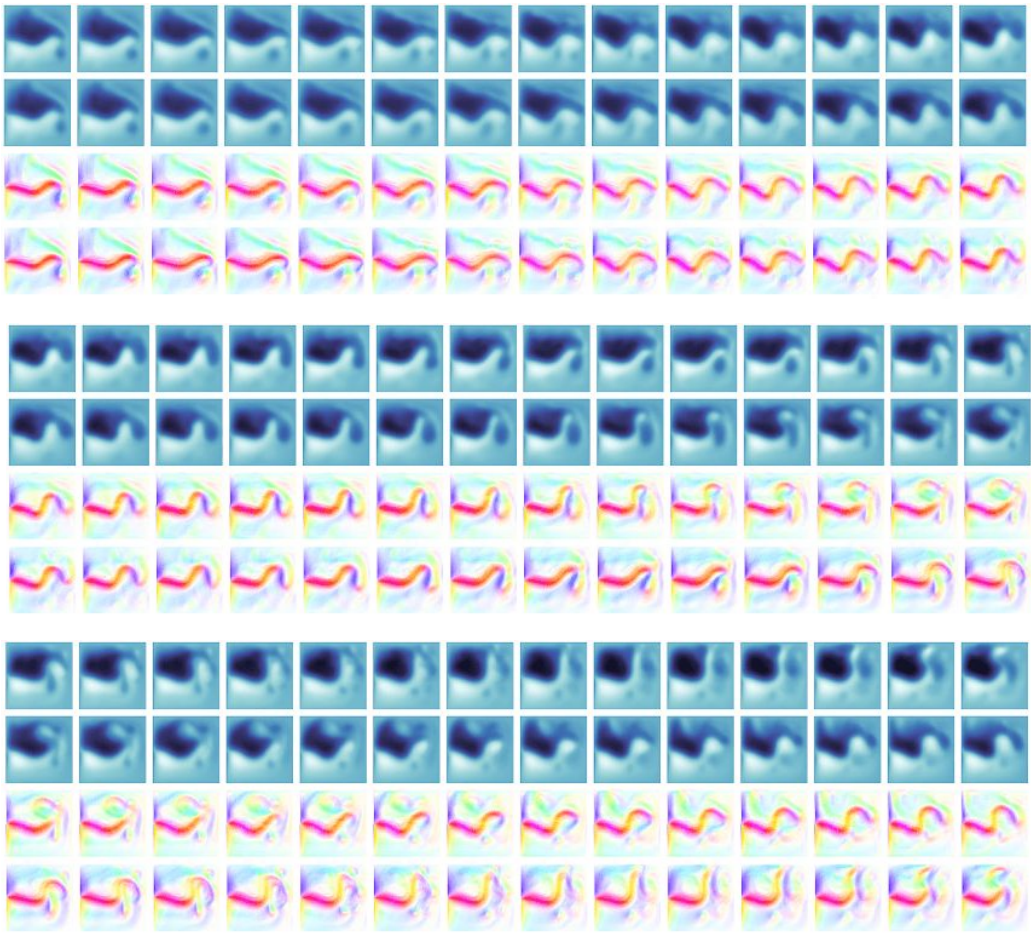


Figure S7. – Forecasting the shallow water equations, starting from a given initial condition (not shown here). We forecast 42 time steps ahead. We show in this figure 3 different sequences of 42 time steps. Top 2 rows correspond to the ground truth and the bottom rows correspond model forecasts. Each sequence is represented as 4 consecutive rows.

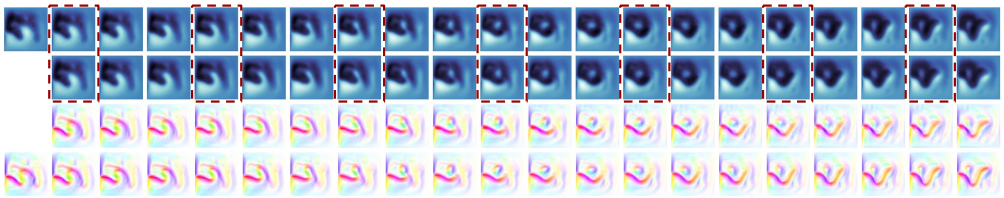


Figure S8. – Time interpolations with our approach on the test set. We train our model by regressing to the targets every 3 images (materialized by the red boxes). We then compare the outputs of the model with the unseen ground truth states.

Table S3. – Ablation study for our model, at different temporal horizons on the Navier Stokes equations

Model	h=5		h=10		h=50	
	MSE	cosine	MSE	cosine	MSE	cosine
Setting 2	**0.118**	**0.798**	**0.180**	**0.679**	0.628	0.483
Setting 1	0.191	0.432	0.288	0.620	**0.49**	**0.534**
Resnet	0.288	0.604	0.391	0.333	0.73	0.032
Unet	0.659	0.069	0.692	0.028	0.84	0.023
Resnet No Skip	0.615	0.162	0.71	0.060	0.897	-0.04

3.2 Incorporating Partial Knowledge

abstract

In this chapter, we consider the use of Deep Learning methods for forecasting complex phenomena, like those occurring in natural physical processes. With the large amount of data gathered on these phenomena the data intensive paradigm could begin to challenge more traditional approaches elaborated over the years in fields like mathematics or physics. Using an example application, namely Sea Surface Temperature Prediction, we show how general background knowledge gained from the physics, under the form of Partial Differential Equation (PDE) could be used as a guideline for designing efficient Deep Learning models. In order to motivate the approach and to assess its generality, we demonstrate a formal link between the solution of a class of differential equations underlying a large family of physical phenomena and the proposed model. Experiments and comparison with series of baselines including a state-of-the-art numerical approach is then provided.

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

Emmanuel de Bezenac, Arthur Pajot, and Patrick Gallinari (Feb. 2018). "Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge". In: URL: https://openreview.net/forum?id=By4HsfWAZ

3.2.1 Introduction

A physical process is a sustained phenomenon marked by gradual changes through a series of states occurring in the physical world. Physicists and environmental scientists attempt to model these processes in a principled way through analytic descriptions of the scientist's prior knowledge of the underlying processes. Conservation laws, physical principles or phenomenological behaviors are generally formalized using differential equations. This physical paradigm has been, and still is the main framework for modeling complex natural phenomena like e.g., those involved in climate.

With the availability of very large datasets captured via different types of sensors, this physical modeling paradigm is being challenged by the statistical Machine Learning (ML) paradigm, which offers a prior-agnostic approach. However, despite impressive successes in a variety of domains as demonstrated by the deployment of Deep Learning (DL) methods in fields such as vision, language, speech, etc., the statistical approach is not yet ready to challenge the physical

paradigm for modeling complex natural phenomena, or at least it has not demonstrated how to. We believe that knowledge and techniques accumulated for modeling physical processes in well-developed fields such as maths or physics could be useful as a guideline to design efficient learning systems and conversely, that the ML paradigm could open new directions for modeling such complex phenomena.

In this chapter we try to answer a fundamental question, that rose from those observations : how general knowledge gained from the physical modeling paradigm could help designing efficient ML models ?

We tackle these questions by considering a specific physical modeling problem: forecasting Sea Surface Temperature (SST). SST plays a significant role in analyzing and assessing the dynamics of weather and other natural systems. Accurately modeling and predicting such dynamics is critical in various applications such as weather forecasting, or planning of coastal activities. Since 1982, weather satellites have made huge quantities of very high resolution SST data available (Bernstein 1982). Standard physical methods for forecasting SST use coupled ocean-atmosphere prediction systems, based on the Navier Stokes equations. These models rely on multiple physical hypotheses and do not optimally exploit the information available in the data. On the other hand, despite the availability of large amounts of data, direct applications of ML methods do not lead to competitive state-of-the-art results, as will be seen in Section 3.2.4.

We use SST as a typical and representative problem of intermediate complexity. Our goal is not to offer one more solution to this problem, but to use it as an illustration for advancing on the challenges mentioned above. The way we handle this problem is general enough to be transferred to a more general class of transport problems.

We propose a Deep Neural Netwok model, inspired from general physical motivations which offers a new approach for solving this family of problems. We first motivate our approach by introducing in Section 3.2.2 the solution of a general class of Partial Differential Equation (PDE) which is a core component of a large family of transport and propagation phenomena in physics. This general solution is used as a guideline for introducing a Deep Learning architecture for SST prediction which is described in Section 3.2.3. Experiments and comparison with a series of baselines are introduced in Section 3.2.4.

The main contributions presented in this chapter are

- An example showing how to incorporate general physical background for designing a Neural Network (NN) aimed at modeling a relatively complex prediction task. We believe the approach to be general enough to be used for a family of transport problems obeying general advection-diffusion principles.

- Formal links between our model's prediction and the solution of a general advection diffusion PDE.

- An unsupervised model for estimating motion fields, given a sequence of images.

- A proof, on a relatively complex physical modeling problem, that full data intensive approaches based on deep architectures can be competitive with state of the art dedicated numerical method.

3.2.2 Physical Motivation

3.2.2.1 Incorporating the Advection Diffusion Equation

In this section, we introduce, the Advection Diffusion Equation, which is an example of PDE characterizing flow displacement. Forecasting consists in predicting future temperature maps using past records. Temperatures are acquired via satellite imagery. If we focus on a specific area, we can formulate the problem as prediction of future temperature images of this area using past images as:

$$I(x,t) = I(x + \Delta x, t + \Delta t). \tag{S12}$$

Applying a first order Taylor expansion of the time and space in the right-hand side and moving the resulting terms to the left-hand side of equation Equation S12, we obtain the *advection equation*, also known as the Brightness Constancy Constraint Equation (BCCE):

$$\frac{\partial I}{\partial t} + (w.\nabla)I = 0, \tag{S13}$$

Where ∇ denotes the gradient operator and w the motion vector $\frac{\Delta x}{\Delta t}$. This equation describes the temporal evolution of quantity I for displacement w. Note that this equation is also the basis for many variational methods for *Optical Flow*. To retrieve the motion, numerical schemes are applied, and the resulting system of equations, along with an additional constraint on w is solved for w. This motion can then be used to forecast the future value of I.

Advection alone is not sufficient to explain the evolution of many physical processes (including SST). *Diffusion* corresponds to the movement which spreads out the quantity I from areas of high concentration to areas of low concentration. Both advection and diffusion should be considered together. The following equation describes the transport of quantity I through advection and diffusion:

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I. \tag{S14}$$

∇^2 denotes the Laplacian operator and D the diffusion coefficient. Note that when $D \to 0$, we recover the advection Equation S13.
This equation describes a large family of physical processes (e.g., fluid dynamics, heat conduction, wind dynamics, etc.). Let us now state a result, characterizing the general solutions of Equation S14.

Theorem S2. [6] *For any initial condition $I_0 \in L^1(\mathbb{R}^2)$ with $I_0(\pm\infty) = 0$, there exists a unique global solution $I(x,t)$ to the advection-diffusion equation Equation S14:*

$$I(x,t) = \int_{\mathbb{R}^2} k(\,x - w,\, y)\; I_0(y)\, dy, \tag{S15}$$

Where $k(u,v) = \frac{1}{4\pi Dt} e^{-\frac{1}{4Dt}\|u-v\|^2}$ is a radial basis function kernel, or alternatively, a 2 dimensional Gaussian probability density with mean u and variance $2Dt$.

For this theorem, we make the hypothesis that w is constant locally (around x) in space and time.

Equation S15 provides a principled way to calculate $I(x,t)$ for any time t using the initial condition I_0, provided the motion w and the diffusion coefficient D are known. It states that quantity $I(x,t)$ can be computed from the initial condition I_0 via a convolution with a Gaussian probability density function. In other words, if I was used as a initial condition for the evolution of the SST and the surface's underlying advecting mechanisms were known, future surface temperatures could be predicted from previous ones. Unfortunately neither the initial conditions, the motion vector nor the diffusion coefficient are known.

They have to be estimated from the data. Inspired from the general form of Equation S15, we propose a method, expressed as a Deep Learning architecture for predicting the SST evolution. This model will learn to predict a motion field analog to the w in Equation S15, which will be used to predict future images. This method can then be see as a warping of previous acquisitions along motion w.

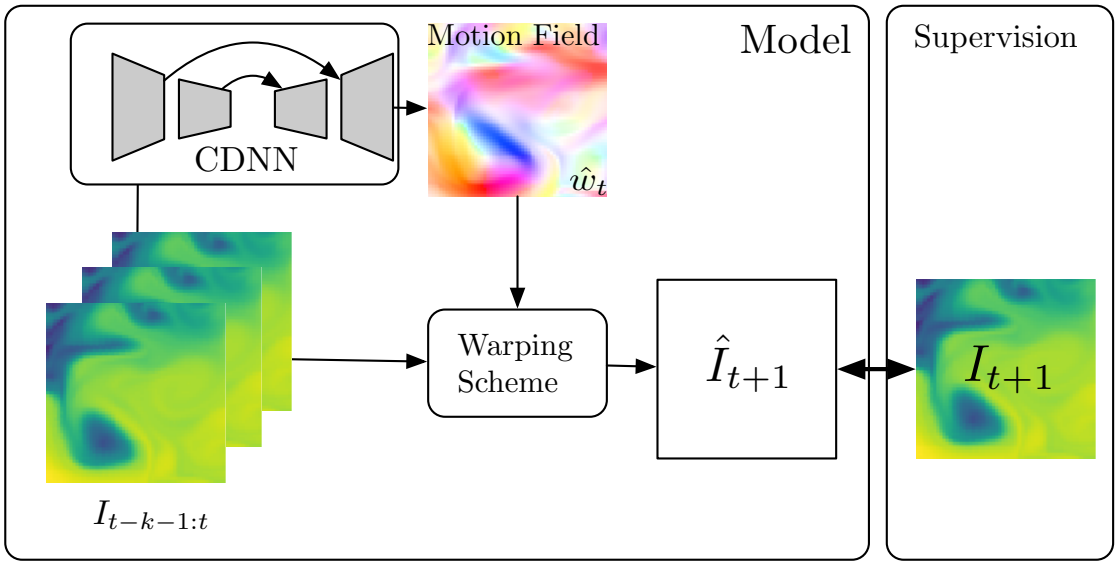


Figure S9. – Motion is estimated from the input images ($I_{t-k-1:t}$) with a Neural Network. A warping scheme then displaces the last input image along this motion estimate to produce the future image. The error signal is calculated using the target future image I_{t+1}, and is backprogated through the warping scheme to correct the Neural Network. To produce multiple time-step forecasts, the predicted image is fed back in the Neural Network in an autoregressive manner.

3.2.3 Model

3.2.3.1 Model Description

The model consists of two main components, as illustrated in Figure S9. One predicts the motion field from a sequence of past input images and the other warps the last input image using the motion field from the first component, in order to produce an image forecast. The entire system is trained in an end-to-end fashion, using only the supervision from the target SST image. By doing so, we are able to produce an interpretable latent state which corresponds in our problem to the velocity field advecting the SST.

Let us first introduce some notations. Each SST image I_t is acquired on a bounded rectangle of $\mathbb{R}^2$, named Ω. We denote $I_t(x)$ and $w_t(x)$ the sea surface temperature and the two-dimensional motion vector at time $t \in \mathbb{R}$ at position $x \in \Omega$. $I_t : \Omega \to \mathbb{R}$ and $w_t : \Omega \to \mathbb{R}^2$ represent the temperatures and the motion vector field at time t defined on Ω. When time t and position x are available from the context, we will drop the subscript t from $w_t(x)$ and $I_t(x)$, along with x for

6. A proof of this theorem is available in Section A.1.2

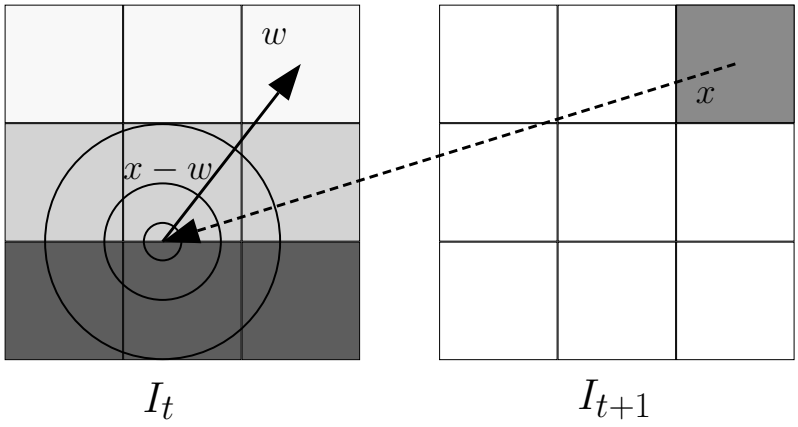


Figure S10. – Warping scheme. To calculate the pixel value for time $t+1$ at position x, we first compute its previous position at time t, *i.e.* $x - w$. We then center a Gaussian in that position in order to obtain a weight value for each pixel in I_t based on its distance with $x - w$, and compute a weighted average of the pixel values of I_t. This corresponds to the diffusion mechanism. This weighted average will correspond to the new pixel value at x in I_{t+1}.

clarity. Given a sequence of k consecutive SST images $\{I_{t-k-1}, ..., I_t\}$ (also denoted as $I_{t-k-1:t}$), our goal is to predict the next image I_{t+1}.

As indicated in Section 3.2.2.1, provided the underlying motion field is known, one can compute SST forecasts. Let us introduce how the motion field is estimated in our architecture. We are looking for a vector field w which when applied to the geometric space Ω renders I_t close to I_{t+1}, i.e. $I_{t+1}(x) \simeq I_t(x + w(x))$, $\forall x \in \Omega$. During inference, if I_{t+1} were known, we could estimate w, but I_{t+1} is precisely what we are looking for. Instead, we choose to use a NN architecture to predict a motion vector for each pixel.

Generally, and this is the case for our problem, we do not have a direct supervision on the motion vector field, since the target motion is usually not available. Using the warping scheme introduced below, we will nonetheless be able to supervise w, based on the discrepancy of the warped version of the I_t image and the target image I_{t+1}.

3.2.3.2 Warping Scheme

Discretizing the solution of the advection-diffusion equation in Section 3.2.2.1 by replacing the integral with a sum, and setting image I_t as the initial condition, we obtain a method to calculate the future image, based on the motion field estimate $\hat{w}$. The latter is used as a warping scheme:

$$\hat{I}_{t+1}(x) = \sum_{y \in \Omega} k(\, x - \hat{w}(x),\, y)\; I_t(y) \tag{S16}$$

Where $k(x - \hat{w}, y) = \frac{1}{4\pi D \Delta t} e^{-\frac{1}{4D\Delta t}\|x - \hat{w} - y\|^2}$ is a radial basis function kernel, as in Equation S15, parameterized by the diffusion coefficient D and the time step value Δt between t and $t+1$ and $\hat{w}$ is the estimated value of the vector flow w. To calculate the temperature for time $t+1$ at position x, we compute the scalar product between $k(x - \hat{w}, .)$, a Gaussian centered in $x - \hat{w}$, and the previous image I_t. Simply put, it is a weighted average of the temperatures I_t, where the weight values are larger when the pixel's positions that are closer to $x - \hat{w}$. Informally, $x - \hat{w}$ corresponds to the pixel's previous position at time t. See Figure S10.

Equation S16 simply says that estimate $\hat{I}_{t+1}(x)$ is the result of a convolution of the whole image I_t with the kernel $k(\, x - \hat{w}(x),\, y)$ where y describes the domain Ω of I_t.

As seen by the relation with the solution of the advection-diffusion equation, the proposed warping mechanism is then clearly adapted to the modeling of phenomena governed by the advection-diffusion equation. Fluid forecasting is a particular case, but the proposed scheme can be used for any problems in which advection and diffusion are occurring. Moreover, this warping scheme is entirely differentiable, allowing backpropagation of the error signal to the motion field estimating module.

This warping mechanism has been inspired by the model presented in (Jaderberg et al. 2015), originally designed to be incorporated as a layer in a convolutional neural network architecture in order to gain invariance under geometric transformations. Using the notations in (Jaderberg et al. 2015), when the inverse geometric transformation $\mathcal{T}_\theta$ of the grid generator step is set to $\mathcal{T}_\theta(x) = x - \hat{w}(x)$, and the kernels $k(\, .\, ; \Phi_x)$ and $k(\, .\, ; \Phi_y)$ in the sampling step are radial basis function kernels, we recover our warping scheme. The latter can be seen as a specific case of the Spatial Transformer Network (STN), without the localization step. It theoretically grounds the use of the STN for Optical Flow in many recent articles (Patraucean et al. 2015; Finn et al. 2016): in Equation S14, when $D \to 0$, we recover the Brightness Constancy Constraint Equation, used in the latter.

3.2.3.3 Loss Function

At each iteration, the model aims at forecasting the next observation, given the previous ones. We evaluate the discrepancy between the warped image $\hat{I}_{t+1}$ and the target image I_{t+1} using the Charbonnier penalty function $\rho(x) = (x + \epsilon)^{\frac{1}{\alpha}}$, where ϵ and α are parameters to be set. Note that with $\epsilon = 0$ and $\alpha = \frac{1}{2}$, we recover the ℓ_2 loss.

The Charbonnier penalty function is known to reduce the influence of outliers compared to an l_2 norm. It behaved slightly better in preliminary experiments. We have also tested the Laplacian pyramid loss (Ling et al. 2016), where we enforce convolutions of all deconvolutional layers to be close to down-sampled versions of the target image in the Charbonnier penalty sense, but we have observed an overall decrease in generalization performance.

The proposed neural network model has been designed according to the intuition gained from general background knowledge of a physical phenomenon, here advection-diffusion equations. Additional prior knowledge – expressed as partial differential equations, or through constraints – can be easily incorporated in our model, by adding penalty terms in the loss function. As the displacement w is explicitly part of our model, one strength of our model is its capacity to apply some regularization term directly on the motion field. The following quantities are prior knowledge that could be seen as constraints. In our experiments, we tested the influence of different terms: divergence $\nabla . \, w_t(x)^2$ which locally control the variation of the motion field, magnitude $\|w_t(x)\|^2$ which controls the amplitude of the motion field, and smoothness $\|\nabla w_t(x)\|^2$ which controls the amplitude of its variation. They are, in our case, hyperparameters set by cross validation. The loss function can be written as :

$$\begin{aligned} \mathcal{L}_t = & \sum_{x \in \Omega} \rho(\hat{I}_{t+1}(x) - I_{t+1}(x)) \\ & + \lambda_{\text{div}} (\nabla . \, w_t(x))^2 + \lambda_{\text{magn}} \|w_t(x)\|^2 + \lambda_{\text{grad}} \|\nabla w_t(x)\|^2 . \end{aligned} \tag{S17}$$

3.2.4 Experiments

In this section we evaluate our model, both quantitatively and qualitatively. We consider a dataset of medium complexity representing the evolution of the sea surface temperature. We evaluate our method with respect to its ability to predict observations and to reproduce the dynamics of the hidden state. For the first two datasets, we use the full initial condition as input. For the last dataset, we only have access to a subset of the states and weight propose a variant of our approach in order to accommodate this situation.

3.2.4.1 Dataset

Since 1982, high resolution SST data has been made available by the *NOAA6* weather satellite (Bernstein 1982). Dealing directly with these data requires a lot of preprocessing (e.g., some regions are not available due to clouds hindering temperature acquisition). In order to avoid such complications which are beyond

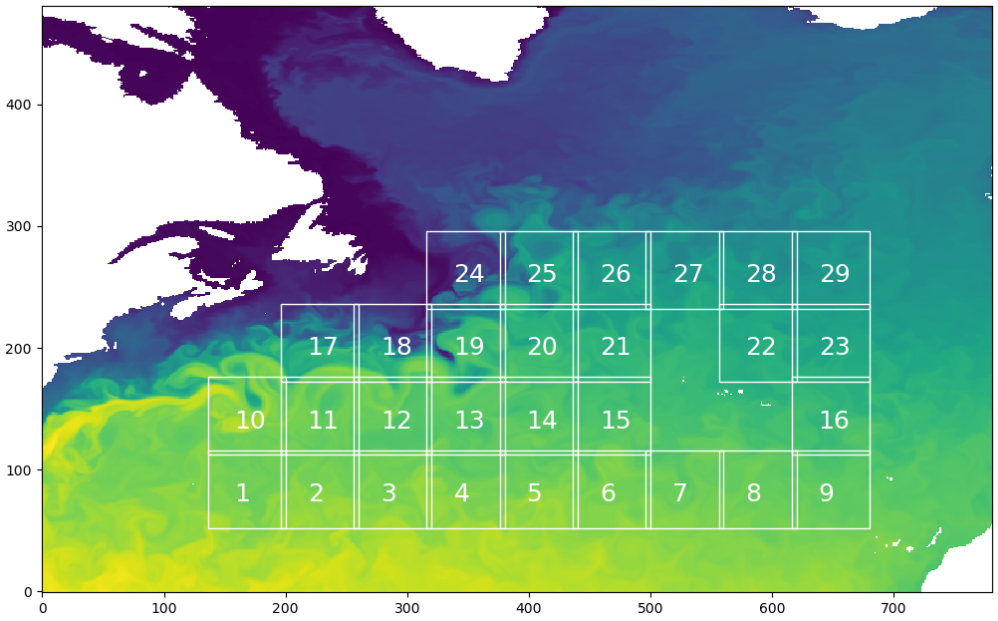


Figure S11. – Sub regions extracted for the dataset. Test regions are regions 17 to 20.

the scope of this work, we used synthetic but realistic SST data of the Atlantic Ocean generated by a sophisticated simulation engine: NEMO (Nucleus for European Modeling of the Ocean) engine G. Madec 2008. NEMO is a state-of-the-art modeling framework of ocean-related engines. It is a primitive equation model adapted to the regional and global ocean circulation problems. Historical data is accumulated in the model to generate a synthesized estimate of the states of the system using data *reanalysis*, a specific data assimilation scheme, which means that the data does follow the true temperatures. The resulting dataset is built of daily temperature acquisitions of 481 by 781 pixels, from 2006-12-28 to 2017-04-05 (3734 acquisitions).

We extract 64 by 64 pixel sized sub-regions as indicated in Figure S11.

3.2.4.2 Experimental Setting

We decompose the simulations into training sequences of fixed length, using 4 time steps as input, and 6 time steps for the target sequence. More precisely, we have $I_{t:t+4}$ as an input of the model. We then estimate I_{t+5} and concatenate it to the previous input in order to estimate the following images until I_{t+10}. The loss is computed between $I_{t+4:t+10}$ and the estimated $\hat{I}_{t+4:t+10}$. In practice, the cost functional $\mathcal{L}$ is estimated on a minibatch of sequences from the dataset and optimized using stochastic gradient descent. We use data from years 2006 to 2015

for training and validation (94743 training examples), and years 2016 to 2017 for testing. We withhold 20% of the training data for validation, selected uniformly at random at the beginning of each experiment. For the tests we used sub-regions enumerated 17 to 20 in Figure S11, where the interactions between hot and cold waters make the dynamics interesting to study. All the regions numbered in Figure S11, from 2006 to 2015 were used for training[7]. Each sequence of images used for training or for evaluation corresponds to a specific numbered sub-region. We make the simplifying hypothesis that the data in a single sub-region contains enough information to forecast the future of the sub-region. As the forecast is for a small temporal horizon, we can assume that the influence from outside the region is small enough. The boundary conditions are set to zero, which means that any pixel dispaced inside of each box, will be set to zero.

Concerning the constraints on the vector field w (Equation S17. the regularization coefficients selected via validation are $\lambda_{\text{div}} = 1$, $\lambda_{\text{magn}} = 0$ and $\lambda_{\text{grad}} = 0.4$. The diffusion coefficient D was set to 0.45 by cross validation. We also compare the results with the model without any regularization.

3.2.4.3 Model Architecture

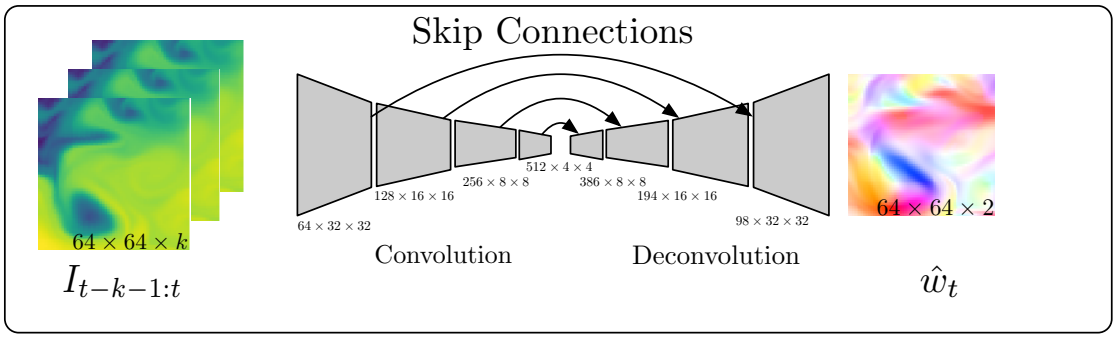


Figure S12. – Architecture of the NN motion estimation component. For the estimated motion flow $\hat{w}_t$, colors correspond to the flow orientation and color intensity to the flow intensity

As shown in Figure S12, the network that we used during our experiments, makes use of skip connections (Kaiming He et al. 2015), allowing fine-grained information from the first layers to flow through in a more direct manner. We use a *Batch Normalization* layer between each convolution, and *Leaky ReLU* (with parameter value set to 0.1) non-linearity between convolutions and transposed-convolutions. We used $k = 4$ concatenated images $I_{t-k-1:t}$ as input for training.

7. non-numbered regions correspond to land and not sea on the figure

We have selected this architecture experimentally, testing different state-of-the-art convolution-deconvolution network architectures.

3.2.4.4 Baselines

We compare our model, which is called Prior Knowledge Network (PKN), with several baselines. Each model is evaluated with a mean square error metric, forecasting images on a horizon of 6 (we forecast from I_{t+1} to I_{t+6} and then average the Mean Squared Error (MSE)). The hyperparameters are tuned using the validation set. Neural network based models are run on a Titan Xp GPU, and runtime is given for comparison purpose.

Our reference model for forecasting is (Bérézíat et al. 2015), a numerical assimilation model which relies on data assimilation. In (Bérézíat et al. 2015), the ocean's dynamics are modeled using shallow water equations (Vallis 2017) and the initial conditions, along with other terms, are estimated using complex data assimilation techniques (Trémolet 2006). This is a state-of-the-art assimilation model for predicting ocean dynamic, here SST.

The other baselines are :

- An autoregressive convolutional-deconvolutional neural network (ACDNN), with an architecture similar to the Neural Network module described in Section 3.2.4.3, but trained to predict the future image directly, without explicitly representing the motion vector field. Each past observation is used as an input channel (the 4 input images used in the experiments are concatenated), and the output is used as new input for multi-step forecasting, as described in Section 3.2.4.2.

- (Shi et al. 2015b), a recurrent model similar to Long-Short Term Memory (LSTM), which uses convolutional transitions in the inner LSTM module.

- The model in (Mathieu et al. 2015) which is a multi-scale ACDNN trained as a Generative Adversarial Network (GAN).

3.2.4.5 Results.

3.2.4.6 Quantitative Results

Quantitatively, our model performs well. The MSE score is better than any of the baselines. The closest neural network baseline is described in (Mathieu et al. 2015) and regularizes a regression convolution-deconvolution model with a GAN. The performance is, however, clearly below the proposed model and it does not

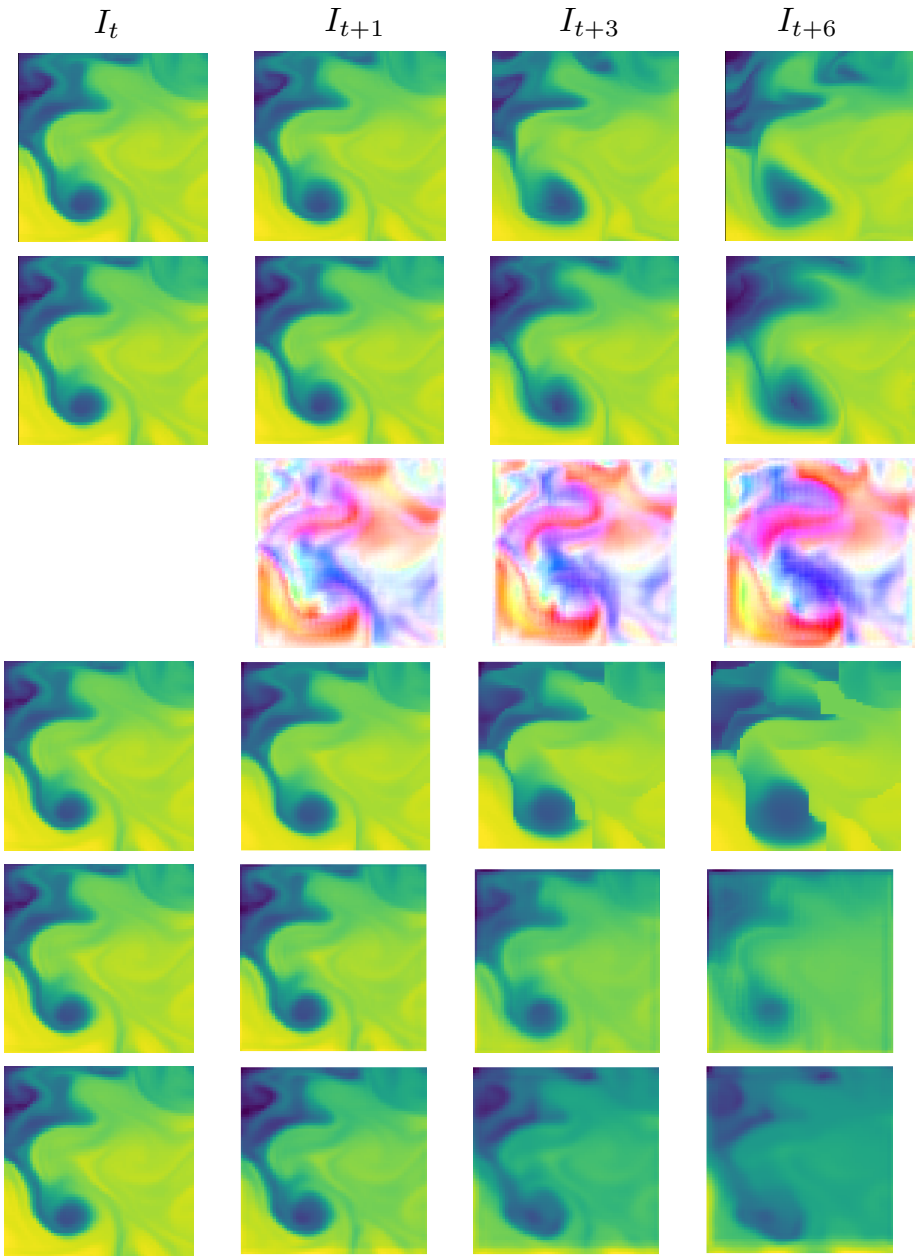


Figure S13. – From top to bottom: target, our model prediction, our model flow, numerical assimilation model, ACDNN, *ConvLSTM*. Data correspond to daily temperatures from January 17 to January 23, 2017, for regian 19.

Model	Average Score (MSE)	Average Time
Numerical model (Bérériat et al. 2015)	1.99	4.8 s
ConvLSTM (Shi et al. 2015b)	5.76	0.018 s
ACDNN	15.84	0.54 s
GAN Video Generation (Mathieu et al. 2015)	4.73	0.096 s
PKN with regularization	**1.42**	0.040 s
PKN without regularization	2.01	0.040 s

Table S4. – Average score and average time on test data. Average score is calculated using the *mean square error* metric (MSE), time is in seconds.

allow to easily incorporate prior constraints inspired from the physics of the phenomenon. ACDNN is a direct predictor of the image sequence, implemented via a NN module identical to the one used in our model. Its performance is poor. Clearly, a straightforward use of prediction models is not adapted to the complexity of the phenomenon. The Convolutional Long-Short Term Memory (ConvLSTM) performs better: as opposed to the ACDNN, it seems to be able to capture a dynamic, although not very accurately. Overall, direct prediction models are not able to capture the complex underlying dynamics and they produce blurry sequences of images. The GAN explicitly forces the network output to eliminate the blurring effect and then makes it able to capture short term dynamics. The state-of-the-art numerical model (Bérériat et al. 2015), performs well and has comparable performance with PKN, although it incorporates more prior constraints. This shows that pure ML models, when conceived adequately and when trained with enough data, can be competitive with state-of-the-art dedicated models. Regularizing the motion vector w notably increases the performance with respect to the unregularized model.

As for the running time, the proposed model is extremely fast, being just above the ConvLSTM model of (Shi et al. 2015b). The running time of (Bérériat et al. 2015)'s model is not comparable to the others. It was run on a CPU (no GPU code) when all the others were run on Titan Xp GPUs. However, an optimization procedure is required to estimate the motion field, and it is clearly slower than the straightforward NN predictions. Moreover, in order to prevent the numerical scheme from diverging, multiple intermediate forecasts are required.

Besides MSE, we need to analyze the prediction samples qualitatively. Figure S13 shows predictions obtained by the different models. On the top row of Figure S13, the ground truth for a sequence of 4 temperature images corresponding to time t, $t+1$, $t+3$ and $t+6$. The second row corresponds to our regularized model prediction at times $t+1$, $t+3$ and $t+6$ (time t corresponds to the last input image, it is repeated on each row). The prediction is close to the target for $t+1$,

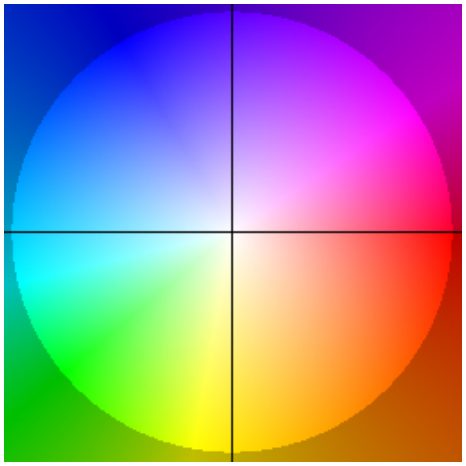


Figure S14.

$t+3$ and starts to move away at time $t+6$. The third row shows the motion flow estimated by the model. Each color in the flow images corresponds to a motion vector (see Figure S14). There is clearly a strong evolving dynamic captured for this sequence. Row 4 is the numerical assimilation model of (Béréziat et al. 2015). It also clearly captures some dynamics and shows interesting patterns, but it tends to diverge when the prediction horizon increases. The ACDNN model (row 5) rapidly produces blurry images; it does not preserve the temperatures and does not seem to capture any dynamics. On row 6 are plotted the predictions of the ConvLSTM model. Temperature is not preserved, as it seems to fade away, and although a dynamic is captured, it does not correspond to the target. Overall, the proposed model seems to forecast SST quite accurately, while retrieving a coherent motion vector field. Additional samples are available at Section 3.2.5.1.

3.2.4.7 On the Generalization in Space and Time

The ability of the model to adapt to other conditions should be evaluated on other regions. We present below complementary experiments aimed at assessing the potential of the proposed model for forecasting SST on sequences distant in time and space from the ones used for training.

Temporal Dimension

In Section 3.2.4, training has been performed on data from 2006 - 2015 and testing on the period 2016-2017. In order to provide some indication of the model behavior on more distant time intervals between train and test data, we have performed experiments using the same regions (17 to 20) as in Section 3.2.4, but using the period 2011 to 2017 for training and period 2006 to 2010 for testing.

Figure S15 shows the MSE curve on this test set, each point corresponding to the mean MSE on predictions performed on 6 days ahead the current date. The most important conclusion is probably that the MSE error remains in the same range for all these years. All the yearly error curve shows a clear seasonal phenomenon with a higher prediction error during summer. A similar behavior has been observed when exchanging train and test data.

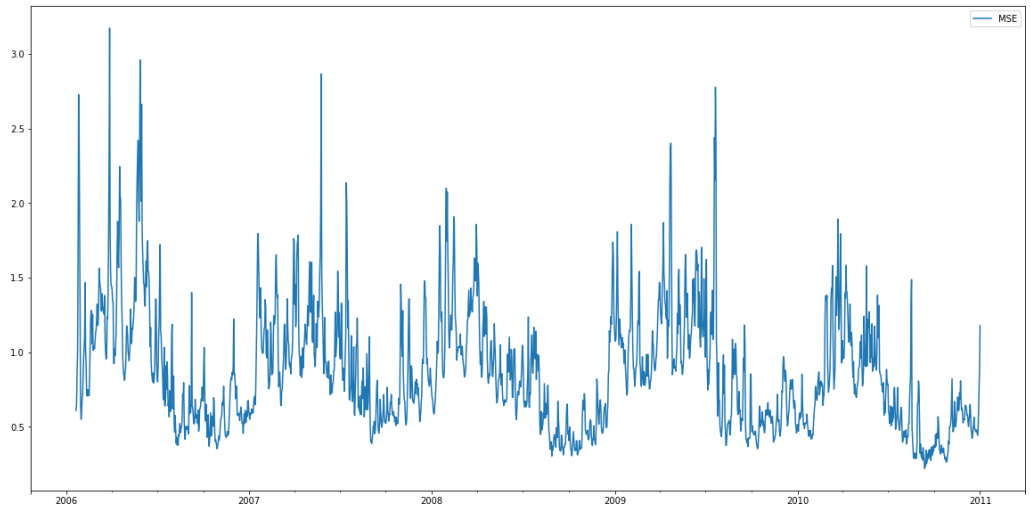


Figure S15. – Evaluation of our model's accuracy in time on data from 2006 to 2010 using data from 2011 to 2017 for training. Regions 17 to 20 were used for both periods. Each day, we produce daily forecasts for 6 days ahead and calculate the associated mean square error. The color of the flow represents the direction of the direction each of the vector as illustrated in Figure S14.

Spatial Dimension

In the experiments, the models have been trained and evaluated on selected regions (numbered 17 to 20 in Figure S11), considered as the most interesting for the observed dynamics.

	Test Regions 17 & 18	Test Regions 8 & 9
Model trained on Regions 17 & 18	1.43	1.22
Model trained on Regions 8 & 9	1.90	1.19

Table S5. – Evaluation of our model's spatial generalization ability. We train our model on two distinct regions and calculate the MSE on both regions for each trained model.

We describe below some results providing indications on how the model performs on regions different from the training ones. For these experiments, the

model has been trained on regions 17 and 18 in Figure S11 and tested on two other regions (regions 8 and 9), and vice versa (trained on 8 and 9 and tested on 17 and 18). The two couples of regions have been selected so as to have different latitude and longitude coordinates. The underlying physical processes generating the data are known to be different in these regions: the overall motion in regions 17 and 18 is greater, and the difference between extreme temperatures is larger, compared to regions 8 and 9. Experimental conditions are similar to the one described in section Section 3.2.4.2, i.e. 2006-2015 have been used for training and 2016-1017 for testing.

Results in Table S5 show that the model generalizes reasonably well to unseen data from distant spatial regions, with a slight decrease in performance when training and test regions do not correspond. The performance loss is 0.47 for regions (17, 18) which show a strong dynamic, whereas it is only 0.03 for regions (8, 9) for which the dynamics are more stable. Most notably, MSE performance depends more on the region itself than on the train/ test conditions. Error is always higher in regions with strong dynamics (17, 18) than on more stable regions (8, 9) whatever the train/ test conditions are. Note that to further improve the results on distant data, it is possible to fine-tune the model using data from the studied regions.

3.2.5 Conclusion

By using as an example application a relatively complex problem concerning ocean dynamics, we proposed a principled way to design Deep Learning models using inspiration from the physics. The proposed approach can be easily generalized to a class of problems for which the underlying dynamics follow advection-diffusion principles. We have compared the proposed approach to a series of baselines. It is able to reach performance comparable to a state-of-the-art numerical model and clearly outperform alternative NN models used as baselines.

3.2.5.1 Additional Samples

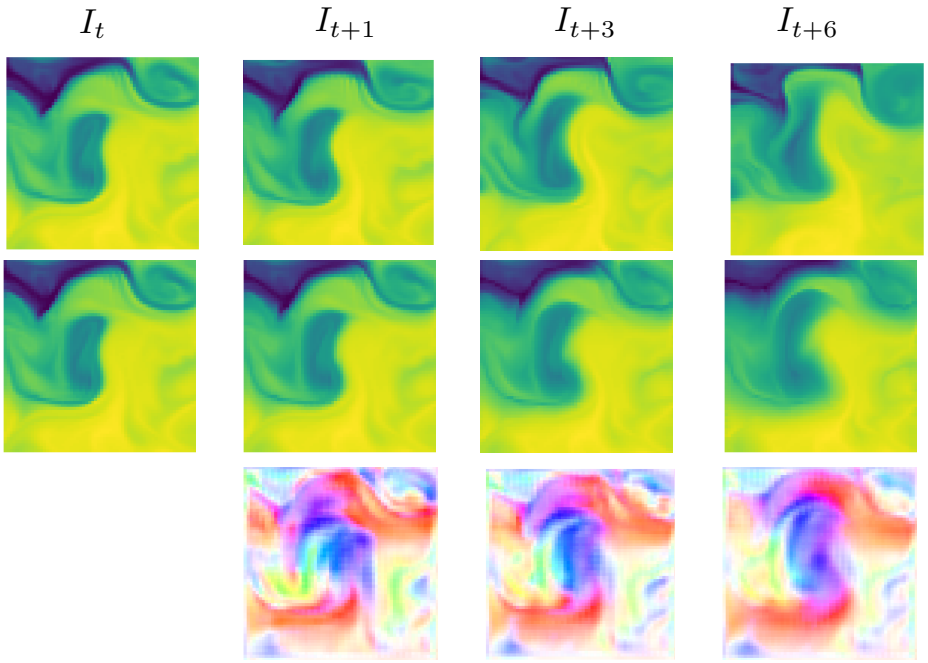


Figure S16. – Output for the 6 of May to the 9 of May 2016, Output , From top to bottom: target, our model prediction, our model flow. The color of the flow represents the direction of each of the vector as illustrated in Figure S14.

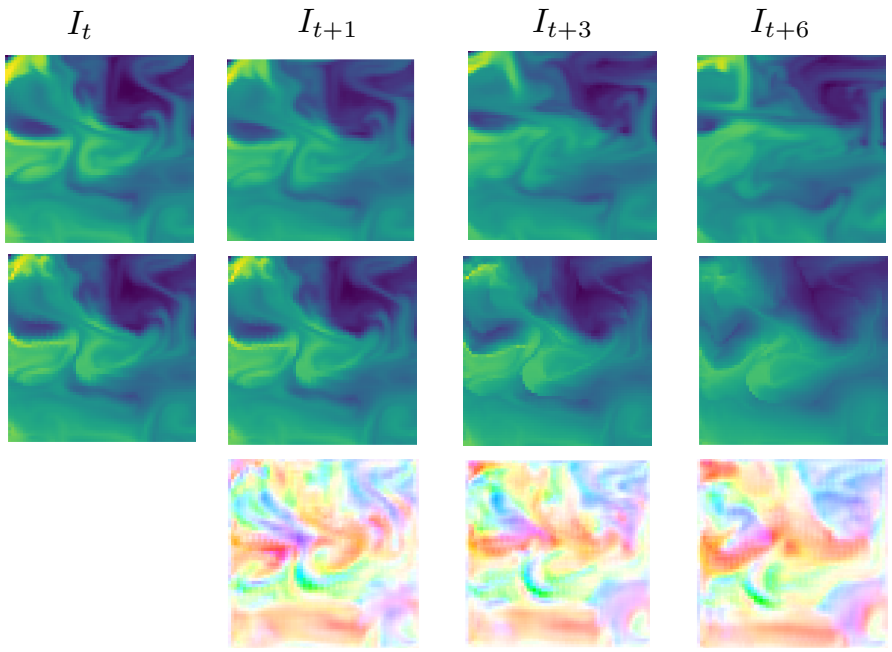


Figure S17. – Output for the 6 of January to the 9 of January 2016. From top to bottom: target, our model prediction, our model flow. The color of the flow represents the direction of each of the vector as illustrated in Figure S14.

3.3 Incorporating Imperfect Knowledge

abstract

Forecasting complex dynamical phenomena in settings where only partial knowledge of their dynamics is available is a prevalent problem across various scientific fields. While purely data-driven approaches are arguably insufficient in this context, standard physical modeling based approaches tend to be over-simplistic, inducing non-negligible errors. In this work, 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 phenomena, i.e. reaction-diffusion equations, wave equations and the non-linear damped pendulum, 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 in this section has led to the publication of a conference paper:

Yuan Yin, Vincent LE GUEN, Jérémie DONA, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas THOME, and patrick gallinari (2021b). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=kmG8vRXTFv

3.3.1 Introduction

Modeling and forecasting complex dynamical systems is a major challenge in domains such as environment and climate (Rolnick et al. 2019), health science (E. Choi et al. 2016), and in many industrial applications (Toubeau et al. 2018). 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 tasks (Shi et al. 2015a; Yunbo Wang et al. 2018b; Oreshkin et al. 2019; Donà et al. 2020), and connections have been drawn between deep architectures and numerical ODE solvers, *e.g.* neural ODEs (T. Q. Chen et al. 2018c; Ayed et al. 2019b). However, modeling complex physical dynamics is still beyond the scope of pure ML methods, which often cannot properly extrapolate to new conditions as MB approaches do.

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

The large majority of aforementioned MB/ML hybrid approaches assume that the physical model adequately describes the observed dynamics. This assumption is, however, commonly violated in practice. This may be due to various factors, *e.g.* idealized assumptions and difficulty to explain processes from first principles (Gentine et al. 2018), computational constraints prescribing a fine grain modeling of the system (Ayed et al. 2019c), unknown external factors, forces and sources which are present (Large et al. 2004). In this paper, 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 S18 on the pendulum problem, is to *augment* the physical model when—and only when—it falls short.

Designing a general method for combining MB and ML approaches is still a widely open problem, and a clear problem formulation for the latter is lacking (Reichstein et al. 2019). Our contributions 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 3.3.3.1) for the decomposition given mild conditions, and show that this formulation ensures interpretability and benefits generalization.

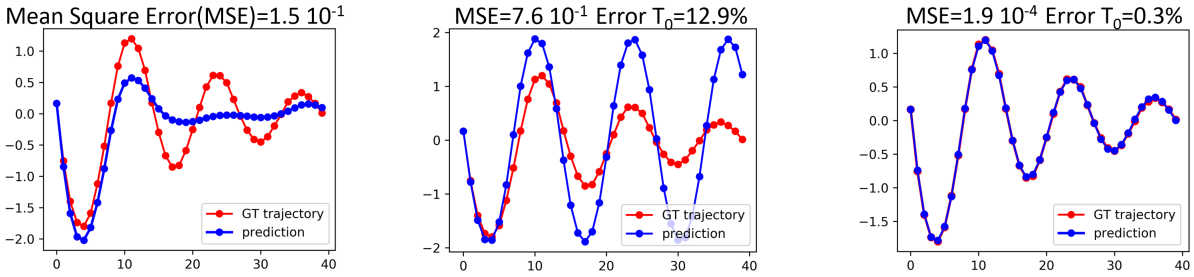
- We propose a trajectory-based training formulation (Section 3.3.3.2) along with an adaptive optimization scheme (Section 3.3.3.3) enabling end-to-end learning for both physical and deep learning components. This allows APHYNITY to *automatically* adjust the complexity of the neural network to different 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 3.3.4). 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 partially observable setting (see discussion in Section 3.3.5).

3.3.2 Related Work

Correction in data assimilation Prediction under approximate physical models has been tackled by traditional statistical calibration techniques, which often rely on Bayesian methods (Pernot et al. 2017). Data assimilation techniques, *e.g.* the Kalman filter (Kalman 1960; Becker et al. 2019), 4D-var (Courtier et al. 1994), prediction errors are modeled probabilistically and a correction using observed data is applied after each prediction step. Similar residual correction procedures are commonly used in robotics and optimal control (W.-H. Chen 2004; S. Li et al. 2014). 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 (Janner et al. 2019) or by model predictive control (Nagabandi et al. 2018). The originality of our approach is to leverage model-based prior knowledge by augmenting it with neurally parametrized dynamics. It does so while ensuring optimal cooperation between the prior model and the augmentation.

Interplay between Machine Learning and Dynamical Systems Combining physical models with machine learning (*gray-box or hybrid* modeling) was first explored from the 1990's: (Psichogios et al. 1992; Thompson et al. 1994; Rico-Martinez et al. 1994) 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



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

Figure S18. – 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 (T. Q. Chen et al. 2018c) 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).

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 (Z. Long et al. 2018c), as discussed in the introduction. Recently, some approaches have specifically targeted the MB/ML cooperation. HybridNet (Y. Long et al. 2018) and PhICNet (Saha et al. 2020) 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. (Q. Wang et al. 2019; Mehta et al. 2020) 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 our proposed learning scheme in terms of forecasting and parameter identification performances (see experiments in Section 3.3.4.2).

3.3.3 The APHYNITY Model

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

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = F(X_t) \tag{S18}$$

defined over a finite time interval $[0, T]$, where the state X is either vector-valued, *i.e.* we have $X_t \in \mathcal{R}^d$ for every t, (pendulum equations in Section 3.3.4), or X_t is a d-dimensional vector field over a spatial domain $\Omega \subset \mathcal{R}^k$, with $k \in \{2, 3\}$, *i.e.* $X_t(x) \in \mathcal{R}^d$ for every $(t, x) \in [0, T] \times \Omega$ (reaction-diffusion and wave equations in Section 3.3.4). 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 $\mathcal{R}^d$ 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.

3.3.3.1 Decomposing dynamics into physical and augmented terms

As introduced in Sec. 3.3.1, 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 $\mathcal{D}$.

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:

$$\min_{F_p \in \mathcal{F}_p, F_a \in \mathcal{F}} \|F_a\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{\mathrm{d}X_t}{\mathrm{d}t} = (F_p + F_a)(X_t) \tag{S19}$$

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

A first key question is whether the minimum in eq. S19 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 A.2.2:

Proposition S2 (Existence of a minimizing pair)**.** *If $\mathcal{F}_p$ is a proximinal set*[8]*, there exists a decomposition minimizing eq. S19.*

Proximinality is a mild condition which, as shown through the proof of the proposition, cannot be weakened. It is a property verified by any boundedly compact set. In particular, it is true for closed subsets of finite dimensional spaces. However, if only existence is guaranteed, while forecasts would be expected to be accurate, non-uniqueness of the decomposition would hamper the interpretability of F_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. S19 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 A.2.2:

Proposition S3 (Uniqueness of the minimizing pair)**.** *If $\mathcal{F}_p$ is a Chebyshev set*[1]*, eq. S19 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 S2 and S3 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.

8. 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 A.2.1.

3.3.3.2 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 $F_p^{\theta_a}$. 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 (Raissi et al. 2019b; Ayed et al. 2019b).

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. S19 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 (Brunton et al. 2016; Greydanus et al. 2019; Cranmer et al. 2020). This numerical scheme requires high space and time resolutions in the observation space in order to get reliable gradient estimates. Furthermore it is often unstable, leading to explosive numerical errors as discussed in Appendix A.2.4. We propose instead to solve eq. S19 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):

$$\min_{\theta_p,\theta_a} \quad \left\|F_a^{\theta_a}\right\| \quad \text{subject to} \quad \forall i, \forall k, \widetilde{X}^{(i)}_{k\Delta t} = X^{(i)}_{k\Delta t} \tag{S20}$$

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 MB + ML formulation leads to different parameter estimates than using the MB formulation alone, as analyzed more thoroughly in Appendix A.2.3. 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 3.3.4). 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 A.2.6 also confirms that the integral formulation gives better forecasting results and a more stable behavior than supervising over finite difference approximations of the derivatives.

3.3.3.3 Adaptively constrained optimization

The formulation in eq. S20 involves constraints which are difficult to enforce exactly in practice. We considered a variant of the method of multipliers (Bertsekas 1996) which uses a sequence of Lagrangian relaxations $\mathcal{L}_{\lambda_j}(\theta_p, \theta_a)$:

$$\mathcal{L}_{\lambda_j}(\theta_p, \theta_a) = \|F_a^{\theta_a}\| + \lambda_j \cdot \mathcal{L}_{traj}(\theta_p, \theta_a) \tag{S21}$$

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 2 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 for
  end for
  $\lambda_{j+1} = \lambda_j + \tau_2 \mathcal{L}_{traj}(\theta_{j+1})$
end for

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 eq. S20. 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[9], 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 2. This adaptive iterative procedure allows us to obtain stable and robust results, in a reproducible fashion, as shown in the experiments.

3.3.4 Experimental validation

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

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

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.

3.3.4.1 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) [10]. Implementation details and architectures are given in Appendix A.2.5.

Reaction-diffusion equations We consider a 2D FitzHugh-Nagumo type model (Klaasen et al. 1984). The system is driven by the PDE $\frac{\partial u}{\partial t} = a\Delta u + R_u(u, v; k)$, $\frac{\partial v}{\partial t} = b\Delta v + R_v(u, v)$ where a and b are respectively the diffusion coefficients of u and v, Δ is the Laplace operator. The local reaction terms are $R_u(u, v; k) = u - u^3 - k - v, R_v(u, v) = u - v$. The state is $X = (u, v)$ and is defined over a compact rectangular domain Ω with periodic boundary conditions. The considered physical models are:

- *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:
  $\mathcal{F}_p = \{F_p^{a,b,k} : (u, v) \mapsto (a\Delta u + R_u(u, v; k), b\Delta v + R_v(u, v) \mid a \geq a_{\min} > 0, b \geq b_{\min} > 0, k \geq k_{\min} > 0\}$.

10. 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.

Damped wave equations We investigate the damped-wave PDE: $\frac{\partial^2 w}{\partial t^2} - c^2 \Delta w + k\frac{\partial w}{\partial t} = 0$ where k is the damping coefficient. The state is $X = (w, \frac{\partial w}{\partial t})$ and we consider a compact spatial domain Ω with Neumann homogeneous boundary conditions. Note that this damping differs from the pendulum, as its effect is global. Our physical models are:

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

  $\mathcal{F}_p = \{F_p^c : (u, v) \mapsto (v, c^2 \Delta u) \mid c \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$;

- *Param PDE (c, k)*:

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

Damped pendulum The evolution follows the ODE ${}^{\mathrm{d}^2\theta}/_{\mathrm{d}t^2} + \omega_0^2 \sin\theta + \alpha {}^{\mathrm{d}\theta}/_{\mathrm{d}t} = 0$, where $\theta(t)$ is the angle, ω_0 the proper pulsation (T_0 the period) and α the damping coefficient. With state $X = (\theta, {}^{\mathrm{d}\theta}/_{\mathrm{d}t})$, the ODE is $F_p^{\omega_0,\alpha} : X \mapsto ({}^{\mathrm{d}\theta}/_{\mathrm{d}t}, -\omega_0^2 \sin\theta - \alpha {}^{\mathrm{d}\theta}/_{\mathrm{d}t})$. Our physical models are:

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

  $\mathcal{F}_p = \{F_p^{\mathcal{H}} : (u, v) \mapsto (\partial_y \mathcal{H}(u, v), -\partial_x \mathcal{H}(u, v)) \mid \mathcal{H} \in H^1(\mathcal{R}^2)\}$, $H^1(\mathcal{R}^2)$ is the first order Sobolev space.

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

  $\mathcal{F}_p = \{F_p^{\omega_0,\alpha=0} \mid \omega_0 \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$

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

  $\mathcal{F}_p = \{F_p^{\omega_0,\alpha} \mid \omega_0, \alpha \in [\epsilon, +\infty) \text{ with } \epsilon > 0\}$.

Baselines As purely data-driven baselines, we use Neural ODE (T. Q. Chen et al. 2018c) for the three problems and PredRNN++ (Yunbo Wang et al. (2018b), for reaction-diffusion only) which are competitive models for datasets generated by differential equations and for spatio-temporal data. As MB/ML methods, in the ablations studies (see Appendix A.2.6), we compare for all problems, to the vanilla MB/ML cooperation scheme found in (Q. Wang et al. 2019; Mehta et al. 2020). We also show results for *True PDE/ODE*, which corresponds to the equation for data simulation (which do not lead to zero error due to the difference between simulation and training integration schemes). For the pendulum, we compare to Hamiltonian neural networks (Greydanus et al. 2019; Toth et al. 2020) and to the the deep Galerkin method (DGM, Sirignano et al. (2018)). See additional details in Appendix A.2.5.

Table S6. – Forecasting and identification results on the (a) reaction-diffusion, (b) wave equation, and (c) damped pendulum datasets. We set for (a) $a = 1\times10^{-3}, b = 5\times10^{-3}, k = 5\times10^{-3}$, for (b) $c = 330$, $k = 50$ and for (c) $T_0 = 6$, $\alpha = 0.2$ as true parameters. $\log$ MSEs are computed respectively over 25, 25, and 40 predicted time-steps. %Err param. averages the results when several physical parameters are present. For each level of incorporated physical knowledge, equivalent best results according to a Student t-test are shown in bold. n/a corresponds to non-applicable cases.

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

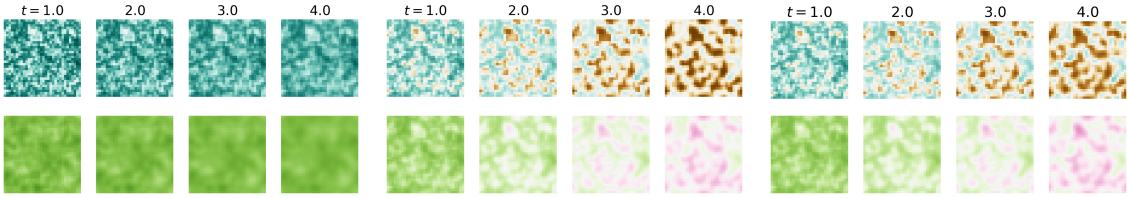
3.3.4.2 Results

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 S6. 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 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 S6. 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 A.2.3, 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 S6. APHYNITY is able to significantly improve physical parameters identification: 2.3% error for the reaction-diffusion, 0.3% for the wave equation, and 4% for the pendulum. This validates the fact that augmenting a simple physical model to compensate its approximations is not only beneficial for prediction, but also helps to limit errors for parameter identification when dynamical models do not fit data well. This is crucial for interpretability and explainability of the estimates.

Ablation study We conduct ablation studies to validate the importance of the APHYNITY augmentation compared to a naive strategy consisting in learning $F = F_p + F_a$ without taking care on the quality of the decomposition, as done in (Q. Wang et al. 2019; Mehta et al. 2020). Results shown in Table S6 of Appendix A.2.6 show a consistent gain of APHYNITY for the three use cases and for all physical models: for instance for *Param ODE* (a, b) in reaction-diffusion, both forecasting performances ($\log$MSE =-5.10 vs. -4.56) and identification parameter (Error= 2.33% vs. 6.39%) improve. Other ablation results are provided in Appendix A.2.6 showing the relevance of the the trajectory-based approach



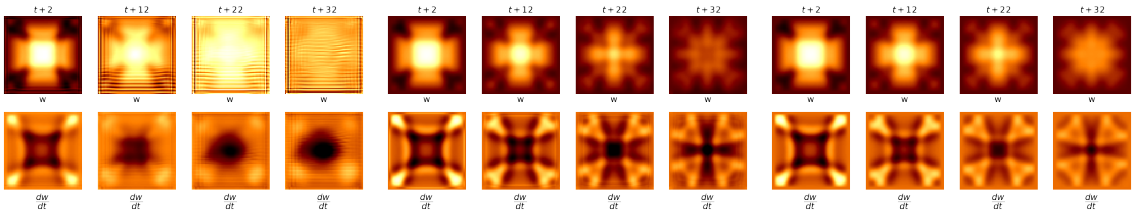
(a) Param PDE (a, b), diffusion-only (b) APHYNITY Param PDE (a, b) (c) Ground truth simulation

Figure S19. – 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 ($t = 2.5$).

described in Section 3.3.3.2 (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 A.2.6). 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 S6 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 A.2.5).

Qualitative visualizations Results in Figure S19 for reaction-diffusion show that the incomplete diffusion parametric PDE in Figure S19(a) is unable to properly match ground truth simulations: the behavior of the two components in Figure S19(a) is reduced to simple independent diffusions due to the lack of interaction terms between u and v. By using APHYNITY in Figure S19(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 prediction quality. In Figure S20, we see for the wave equation that the data-driven Neural ODE model fails at approx-



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

Figure S20. – 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.

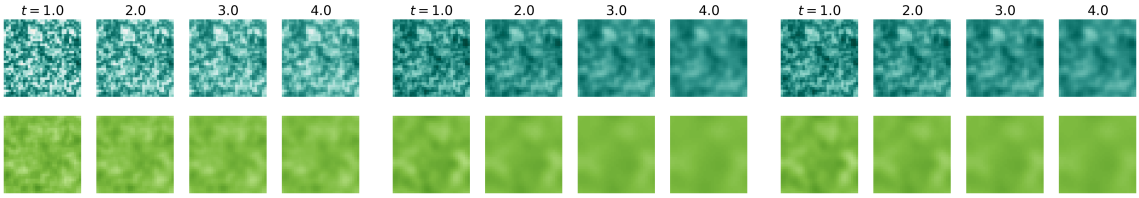
imating $\mathrm{d}w/\mathrm{d}t$ as the forecast horizon increases: it misses crucial details for the second component $\mathrm{d}w/\mathrm{d}t$ which makes the forecast diverge from the ground truth. APHYNITY incorporates a Laplacian term as well as the data-driven F_a thus capturing the damping phenomenon and succeeding in maintaining physically sound results for long term forecasts, unlike Neural ODE.

Extension to non-stationary dynamics We provide additional results in Appendix A.2.7 to tackle datasets where physical parameters of the equations vary in each sequence. To this end, we design an encoder able to perform parameter estimation for each sequence. Results show that APHYNITY accommodates well to this setting, with similar trends as those reported in this section.

Additional illustrations We give further visual illustrations to demonstrate how the estimation of parameters in incomplete physical models is improved with APHYNITY. For the reaction-diffusion equation, we show that the incomplete parametric PDE underestimates both diffusion coefficients. The difference is visually recognizable between the poorly estimated diffusion (Fig. S21(a)) and the true one (Fig. S21(c)) while APHYNITY gives a fairly good estimation of those diffusion parameters as shown in Fig. S21(b).

3.3.5 Conclusion

In this work, we introduce the APHYNITY framework that can efficiently augment approximate physical models with deep data-driven networks, performing similarly to models for which the underlying dynamics are entirely known. We exhibit the superiority of APHYNITY over data-driven, incomplete physics, and state-of-the-art approaches combining ML and MB methods, both in terms of fore-



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

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

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

Figure S21. – 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

casting 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.

An appealing perspective is the applicability of APHYNITY on partially-observable settings, such as video prediction. Besides, we hope that the APHYNITY framework will open up the way to the design of a wide range of more flexible MB/ML models, *e.g.* in climate science, robotics or reinforcement learning. In particular, analyzing the theoretical decomposition properties in a partially-observed setting is an important direction for future work.

3.4 Leveraging the Data From Different Environments

abstract

When modeling dynamical systems from real-world data samples, the distribution of data often changes according to the environment in which they are captured, and the dynamics of the system itself vary from one environment to another. Generalizing across environments thus challenges the conventional frameworks. The classical settings suggest either considering data as i.i.d. and learning a single model to cover all situations or learning environment-specific models. Both are sub-optimal: the former disregards the discrepancies between environments leading to biased solutions, while the latter does not exploit their potential commonalities and is prone to scarcity problems. We propose LEADS, a novel framework that leverages the commonalities and discrepancies among known environments to improve model generalization. This is achieved with a tailored training formulation aiming at capturing common dynamics within a shared model while additional terms capture environment-specific dynamics. We ground our approach in theory, exhibiting a decrease in sample complexity with our approach and corroborate these results empirically, instantiating it for linear dynamics. Moreover, we concretize this framework for neural networks and evaluate it experimentally on representative families of nonlinear dynamics. We show that this new setting can exploit knowledge extracted from environment-dependent data and improves generalization for both known and novel environments.

Yuan Yin, Ibrahim Ayed, Emmanuel de Bézenac, Nicolas Baskiotis, and Patrick Gallinari (2021a). *LEADS: Learning Dynamical Systems that Generalize Across Environments*. arXiv: 2106.04546 [cs.LG]

3.4.1 Introduction

Data-driven approaches offer an interesting alternative and complement to physical-based methods for modeling the dynamics of complex systems and are particularly promising in a wide range of settings: *e.g.* if the underlying dynamics are partially known or understood, if the physical model is incomplete, inaccurate, or fails to adapt to different contexts, if external perturbation sources and forces are not modeled, etc. The idea of deploying ML to model complex dynamical systems picked momentum a few years ago, relying on recent deep learning pro-

gresses and the development of new methods targeting the modeling of temporal and spatio-temporal systems evolution Brunton et al. 2016; Bézenac et al. 2018a; R. T. Q. Chen et al. 2018; Z. Long et al. 2018d; Raissi et al. 2019a; Ayed et al. 2019a; Yin et al. 2021c. It is already being applied in different scientific disciplines (see *e.g.* Willard et al. 2020 for a recent survey) and could help accelerate scientific discovery to address challenging domains such as climate Reichstein et al. 2019 or health Fresca et al. 2020.

However, despite promising results, their success so far is still limited. Current developments usually postulate an idealized setting where data is *abundant* and *the environment does not change*, the so-called "i.i.d. hypothesis". In practice, real-world data may be expensive or difficult to acquire. Moreover, changes in the environment may be caused by many different factors. For example, in climate modeling, there are external forces (*e.g.* Coriolis) which depend on the spatial location Gurvan Madec et al. 2019; or, in health science, parameters need to be personalized for each patient as for cardiac computational models Neic et al. 2017. More generally, data acquisition and modeling are affected by different factors such as geographical position, sensor variability, measuring circumstances, etc. The classical paradigm either considers all the data as i.i.d. and looks for a global model, or proposes specific models for each environment. The former disregards discrepancies between the environments, thus leading to a biased solution with an averaged model which will usually perform poorly. The latter ignores the similarities between environments, thus affecting generalization performance, particularly in settings where per-env. data is limited. This is particularly problematic in dynamical settings, as small changes in initial conditions lead to trajectories not covered by the training data.

In this work, we consider a setting where it is explicitly assumed that the trajectories are collected from different environments. Note that in this setting, the i.i.d. hypothesis is removed twice: by considering the temporality of the data and by the existence of multiple environments. In many useful contexts the dynamics in each environment share similarities, while being distinct which translates into changes in the data distributions. Our objective is to leverage the similarities between environments in order to improve the modeling capacity and generalization performance, while still carefully dealing with the discrepancies across environments. This brings us to consider two research questions:

rq1 Does modeling the differences between environments improve generalization performance compared to classical settings, namely ***One-For-All***, where a unique function is trained for all environments; and ***One-Per-Env.***, where a specific function is fitted for each environment? (*cf.* Sec. 3.4.4 for more details)

rq2 Is it possible to extrapolate to a novel environment that has not been seen during training?

We propose LEarning Across Dynamical Systems (*LEADS*), a novel learning methodology decomposing the learned dynamics into *shared* and *environment-specific* components. The learning problem is formulated in such a way that the *shared* component captures the dynamics common across environments and exploits all the available data, while the *environment-specific* component only models the remaining dynamics, *i.e.* those that cannot be expressed by the former, based on environment-specific data. We show, under mild conditions, that the learning problem is well-posed, as the resulting decomposition exists and is unique (Sec. 3.4.2.2). We then analyze the properties of this decomposition from a sample complexity perspective. While, in general, the bounds might be too loose to be practical, a more precise study is conducted in the case of linear dynamics for which theory and practice are closer. We then instantiate this framework for more general hypothesis spaces and dynamics leading to a heuristic for the control of generalization that will be validated experimentally. Overall, we show that this framework provides better generalization properties than *One-Per-Env.*, requiring less training data to reach the same performance level (*RQ1*). The shared information is also useful to extrapolate to unknown environments: the new function for this environment can be learned from very little data (*RQ2*). We experiment with these ideas on three representative cases (Sec. 3.4.4) where the dynamics are provided by differential equations: ODEs with the Lotka-Volterra predator-prey model, and PDEs with the Gray-Scott reaction-diffusion and the more challenging incompressible Navier-Stokes equations. Experimental evidence confirms the intuition and the theoretical findings: with a similar amount of data, the approach drastically outperforms *One-For-All* and *One-Per-Env.* settings, especially in low data regimes. Code is provided in the supplemental material. Up to our knowledge, this is the first time it is addressed from an ML viewpoint.

3.4.2 Approach

3.4.2.1 Problem setting

We consider the problem of learning models of dynamical physical processes with data acquired from a set of environments E. Throughout the paper, we will assume that the dynamics in an environment $e \in E$ are defined through the evolution of differential equations. This will provide in particular a clear setup for the experiments and the validation. For a given problem, we consider that the dynamics of the different environments share common factors while each environments has its own specificity, resulting in a distinct model per environment. Both the general form of the differential equations and the specific terms of each environment are assumed to be completely unknown. x_t^e denotes the state

of the equation for environment e, taking its values from a bounded set $\mathcal{A}$, with evolution term $f_e : \mathcal{A} \to T\mathcal{A}$, $T\mathcal{A}$ being the tangent bundle of $\mathcal{A}$. In other words, over a fixed time interval $[0, T]$, we have:

$$\frac{\mathrm{d}x_t^e}{\mathrm{d}t} = f_e(x_t^e) \tag{S22}$$

We assume that, for any e, f_e lies in a functional vector space $\mathcal{F}$. In the experiments, we will consider one ODE, in which case $\mathcal{A} \subset \mathcal{R}^d$, and two PDEs, in which case $\mathcal{A}$ is a d'-dimensional vector field over a bounded spatial domain $S \subset \mathcal{R}^{d'}$. The term of the data-generating dynamical system in Eq. S22 is sampled from a distribution for each e, *i.e.* $f_e \sim Q$. From f_e, we define $\mathcal{T}_e$, the data distribution of trajectories $x_\cdot^e$ verifying Eq. S22, induced by a distribution of initial states $x_0^e \sim P_0$. The data for this environment is then composed of l trajectories sampled from $\mathcal{T}_e$, and is denoted as $\hat{\mathcal{T}}_e$ with $x_\cdot^{e,i}$ the i-th trajectory. We will denote the full dataset by $\hat{\mathcal{T}} = \bigcup_{e \in E} \hat{\mathcal{T}}_e$.

The classical empirical risk minimization (ERM) framework suggests to model the data dynamics either at the global level (*One-For-All*), taking trajectories indiscriminately from $\hat{\mathcal{T}}$, or at the specific environment level (*One-Per-Env.*), training one model for each $\hat{\mathcal{T}}_e$. Our aim is to formulate a new learning framework with the objective of explicitly considering the existence of different environments to improve the modeling strategy w.r.t. the classical ERM settings.

3.4.2.2 *LEADS* framework

We decompose the dynamics into two components where $f \in \mathcal{F}$ is shared across environments and $g_e \in \mathcal{F}$ is specific to the environment e, so that

$$\forall e \in E, f_e = f + g_e \tag{S23}$$

Since we consider functional vector spaces, this additive hypothesis is not restrictive and such a decomposition always exists. It is also quite natural as a sum of evolution terms can be seen as the sum of the forces acting on the system. Note that the sum of two evolution terms can lead to behaviors very different from those induced by each of those terms. However, learning this decomposition from data defines an ill-posed problem: for any choice of f, there is a $\{g_e\}_{e \in E}$ such that Eq. S23 is verified. A trivial example would be $f = 0$ leading to a solution where each environment is fitted separately.

Our core idea is that f should capture as much of the shared dynamics as is possible, while g_e should focus only on the environment characteristics not captured by f. To formalize this intuition, we introduce $\Omega(g_e)$, a penalization on

g_e, which precise definition will depend on the considered setting. We reformulate the learning objective as the following constrained optimization problem:

$$\min_{f,\{g_e\}_{e\in E}\in\mathcal{F}} \sum_{e\in E}\Omega(g_e) \quad \text{subject to} \quad \forall x^{e,i}\in\hat{\mathcal{T}},\forall t, \frac{\mathrm{d}x_t^{e,i}}{\mathrm{d}t} = (f+g_e)(x_t^{e,i}) \tag{S24}$$

Minimizing Ω aims to reduce g_es' complexity while correctly fitting the dynamics of each environment. This argument will be made formal in the next section. Note that f will be trained on the data from all environments contrary to g_es. A key question is then to determine under which conditions the minimum in Eq. S24 is well-defined. The following proposition provides an answer (proof cf. Sup. A.3.1):

Proposition S4 (Existence and Uniqueness)**.** *Assume Ω is convex, then the existence of a minimal decomposition $f^\star, \{g_e^\star\}_{e\in E}\in\mathcal{F}$ of Eq. S24 is guaranteed. Furthermore, if Ω is strictly convex, this decomposition is unique.*

In practice, we consider the following relaxed formulation of Eq. S24:

$$\min_{f,\{g_e\}_{e\in E}\in\mathcal{F}} \sum_{e\in E}\left(\frac{1}{\lambda}\Omega(g_e) + \sum_{i=1}^{l}\int_0^T \left\|\frac{\mathrm{d}x_t^{e,i}}{\mathrm{d}t} - (f+g_e)(x_\tau^{e,i})\right\|^2 \mathrm{d}t\right) \tag{S25}$$

where f, g_e are taken from a hypothesis space $\hat{\mathcal{F}}$ approximating $\mathcal{F}$. λ is a regularization weight and the integral term constrains the learned $f+g_e$ to follow the observed dynamics. The form of this objective and its effective calculation will be detailed in Sec. 3.4.4.4.

3.4.3 Improving generalization with *LEADS*

Defining an appropriate Ω is crucial for our method. In this section, we show that the generalization error should decrease with the number of environments. While the bounds might be too loose for NNs, our analysis is shown to adequately model the decreasing trend in the linear case, linking both our intuition and our theoretical analysis with empirical evidence. This then allows us to construct an appropriate Ω for NNs.

3.4.3.1 General case

After introducing preliminary notations and definitions, we define the hypothesis spaces associated with our multiple environment framework. Considering a first setting where all environments of interest are present at training time, we

prove an upper-bound of their effective size based on the covering numbers of the approximation spaces. This allows us to quantitatively control the sample complexity of our model, depending on the number of environments m and other quantities that can be considered and optimized in practice. We then consider an extension for learning on a new, unseen environment. The theory used here was inspired by Baxter 2000, originally developed for learning multiple related tasks, and adapted to our dynamical setting.

Definitions. Sample complexity theory is usually defined for supervised contexts, where for a given input x we want to predict some target y. In our setting, we want to learn trajectories $(x_t^e)_{0\leq t\leq T}$ starting from an initial condition x_0. We reformulate this problem and cast it as a standard supervised learning problem: $\mathcal{T}_e$ being the data distribution of trajectories for environment e, as defined in Sec. 3.4.2.1, let us consider a trajectory $x_\cdot^e \sim \mathcal{T}_e$, and time $\tau \sim \mathrm{Unif}([0,T])$; we define system states $x = x_\tau^e \in \mathcal{A}$ as input, and the corresponding values of derivatives $y = f_e(x_\tau^e) \in T\mathcal{A}$ as the associated target. We will denote $\mathcal{P}_e$ the underlying distribution of (x,y), and $\hat{\mathcal{P}}_e$ the associated dataset of size n.

We are searching for $f, g_e : \mathcal{A} \to T\mathcal{A}$ in an approximation function space $\hat{\mathcal{F}}$ of the original space $\mathcal{F}$. Let us define $\hat{\mathcal{G}} \subseteq \hat{\mathcal{F}}$ the effective function space from which the g_es are sampled. Let $f + \hat{\mathcal{G}} := \{f + g : g \in \hat{\mathcal{G}}\}$ be the hypothesis space generated by function pairs (f,g), with a fixed $f \in \hat{\mathcal{F}}$. For any $h : \mathcal{A} \to T\mathcal{A}$, the error on some test distribution $\mathcal{P}_e$ is given by $\mathrm{er}_{\mathcal{P}_e}(h) = \int_{\mathcal{A}\times T\mathcal{A}} \|h(x) - y\|^2 \,\mathrm{d}\mathcal{P}_e(x,y)$ and the error on the training set by $\hat{\mathrm{er}}_{\hat{\mathcal{P}}_e}(h) = \frac{1}{n}\sum_{(x,y)\in\hat{\mathcal{P}}_e} \|h(x) - y\|^2$.

***LEADS* sample complexity.** Let $\mathcal{C}_{\hat{\mathcal{G}}}(\varepsilon, \hat{\mathcal{F}})$ and $\mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}})$ denote the capacity of $\hat{\mathcal{F}}$ and $\hat{\mathcal{G}}$ at a certain scale $\varepsilon > 0$. Such capacity describes the approximation ability of the space. The capacity of a class of functions is defined based on covering numbers, and the precise definition is provided in Sup. A.3.2.2, Table S10. The following result is general and applies for *any* decomposition of the form $f + g_e$. It states that to guarantee a given average test error, the minimal number of samples required is a function of both capacities and the number of envs. m, and it provides a step towards *RQ1* (proof see Sup. A.3.2.2):

Proposition S5. *Given m envs., let $\varepsilon_1, \varepsilon_2, \delta > 0, \varepsilon = \varepsilon_1 + \varepsilon_2$. Assume the number of examples n per env. satisfies*

$$n \geq \max\left\{\frac{64}{\varepsilon^2}\left(\frac{1}{m}\log\frac{4\mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_1}{16}, \hat{\mathcal{F}})}{\delta} + \log \mathcal{C}_{\hat{\mathcal{F}}}\left(\frac{\varepsilon_2}{16}, \hat{\mathcal{G}}\right)\right), \frac{16}{\varepsilon^2}\right\} \tag{S26}$$

Then with probability at least $1 - \delta$ (over the choice of training sets $\{\hat{\mathcal{P}}_e\}$), any learner $(f + g_1, \ldots, f + g_m)$ will satisfy $\frac{1}{m}\sum_{e\in E} \mathrm{er}_{\mathcal{P}_e}(f + g_e) \leq \frac{1}{m}\sum_{e\in E} \hat{\mathrm{er}}_{\hat{\mathcal{P}}_e}(f + g_e) + \varepsilon$.

The contribution of $\hat{\mathcal{F}}$ to the sample complexity decreases as m increases, while that of $\hat{\mathcal{G}}$ remains the same: this is due to the fact that shared functions f have access to the data from all environments, which is not the case for g_e. From this finding, one infers the basis of *LEADS*: when learning from several environments, to control the generalization error through the decomposition $f_e = f + g_e$, *f should account for most of the complexity of f_e while the complexity of g_e should be controlled and minimized*. We then establish an explicit link to our learning problem formulation in Eq. S24. Further in this section, we will show for linear ODEs that the optimization of $\Omega(g_e)$ in Eq. S12 controls the capacity of the effective set $\hat{\mathcal{G}}$ by selecting g_es that are as "simple" as possible.

As a corollary, we show that for a fixed total number of samples in $\hat{\mathcal{T}}$, the sample complexity will decrease as the number of envs. increases. To see this, suppose that we have two situations corresponding to data generated respectively from m and m/b envs. The total sample complexity for each case will be respectively bounded by $O(\log \mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_1}{16}, \hat{\mathcal{F}}) + m \log \mathcal{C}_{\hat{\mathcal{F}}}(\frac{\varepsilon_2}{16}, \hat{\mathcal{G}}))$ and $O(b \log \mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_1}{16}, \hat{\mathcal{F}}) + m \log \mathcal{C}_{\hat{\mathcal{F}}}(\frac{\varepsilon_2}{16}, \hat{\mathcal{G}}))$. The latter being larger than the former, a situation with more envs. presents a clear advantage. This result is confirmed by empirical evidence in Sec. 3.4.4, Fig. S25.

***LEADS* sample complexity for novel environments.** Suppose that problem Eq. S24 has been solved for a set of envs. E, can we use the learned model for a new env. not present in the initial training set (*RQ2*)? Let e' be such a new env. $\mathcal{P}_{e'}$ the trajectory distribution of e', generated from dynamics $f_{e'} \sim Q$, and $\hat{\mathcal{P}}_{e'}$ an associated training set of size n'. The following results show that the number of required examples for reaching a given performance is much lower when training $f + g_{e'}$ with f fixed on this new env. than training another $f' + g_{e'}$ from scratch (proof see Sup. A.3.2.2).

Proposition S6. *For all ε, δ with $0 < \varepsilon, \delta < 1$ if the number of samples n' satisfies*

$$n' \geq \max \left\{ \frac{64}{\varepsilon^2} \log \frac{4\mathcal{C}(\frac{\varepsilon}{16}, f + \hat{\mathcal{G}})}{\delta}, \frac{16}{\varepsilon^2} \right\}, \tag{S27}$$

then with probability at least $1 - \delta$ (over the choice of novel training set $\hat{\mathcal{P}}_{e'}$), any learner $f + g_{e'} \in f + \hat{\mathcal{G}}$ will satisfy $\mathrm{er}_{\mathcal{P}_{e'}}(f + g_{e'}) \leq \hat{\mathrm{er}}_{\hat{\mathcal{P}}_{e'}}(f + g_{e'}) + \varepsilon$.

In Prop. S6 as the capacity of $\hat{\mathcal{F}}$ no longer appears, the number of required samples now depends only on the capacity of $f + \hat{\mathcal{G}}$. This sample complexity is then smaller than learning from scratch $f_{e'} = f + g_{e'}$ as can be seen by comparing with Prop. S5 at $m = 1$.

From the previous propositions, it is clear that the environment-specific functions g_e need to be explicitly controlled. We now introduce a practical way to do that. Let $\omega(r, \varepsilon)$ be a strictly increasing function w.r.t. r such that

$$\log \mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}}) \leq \omega(r, \varepsilon), \quad r = \sup_{g \in \hat{\mathcal{G}}} \Omega(g)$$

Minimizing Ω would reduce r and then the sample complexity of our model by constraining $\hat{\mathcal{G}}$. Our goal is thus to construct such a pair (ω, Ω). In the following, we will first show in Sec. 3.4.3.2, how one can construct a penalization term Ω based on the covering number bound for linear approximators and linear ODEs. We show with a simple use case that the generalization error obtained in practice follows the same trend as the theoretical error bound when the number of environments varies. Inspired by this result, we then propose in Sec. 3.4.3.3 an effective Ω to penalize the complexity of the neural networks g_e.

3.4.3.2 Linear case: theoretical bounds correctly predict the trend of test error

Results in Sec. 3.4.3.1 provide general guidelines for our approach. We now apply them to a linear system to see how the empirical results meet the tendency predicted by theoretical bound.

Let us consider a linear ODE $\frac{\mathrm{d}x_t^e}{\mathrm{d}t} = \mathrm{L}_{\beta F_e}(x_t^e)$ where $\mathrm{L}_{\beta F_e} : x \mapsto \beta F_e x$ is a linear transformation associated to the square real valued matrix $\beta F_e \in M_{d,d}(\mathcal{R})$. We choose as hypothesis space the space of linear functions $\hat{\mathcal{F}} \subset \mathcal{L}(\mathcal{R}^d, \mathcal{R}^d)$ and instantiate a linear *LEADS* $\frac{\mathrm{d}x_t^e}{\mathrm{d}t} = (\mathrm{L}_{\beta F} + \mathrm{L}_{\beta G_e})(x_t^e)$, $\mathrm{L}_{\beta F} \in \hat{\mathcal{F}}, \mathrm{L}_{\beta G_e} \in \hat{\mathcal{G}} \subseteq \hat{\mathcal{F}}$. As suggested in Bartlett et al. 2017, we have that (proof in Sup. A.3.2.3):

Proposition S7. *If for all linear maps $\mathrm{L}_{\beta G_e} \in \hat{\mathcal{G}}$, $\|\beta G\|_F^2 \leq r$, if the input space is bounded s.t. $\|x\|_2 \leq b$, and the MSE loss function is bounded by c, then*

$$\log \mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}}) \leq \lceil rcd(2b)^2/\varepsilon^2 \rceil \log 2d^2 =: \omega(r, \varepsilon)$$

$\omega(r, \varepsilon)$ is a strictly increasing function w.r.t. r. This indicates that we can choose $\Omega(\mathrm{L}_{\beta G}) = \|\beta G\|_F$ as our optimization objective in Eq. S24. The sample complexity in Eq. S26 will decrease with the size the largest possible $r = \sup_{\mathrm{L}_{\beta G} \in \hat{\mathcal{G}}} \Omega(\mathrm{L}_{\beta G})$. The optimization process will reduce $\Omega(\mathrm{L}_{\beta G})$ until a minimum is reached. The maximum size of the effective hypothesis space is then bounded and decreases throughout training thanks to the penalty. Then in linear case Prop. 2 becomes (proof *cf.* Sup. A.3.2.3):

Proposition S8. *If for linear maps $\mathrm{L}_{\beta F} \in \hat{\mathcal{F}}$, $\|\beta F\|_F^2 \leq r'$, $\mathrm{L}_{\beta G} \in \hat{\mathcal{G}}$, $\|\beta G\|_F^2 \leq r$, $\|x\|_2 \leq b$, and if the MSE loss function is bounded by c, given m envs. and n samples per env., with*

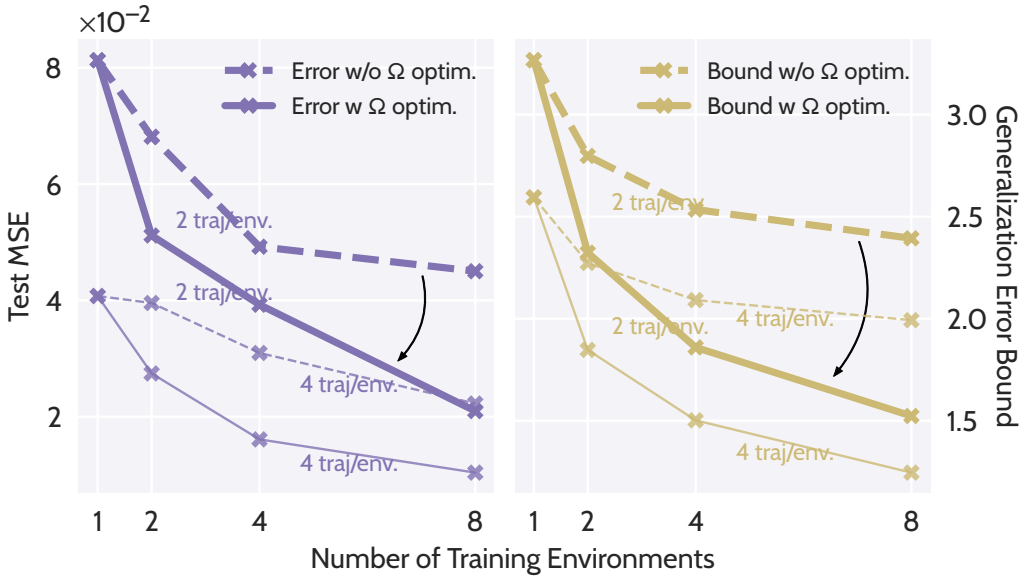


Figure S22. – Test error compared with corresponding theoretical bound. The arrows indicate the changes after applying $\Omega(g_e)$ penalty.

the probability $1-\delta$*, the generalization error upper bound is* $\varepsilon = \max\left\{\sqrt{(p+\sqrt{p^2+4q})/2}, \sqrt{16/n}\right\}$ *where* $p = \frac{64}{mn}\log\frac{4}{\delta}$ *and* $q = \frac{64}{n}\left\lceil\left(\frac{r'}{mz^2} + \frac{r}{(1-z)^2}\right)cd(32b)^2\right\rceil \log 2d^2$ *for any* $0 < z < 1$.

In Fig. S22, we take an instance of linear ODE defined by $\beta F_e = \beta Q \beta\Lambda_e \beta Q^\top$ with the diagonal $\beta\Lambda_e$ specific to each env. After solving Eq. S24 we have at the optimum that $\beta G_e = \beta F_e - \beta F^\star = \beta F_e - \frac{1}{m}\sum_{e'\in E}\beta F_{e'}$. Then we can take $r = \max_{\{\mathrm{L}_{\beta G_e}\}} \Omega(\mathrm{L}_{\beta G_e})$ as the norm bound of $\hat{\mathcal{G}}$ when $\Omega(g_e)$ is optimized. Fig. S22 shows on the left the test error with and without penalty and the corresponding theoretical bound on the right. We observe that, after applying the penalty Ω, the test error is reduced as well as the theoretical generalization bound, as indicated by the arrows from the dashed line to the concrete one. See Sup. A.3.2.3 for more details on this experiment.

3.4.3.3 Nonlinear case: instantiation for neural nets

Motivated by this finding, we instantiate the general case by choosing an appropriate approximating space $\hat{\mathcal{F}}$ and a penalization function Ω. For $\hat{\mathcal{F}}$, we choose the space of feed-forward neural networks with a fixed architecture. We choose the following penalty function:

$$\Omega(g_e) = \|g_e\|_\infty^2 + \alpha\|g_e\|_{\mathrm{Lip}}^2 \tag{S28}$$

where $\|g\|_\infty = \operatorname{ess\,sup}|g|$ and $\|\cdot\|_{\mathrm{Lip}}$ is the Lipschitz semi-norm, α is a hyperparameter. This is inspired by the existing capacity bound for NNs Haussler 1992 (see

Sup. A.3.2.4 for details). Note that constructing tight generalization bounds for neural networks is still an open research problem Nagarajan et al. 2019; however, it may still yield valuable intuitions and guide algorithm design. This heuristic is tested successfully on three different datasets with different architectures in the experiments (Sec. 3.4.4).

3.4.4 Experiments

Our experiments are conducted on three families of dynamical systems described by three broad classes of differential equations. All exhibit complex and nonlinear dynamics. The first one is an ODE-driven system used for biological system modeling. The second one is a PDE-driven reaction-diffusion model, well-known in chemistry for its variety of spatiotemporal patterns. The third one is the more physically complex Navier-Stokes equation, expressing the physical laws of incompressible Newtonian fluids. To show the general validity of our framework, we will use 3 different NN architectures (MLP, ConvNet, and FNO Z. Li et al. 2021). Each architecture is well-adapted to the corresponding dynamics. This also shows that the framework is valid for a variety of approximating functions

3.4.4.1 Dynamics, environments, and datasets

Lotka-Volterra (LV). This classical model Lotka 1926 is used for describing the dynamics of interaction between a predator and a prey. The dynamics follow the ODE:

$$\mathrm{d}u/\mathrm{d}t = \alpha u - \beta uv, \mathrm{d}v/\mathrm{d}t = \delta uv - \gamma v$$

with u, v the number of prey and predator, $\alpha, \beta, \gamma, \delta > 0$ defining how the two species interact. The system state is $x_t^e = (u_t^e, v_t^e) \in \mathcal{R}_+^2$. The initial conditions u_0^i, v_0^i are sampled from a uniform distribution P_0. We characterize the dynamics by $\theta = (\alpha/\beta, \gamma/\delta) \in \Theta$. An env. e is then defined by parameters θ_e sampled from a uniform distribution over a parameter set Θ. We then sample two sets of env. parameters: one used as training envs. for *RQ1*, the other treated as novel envs. for *RQ2*.

Gray-Scott (GS). This reaction-diffusion model is famous for its complex spatiotemporal behavior given its simple equation formulation Pearson 1993. The governing PDE is:

$$\partial u/\partial t = D_u \Delta u - uv^2 + F(1-u), \partial v/\partial t = D_v \Delta v + uv^2 - (F+k)v$$

where the u, v represent the concentrations of two chemical components in the spatial domain S with periodic boundary conditions, the spatially discretized state at time t is $x_t^e = (u_t^e, v_t^e) \in \mathcal{R}_+^{2\times 32^2}$. D_u, D_v denote the diffusion coefficients respectively for u, v, and are held constant, and F, k are the reaction parameters determining the spatio-temporal patterns of the dynamics Pearson 1993. As for the initial conditions $(u_0, v_0) \sim P_0$, we consider uniform concentrations, with 3 2-by-2 squares fixed at other concentration values and positioned at uniformly sampled positions in S to trigger the reactions. An env. e is defined by its parameters $\theta_e = (F_e, k_e) \in \Theta$. We consider a set of θ_e parameters uniformly sampled from the environment distribution Q on Θ.

Navier-Stokes (NS). We consider the Navier-Stokes PDE for incompressible flows:

$$\partial w/\partial t = -v \cdot \nabla w + \nu \Delta w + \xi \qquad \nabla \cdot v = 0$$

where v is the velocity field, $w = \nabla \times v$ is the vorticity, both v, w lie in a spatial domain S with periodic boundary conditions, ν is the viscosity and ξ is the constant forcing term in the domain S. The discretized state at time t is the vorticity $x_t^e = w_t^e \in \mathcal{R}^{32^2}$. Note that v is already contained in w. We fix $\nu = 10^{-3}$ across the envs. We sample the initial conditions $w_0^e \sim P_0$ as in Z. Li et al. 2021. An env. e is defined by its forcing term $\xi_e \in \Theta_\xi$. We uniformly sampled a set of forcing terms from Q on Θ_ξ.

Datasets. For training, we create two datasets for LV by simulating trajectories of $K = 20$ successive points with temporal resolution $\Delta t = 0.5$. We use the first one as a set of training dynamics to validate the *LEADS* framework. We choose 10 envs. and simulate 8 trajectories (thus corresponding to $n = 8 \cdot K$ data points) per env. for training. We can then easily control the number of data points and envs. in experiments by taking different subsets. The second one is used to validate the improvement with *LEADS* while training on novel envs. We simulate 1 trajectory ($n = 1 \cdot K$ data points) for training. We create two datasets for further validation of *LEADS* with GS and NS. For GS, we simulate trajectories of $K = 10$ steps with $\Delta t = 40$. We choose 3 parameters and simulate 1 trajectory ($n = 1 \cdot K$ data points) for training. For NS, we simulate trajectories of $K = 10$ steps with $\Delta t = 1$. We choose 4 forcing terms and simulate 8 trajectories ($n = 8 \cdot K$ states) for training. For test-time evaluation, we create for each equation in each environment a test set of 32 trajectories ($32 \cdot K$) data points. Note that every environment dataset has the same number of trajectories and the initial conditions are fixed to equal values across the environments to ensure that the data variations only come from the dynamics themselves, *i.e.* for the i-th trajectory in $\hat{\mathcal{P}}_e$, $\forall e, x_0^{e,i} = x_0^i$. LV and GS data are simulated with the DOPRI5 solver in NumPy J. Dormand et al. 1980; Harris

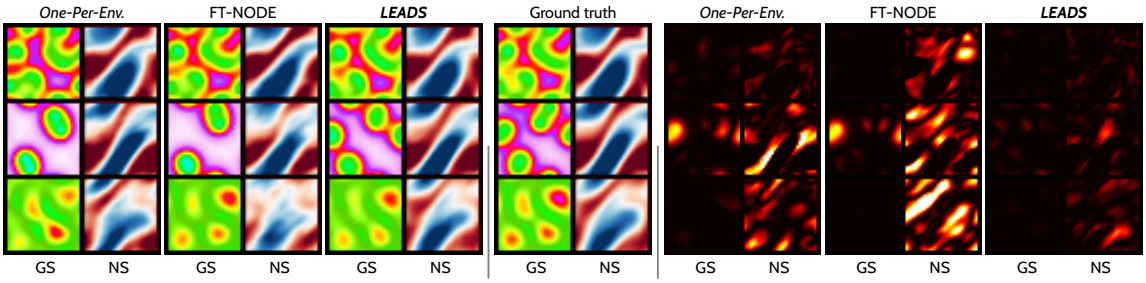


Figure S23. – Left: final states for GS and NS predicted by the two best baselines (*One-Per-Env.* and FT-NODE) and *LEADS* compared with ground truth. Different environment are arranged by row (3 in total). Right: the corresponding MSE error maps; darker is smaller. (See Sup. A.3.4 for full sequences)

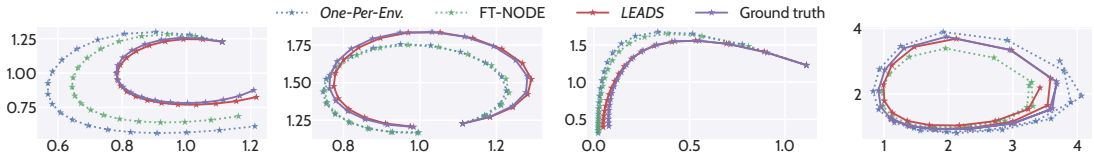


Figure S24. – Test prediction obtained with two baselines (*One-Per-Env.* and FT-ODE) and *LEADS* compared with ground truth for LV in phase space for 4 envs., one per figure from left to right.

et al. 2020. NS data is simulated with the pseudo-spectral method as in Z. Li et al. 2021.

3.4.4.2 Experimental settings and baselines

We validate *LEADS* in two settings: in the first one all the envs. in E are available at once and then f and all the g_es are all trained on E. In the second one, training has been performed on E as before, and we consider a novel env. $e' \notin E$: the shared term f being kept fixed, the approximating function $f_{e'} = f + g_{e'}$ is trained on the data from e' (*i.e.* only $g_{e'}$ is modified).

All environments available at once. We introduce five baselines used for comparing with *LEADS*: (a) ***One-For-All***: learning on the entire dataset $\hat{\mathcal{P}}$ over all envs. with the sum of a pair of NNs $f + g$, with the standard ERM principle, as in Ayed et al. 2019a. Although this is equivalent to use only one function f, we use this formulation to indicate that the number of parameters is the same for this experiment and for the *LEADS* ones. (b) ***One-Per-Env.***: learning a specific function for each dataset $\hat{\mathcal{P}}_e$. For the same reason as above, we keep the sum formulation $(f + g)_e$. (c) Factored Tensor RNN or **FT-RNN** Spieckermann et al.

Table S7. – Results for LV, GS, and NS datasets, trained on m envs. with n data points per env.

Method	LV ($m = 10, n = 1 \cdot K$)		GS ($m = 3, n = 1 \cdot K$)		NS ($m = 4, n = 8 \cdot K$)	
	MSE train	MSE test	MSE train	MSE test	MSE train	MSE test
One-For-All	4.57e-1	5.08±0.56 e-1	1.55e-2	1.43±0.15 e-2	5.17e-2	7.31±5.29 e-2
One-Per-Env.	2.15e-5	7.95±6.96 e-3	8.48e-5	6.43±3.42 e-3	5.60e-6	1.10±0.72 e-2
FT-RNNSpieckermann et al. 2015	5.29e-5	6.40±5.69 e-3	8.44e-6	8.19±3.09 e-3	7.40e-4	5.92±4.00 e-2
FT-ODE	7.74e-5	3.40±2.64 e-3	3.51e-5	3.86±3.36 e-3	1.80e-4	2.96±1.99 e-2
LEADS no min.	3.28e-6	3.07±2.58 e-3	7.65e-5	5.53±3.43 e-3	3.20e-4	7.10±4.24 e-3
LEADS (Ours)	5.74e-6	**1.16±0.99 e-3**	5.75e-5	**2.08±2.88 e-3**	1.03e-4	**5.95±3.65 e-3**

2015: it modifies the recurrent neural network to integrate a one-hot environment code into each linear transformation of the network. Instead of being encoded in a separate function g_e like in *LEADS*, the environment appears here as an extra one-hot input for the RNN linear transformations. This can be implemented for representative SOTA (spatio-)temporal predictors such as GRU Cho et al. 2014 or PredRNN Yunbo Wang et al. 2017. (d) **FT-NODE**: a baseline for which the same environment encoding as FT-RNN is incorporated in a Neural ODE solver R. T. Q. Chen et al. 2018. (e) ***LEADS no min.***: ablation baseline, our proposal without the $\Omega(g_e)$ penalization. A comparison with the different baselines is proposed in Table S7 for the three dynamics. For concision, we provide a selection of results corresponding to 1 training trajectory per env. for LV and GS and 8 for NS. This is the minimal training set size for each dataset. Further experimental results when varying the number of environments from 1 to 8 are provided in Fig. S25 for LV.

Learning on novel environments. We consider the following training schemes with a pre-trained, fixed f: (a) ***Pre-trained-f-Only***: only the pre-trained f is used for prediction; a sanity check to ensure that f cannot predict in any novel env. without further adaptation. (b) ***One-Per-Env.***: training from scratch on $\{\hat{\mathcal{P}}_{e'}\}$ as *One-Per-Env.* in the previous section. (c) ***Pre-trained-f-Plus-Trained-g_e***: we train g on each dataset $\hat{\mathcal{P}}_{e'}$ based on pre-trained f, *i.e.* $f + g_{e'}$, leaving only $g_{e'}$s adjustable. We compare the test error evolution during training for 3 schemes above for a comparison of convergence speed and performance. Results are given in Fig. S26.

3.4.4.3 Experimental results

All environments available at once. We show the results in Table S7. For LV systems, we confirm first that the entire dataset cannot be learned properly with a single model (*One-For-All*) when the number of envs. increases. Comparing with other baselines, our method *LEADS* reduces the test MSE over 85% w.r.t. *One-Per-Env.* and over 60% w.r.t. *LEADS no min.*, we also cut 75% and 50% of error w.r.t. FT-RNN and FT-NODE. Fig. S24 shows samples of predicted trajectories in

Figure S25. – Test error for LV w.r.t. the number of envs. We apply the models in 1 to 8 envs. 4 groups of curves correspond to models trained with 1 to 8 trajectories per env. All groups highlight the same tendencies: increasing *One-For-All*, stable *One-Per-Env.*, and decreasing *LEADS*.

test, *LEADS* follows very closely the ground truth trajectory, while *One-Per-Env.* under-performs in most envs. We observe the same tendency for the GS and NS systems. The error is reduced by: around 2/3 (GS) and 45% (NS) w.r.t. *One-Per-Env.*; over 60% (GS) and 15% (NS) w.r.t. *LEADS no min.*; 75% (GS) and 90% (NS) w.r.t. FT-RNN; 45% (GS) and 80% (NS) w.r.t. FT-NODE. In Fig. S23, the final states obtained with *LEADS* are qualitatively closer to the ground truth. Looking at the error maps on the right, we see that the errors are systematically reduced across all envs. compared to the other baselines. This shows that *LEADS* accumulates less errors through the integration, which suggests that *LEADS* alleviates overfitting.

We have also conducted a larger scale experiment on LV (Fig. S25) to analyze the behavior of the different training approaches as the number of envs. increases. We consider three models *One-For-All*, *One-Per-Env.* and *LEADS*, 1, 2, 4 and 8 environments, and for each such case, we have 4 groups of curves, corresponding to 1, 2, 4 and 8 training trajectories per environment. We summarize the main observations. With *One-For-All* (blue), the error increases as the number of envs. increases: the dynamics for each env. being indeed different, this introduces an increasingly large bias, and thus the data cannot be fit with one single model. The performance of *One-Per-Env.* (in red), for which models are trained independently for each env., is constant as expected when the number of envs. changes. *LEADS* (green) circumvents these issues and shows that the shared characteristics among the envs. can be leveraged so as to improve generalization: it is particularly effective when the number of samples per env. is small. (See Sup. A.3.4 for more details on the experiments and on the results).

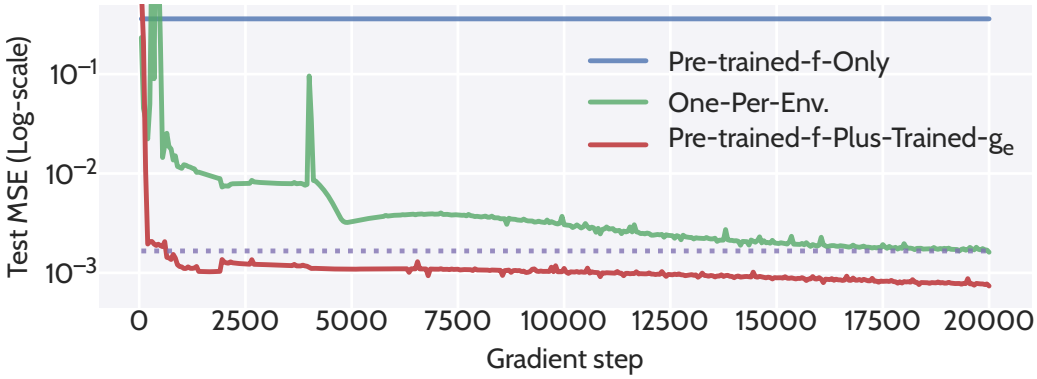


Figure S26. – Test error evolution during training on 2 novel environments for LV.

Learning on novel environments. We demonstrate how the pre-trained dynamics can help to fit a model for novel environments. We took an f pre-trained by *LEADS* on a set of LV envs. Fig. S26 shows the evolution of the test loss during training for three systems: a f function pre-trained by *LEADS* on a set of LV training envs., a g_e function trained from scratch on the new env. and *LEADS* that uses a pre-trained f and learns a g_e residue on this new environment. *Pre-trained-f-Only* alone cannot predict in any novel envs. Very fast in the training stages, *Pre-trained-f-Plus-Trained-g_e* already surpasses the best error of the model trained from scratch (indicated with dotted line). This clearly shows that the learned shared dynamics accelerates and improves the learning in novel envs.

3.4.4.4 Training and implementation details

Discussion on trajectory-based optimization. Solving the learning problem Eq. S23 in our setting, involves computing a trajectory loss (integral term in Eq. S12). However, in practice, we do not have access to the continuous trajectories at every instant t but only to a finite number of snapshots for the state values $\{x_{k\Delta t}\}_{0\leq k\leq \frac{T}{\Delta t}}$ at a temporal resolution Δt. From these discrete observed trajectories, it is still possible to recover an approximate derivative $d^{\Lambda}_{k\Delta t} \simeq \frac{\mathrm{d}x_{k\Delta t}}{\mathrm{d}t}$ using a numerical scheme Λ. The integral term for a given sample in the objective Eq. S12 would then be estimated as $\sum_{k=1}^{K} \left\| d^{\Lambda}_{k\Delta t} - (f+g_e)(x_{\Delta tk}) \right\|^2$. This is not the best solution and we have observed much better prediction performance for all models, including the baselines, when computing the error directly on the states, using an integral formulation $\sum_{k=1}^{K} \|x_{(k+1)\Delta t} - \tilde{x}_{(k+1)\Delta t}\|^2$, where $\tilde{x}_{(k+1)\Delta t}$ is the solution given by a numerical solver approximating the integral $x_{k\Delta t} + \int_{k\Delta t}^{(k+1)\Delta t} (f+g_e)(\tilde{x}_s)\,\mathrm{d}s$ starting from $x_{k\Delta t}$. Comparing directly in the state space yields more accurate results for prediction as the learned network tends to correct the solver's numerical errors, as first highlighted in Yin et al. 2021c.

Calculating Ω. Given finite data and time, the exact infinity norm and Lipschitz norm are both intractable. We opt for more practical forms in the experiments. For the infinity norm, we chose to minimize the empirical norm of the output vectors on known data points, this choice is motivated in Sup. A.3.3. In practice, we found out that dividing the output norm by its input norm works better: $\frac{1}{n}\sum_{i,k}\|g_e(x^{e,i}_{k\Delta t})\|^2/\|x^{e,i}_{k\Delta t}\|^2$, where the $x^{e,i}_{k\Delta t}$ are known states in the training set. For the Lipschitz norm, as suggested in Bietti et al. 2019b, we optimize the sum of the spectral norms of the weight at each layer $\sum_{l=1}^{D}\|W_l^{g_e}\|^2$. We use the power iteration method in Miyato et al. 2018a for fast spectral norm approximation.

Implementation. We used 4-layer MLPs for LV, 4-layer ConvNets for GS and Fourier Neural Operator (FNO) Z. Li et al. 2021 for NS. For FT-RNN baseline, we adapted GRU Cho et al. 2014 for LV and PredRNN Yunbo Wang et al. 2017 for GS and NS. We apply the Swish function Ramachandran et al. 2017 as the default activation function. Networks are integrated in time with RK4 (LV, GS) or Euler (NS), using the basic back-propagation through the internals of the solver. We apply an exponential Scheduled Sampling Lamb et al. 2016 with exponent of 0.99 to stabilize the training. We use the Adam optimizer Kingma et al. 2015 with the same learning rate 10^{-3} and $(\beta_1, \beta_2) = (0.9, 0.999)$ across the experiments. For the hyperparamters in Eq. S28, we chose respectively $\lambda = 5 \times 10^3, 10^2, 10^5$ and $\alpha = 10^{-3}, 10^{-2}, 10^{-5}$ for LV, GS and NS. All experiments are performed with a single NVIDIA Titan Xp GPU on an internal cluster.

3.4.5 Related work

Recent approaches linking invariances to Out-of-Distribution (OoD) Generalization, such as Arjovsky et al. 2020; Krueger et al. 2020; Teney et al. 2020, aim at finding a single classifier that predicts well invariantly across environments with power of extrapolating outside the known distributions. However, in our dynamical systems context, the optimal regression function should be different in each environment, and modeling environment bias is as important as modeling the invariant information, as both are indispensable for prediction. Thus such invariant learners are incompatible with our setting. Meta-learning methods have recently been considered for dynamical systems as in Finn et al. 2017; S. Lee et al. 2021. Their objective is to train a single model that can be quickly adapted to a novel environment with a few data-points in limited training steps. However, in general they result in a single model and there is no special focus on the generalization in training environments, which is the core of our setting. Multi-task learning Yu Zhang et al. 2017 seek for learning shared representations of inputs that exploit the domain information. Current multi-task methods are

not well motivated for dynamical systems due to the lack of consideration of their specificity. Spieckermann et al. 2015 try to directly apply the existing multi-task learning ideas on interactive physical environments. Nonetheless, the approach disregards the specificity of the dynamical systems, and does not guarantee that the common dynamics are totally exploited from data. Other approaches like Yıldız et al. 2019; Norcliffe et al. 2021 integrate some probabilistic methods into a Neural ODE, to learn a distribution of the underlying physical processes. Their focus is principally on the uncertainty of a single model learned with observations from the same dynamics, rather than learning for different ones.

3.4.6 Discussions

Limitations Our framework is generic and could be used in many different contexts. On the theoretical side, the existence and uniqueness properties (Prop. S4) rely on relatively mild conditions covering a large number of situations. The complexity analysis, on the other side, is only practically relevant for simple hypothesis spaces (here linear), and then serves for developing the intuition on more complex spaces (NNs here) where bounds are too loose to be informative. Another limitation is that the theory and experiments consider deterministic systems only: the experimental validation is performed on simulated deterministic data. Note however that this is the case in the vast majority of the ML literature on ODE/PDE spatio-temporal modeling Raissi et al. 2019a; Z. Long et al. 2018a; Z. Li et al. 2021; Yin et al. 2021c. In addition, modeling complex dynamics from real world data is a problem by itself.

Conclusion We introduce *LEADS*, a data-driven framework to learn dynamics from data collected from a set of distinct dynamical systems with commonalities. Experimentally validated with three families of equations, our framework can significantly improve the test performance in every environment w.r.t. classical training, especially when the number of available trajectories is limited. We further show that the dynamics extracted by *LEADS* can boost the learning in similar new environments, which gives us a flexible framework for generalization in novel environments. More generally, we believe that this method is a promising step towards addressing the generalization problem for learning dynamical systems and has the potential to be applied to a large variety of problems.

CHAPTER 4

A DYNAMICAL SYSTEMS VIEWPOINT ON DEEP LEARNING

Contents

4.1 Analysis of CycleGAN **93**
4.1.1 Introduction 93
4.1.2 Desiderata for UDT and Analysis of CycleGAN 95
4.1.3 UDT as Optimal Transport 99
4.1.4 A Residual Instantiation from Dynamical OT 102
4.1.5 Related Work 109
4.1.6 Discussion and Conclusion 109
4.2 Analysis of Classification Networks **111**
4.2.1 Introduction 111
4.2.2 Background 113
4.2.3 General Setting 115
4.2.4 Empirical Analysis of Transport Dynamics in ResNets 117
4.2.5 Least Action Principle for Training Neural Networks 120
4.2.6 Experiments 125
4.2.7 Related Work 127
4.2.8 Discussion and Conclusion 127

4.1 Analysis of CycleGAN

abstract

Unsupervised Domain Translation (UDT) is the problem of finding a meaningful correspondence between two given domains, without explicit pairings between elements of the domains. Following the seminal CycleGAN model, variants and extensions have been used successfully for a wide range of applications. However, although there have been some attempts, they remain poorly understood, and lack theoretical guarantees. In this work, we explore the implicit biases present in current approaches and demonstrate where and why they fail. By expliciting these biases, we show that UDT can be reframed as an Optimal Transport (OT) problem. Using the dynamical formulation of Optimal Transport, this allows us to improve the CycleGAN model into a simple and practical formulation which comes with theoretical guarantees and added robustness. Finally, we show how our improved model behaves on the CelebA dataset in a standard then in a more challenging setting, thus paving the way for new applications of UDT.

4.1.1 Introduction

Given pairs of elements from two different domains, *domain translation* consists in learning a mapping from one domain to another, linking paired elements together. A wide range of problems can be formulated as translation, including image-to-image Isola et al. 2016a, video-to-video T. Wang et al. 2018, image captioning H. Zhang et al. 2016, natural language translation Bahdanau et al. 2015, etc. However, obtaining paired examples is often difficult and for this reason has motivated a growing interest towards the more general *unpaired* or *unsupervised* setting where only samples from both domains are available without pairing. A seminal and influential work for solving Unsupervised Domain Translation (UDT) has been the CycleGAN model J. Zhu et al. 2017. It has spurred many variants and extensions leading to impressive results in several application domains Lample et al. 2018; Yuan et al. 2018; Chung et al. 2018; Y. Choi et al. 2018; Felix et al. 2018.

More formally, Unsupervised Domain Translation (UDT) is the problem of finding, for any element a of a domain $\mathcal{A}$, its best representative b in another given domain $\mathcal{B}$. Both domains are generally provided in the form of a finite number of samples and we will model them here as absolutely continuous probability measures, respectively α and β.

We will make the additional hypothesis that both domain are open and bounded in $\mathcal{R}^d$, with regular boundaries. CycleGAN-like models can then be framed as follows: Given samples from the two probability measures α and β, learn transformations T and S that map one distribution onto the other, while being each other's mutual inverse. This problem thus involves minimizing the following loss:

$$\mathcal{L}(T, S, \mathcal{A}, \mathcal{B}) = \mathcal{L}_{\text{gan}}(T, S, \mathcal{A}, \mathcal{B}) + \mathcal{L}_{\text{cyc}}(T, S, \mathcal{A}, \mathcal{B}) \tag{S1}$$

where $\mathcal{L}_{\text{gan}}$ ensures, at optimality, that [1]

$$T_\sharp \alpha = \beta \quad \text{and} \quad S_\sharp \beta = \alpha$$

while $\mathcal{L}_{\text{cyc}}$ ensures cycle-consistency, namely that both transformations are mutual inverses.

Despite its popularity and empirical successes, there is no clear understanding on why CycleGAN is so effective. As shown in Galanti et al. 2018; C. Yang et al. 2018; Moriakov et al. 2020, the kernel or null space of the CycleGAN loss, *i.e.* the set of couples (T, S) such that $\mathcal{L}(T, S, \mathcal{A}, \mathcal{B}) = 0$, is not reduced to a singleton except in trivial cases and is often infinite in most cases of interest. By studying the kernel of the loss, Moriakov et al. 2020 show more precisely that elements of the null space as well as solutions obtained through the extended version of the loss, where the loss is regularized so that the transformations are close to the identity function, can lead to arbitrarily undesirable solutions of UDT. Thus, there is a discrepancy between what the loss of CycleGAN-like models captures and their practical usefulness. Galanti et al. 2018 postulate that obtained solutions are of minimal *complexity*, a notion related in their work to the minimal number of neural layers necessary to represent a function, and conjecture that mappings of minimal complexity represent a small subset of the CycleGAN's loss kernel. Although their definition of complexity not satisfying and they do not explain why these solutions would correspond to satisfactory ones, this intuition is a valuable one and we build upon it in this work.

More generally, this paper attempts to explain empirically and theoretically why and in which conditions CycleGAN works, and proposing a framework which opens the way for more robust and more flexible CycleGAN models. More precisely,

1. The *push-forward measure* $f_\sharp \rho$ is defined as $f_\sharp \rho(B) = \rho(f^{-1}(B))$, for any measurable set B. Said otherwise, we need T to map α to β and S does the reverse.

- We assess the desiderata ensuring satisfactory results for UDT and conduct an empirical analysis of CycleGAN which shows a systematic implicit bias towards low energy transformations, i.e. transformations that displace the inputs as little as possible, and that this bias not only explains its success, but predicts where it fails.
- Building on this idea, we reformulate the general problem of UDT as an Optimal Transport (OT) problem, thus allowing us to use results from OT theory. This ensures the well-posedness of the problem and regularity of the solution. We are also able solve problems where CycleGAN methods fail.
- We illustrate our findings by proposing a simple instance of the formulation and conducting illustrative experiments. Using the dynamical formulation of OT, our model is more robust, allows for smooth interpolations and halves the number of necessary parameters by providing an inverse mapping for free after training.

4.1.2 Desiderata for UDT and Analysis of CycleGAN

Here, we characterize qualitatively then quantitatively how a good UDT model should behave and show that CycleGAN-like models tend to compute low-energy transformations.

4.1.2.1 What should be the properties of a UDT solution?

Qualitatively, good solutions of a UDT problem are the ones which translate an input a from $\mathcal{A}$ to $\mathcal{B}$ while still conserving as much as possible the characteristics of a, and conversely from $\mathcal{B}$ to $\mathcal{A}$. The CycleGAN seminal paper tries to enforce this through the cycle-consistency loss but, as discussed above and in previous papers, this loss is null for any invertible mapping T by taking the couple (T, T^{-1}), without necessarily conserving any characteristics across domains. In other words, this loss doesn't really add any constraints on the mapping and infinitely many undesirable can still be theoretically recovered by the model.

This intuition has already been formulated in Galanti et al. 2018 in the notion of "semantics preserving mappings". The authors, recognizing that preserving semantics is a vague notion, propose to measure it through the minimal number of layers necessary for neural networks to represent the transformation. However, while we think that it provides a useful step forward in understanding UDT, such a formulation

has several shortcomings: There is no reason why complexity should always be measured as the number of layers of a non-residual NN Chiyuan Zhang et al. 2017 and it is not even clear whether such a minimal number is always finite for relevant transformations; This notion doesn't provide theoretical insights on how and why CycleGAN performs so well in practice or why it seems to work well even with very deep networks; Crucially, there are no guarantees regarding the uniqueness of minimal complexity mappings.

It is also interesting to consider the extended loss for CycleGAN introduced in the original paper J. Zhu et al. 2017 as a regularization forcing T and S to be close from the identity mapping. While, as shown theoretically and empirically in Moriakov et al. 2020, adding this regularization doesn't prevent undesirable mappings to be reached by the model, the fact that it was necessary to further constrain the objective for certain tasks in this way shows that it can be helpful to have transformations which do not transform inputs too much. This is coherent with the view of Galanti et al. 2018. We aim to extend both approaches in a more adaptive, robust and theoretically grounded formalism.

Generalizing those discussions, in our view, there are two main important desirable features in UDT models as used in many practical settings:

- The mapping T (and, symmetrically S) should be constrained to be as conservative as is possible, in the sense that they should be as close to the identity as is possible.

- The mappings T and S should also be regular. Indeed, in the case of image-to-image translation from paintings to photographs for example, if we take two paintings a, a' representing nearly the same scene then we would want the corresponding photos $T(a), T(a')$ to be similar as well. This property would mean that T and S are endowed with some functional regularity, at least a form of continuity.

While the first feature extends the points of view already discussed in previous works, the second one is novel, up to our knowledge. It seems difficult to enforce directly the regularity of the estimated mappings but we show in the following that our approach seamlessly satisfies both properties.

4.1.2.2 CycleGAN is Biased Towards Low Energy Transformations

In practice, the success of CycleGAN models is made possible by the presence of inductive biases that constrain the set of solutions and that are imposed through the combination of the choices made for SGD-based methods, networks architectures, weight parameterization and initialization. In order to develop a better understanding and identify implicit biases, we have conducted an exploratory analysis to characterize the influence of CycleGAN hyperparameters. Our main finding is that the initialization gain σ, *i.e.* the standard deviation of the weights of the residual network (along with a fixed small learning rate), has the most substantial and consistent impact, among all the hyperparameters, on the retrieved mappings. These findings are illustrated in the following experiments.

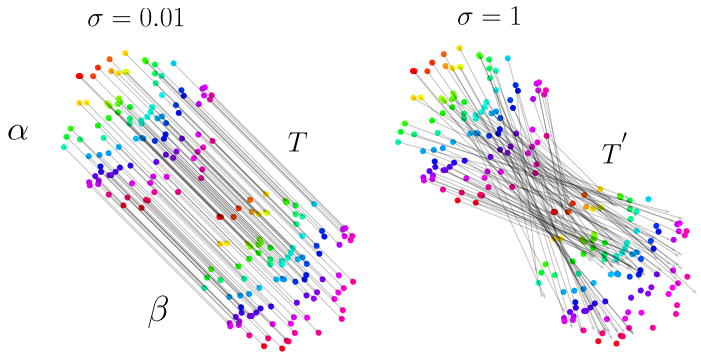


Figure S1. – Pairings between domains obtained with CycleGAN. Both domains correspond to uniform distributions on a 2d-sphere with shifted centers. Small initialization values lead to simple and ordered mappings (Left), whereas larger ones yield complex and disordered ones (Right). Colors highlight original pairing between domains, before shifting.

2D Toy Example Figure S1 shows the effect of changing the gain from a small value, $\sigma = 0.01$, to a higher one, $\sigma = 1$ when learning to map one circular distribution to another. This changes the obtained mapping from a simple translation aligning the two distributions with a minimum displacement to a more disorderly one. In other words, it seems that higher initialization gains lead to higher *energy* mappings. Further quantifying the effect of initialization gain on the retrieved mappings, we use a natural characterisation of *disorder / complexity* of a mapping: the average distance between a sample x from α and its image $T(x)$. Using the squared Euclidean distance, this corresponds to the *kinetic energy* of the displacement and can be written as $\int_{\mathcal{R}^d} \|x -$

$T(x)\|_2^2 \, \mathrm{d}\alpha(x)$. This quantity is also the *quadratic transport cost* used in Optimal Transport Santambrogio 2015. Using the mapping minimizing this cost as a reference, we see, on the left of Figure S2, that the larger σ becomes, the further CycleGAN's mapping (blue curve) is from it. The right curve confirms this finding: As σ grows, so does the transport cost of the trained CycleGAN. For both experiments, the variance across runs increases i.e. the model yields very different mappings across runs, corroborating the ill-posed nature of CycleGAN's optimization problem.

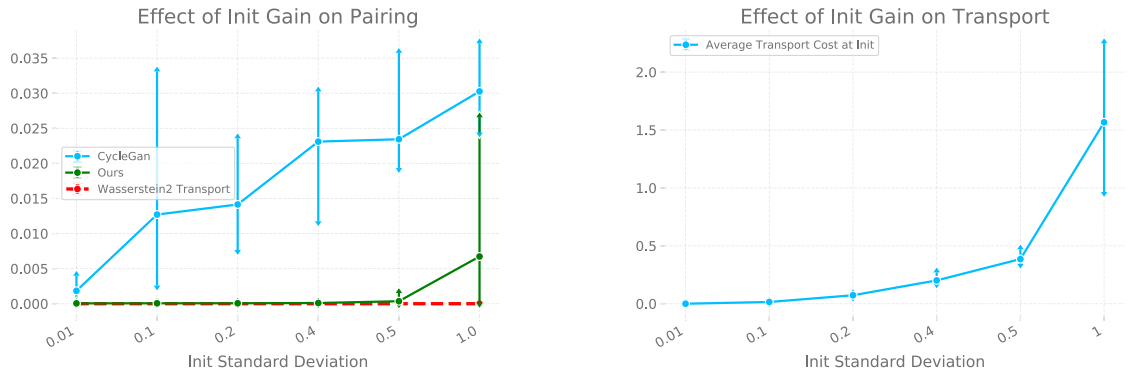


Figure S2. – Left: L^2 distance to the Optimal Transport mapping "*Wasserstein 2 Transport*" as a function of the initialization gain (domains are illustrated in Figure S1). "Ours" refers to the model presented subsequently. Right: Transport cost of the CycleGAN mapping as a function of initialization gain. Metrics are averaged across 5 runs, and the standard deviation is plotted.

High-Dimensional Analysis We also conducted a similar analysis with high-dimensional distributions of images on the CelebA dataset. While in this case calculating the exact OT map is intractable, we can visualize samples obtained with the CycleGAN mapping for different values of σ. The task is male to female transformation where one wants to keep as many characteristics as possible from the original image in the generated one. The result in Figure S5 confirms the low-dimensional findings: while in all cases the distributions have been successfully aligned, as all males are transformed into females, the mappings initialized with low σ values perform a minimal transformation of the input while high σ values produce unwanted changes in the features (hair color, face, skin color,...). This is corroborated by measuring the transportation cost incurred by CycleGAN which goes

from 0.15 for $\sigma = 0.01$ to 9.7 for $\sigma = 1.5$, showing that this behaviour is linked with *high transport costs*.

In summary, for common UDT tasks where the input is to be preserved as much as possible, successful CycleGAN models tend to consistently converge to low energetic mappings and this bias is induced by a small initialization gain. However, the CycleGAN model doesn't give any explicit control over this bias, thus warranting a blind hyperparameter / architecture search for each new task. In the following section, we use OT to define a class of explicitly controllable models with theoretical guarantees.

4.1.3 UDT as Optimal Transport

Using Optimal Transport theory, this section formalizes the findings of the previous one.

4.1.3.1 A (Dynamical) OT Model for UDT

Let us consider the classical Monge problem formulation for OT:

$$\begin{aligned} \underset{T}{\text{minimize}} \quad & \mathcal{C}(T) = \int_{\mathcal{R}^d} c(x, T(x)) \, \mathrm{d}\alpha(x) \\ \text{subject to} \quad & T_\sharp \alpha = \beta \end{aligned} \tag{S2}$$

with the *ground cost* being defined as $c(x, y) = h(x - y)$ with h strictly convex.

Using OT as a way to solve UDT seems very natural as, for most applications, the user's criteria are about preserving input features as much as possible: this is precisely what is given by the OT mapping, its associated cost defining which features are to be preserved. Our idea is that any solution of the Monge problem would be a good candidate for a UDT forward mapping.

Moreover, for a wide range of costs, *e.g.* cost of the form [2] $c(x, y) = \|x - y\|^p$ for $p > 1$, there exists a dynamical point of view of OT equivalent to the Monge formulation [3], similar in intuition and formulation to the equations of fluid dynamics. The general idea is to produce T by using a velocity field v which gradually transports particles from α to β. The

2. A larger family of costs can be considered at the expense of some technicalities, see Alessio Figalli 2008.

3. Which was pioneered in Benamou et al. 2000 and for which a detailed modern presentation is given in chapters 4 and 5 of Santambrogio 2015.

OT map can then be recovered from a path of minimal length, with v solving the optimization problem:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \mathcal{C}^{\text{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p((\phi_t^\cdot)_\sharp \alpha)} \, \mathrm{d}t \\ \text{subject to} \quad & \partial_t \phi_t^x = v_t(\phi_t^x), \\ & \phi_0^\cdot = \mathrm{id}_{\mathcal{A}}, \\ & (\phi_1^\cdot)_\sharp \alpha = \beta \end{aligned} \tag{S3}$$

where the function $\phi_t^\cdot : \mathcal{A} \to \mathcal{R}^d$, induced by the vector field v, is the transport map at time t. This problem can be treated as a continuous-time optimal control problem, and can thus be solved using standard techniques Santambrogio 2015.

We then have, using results from Optimal Transport theory:

Proposition S1 (Existence, Uniqueness and Interpolation)**.** *With the hypothesis already made for α, β and c, eq. S2 admits a unique minimum realized with an invertible map $T^\star$.*

Moreover, for $p > 1$, when $c(x, y) = \|x - y\|^p$, eq. S3 also admits a unique minimal vector field $v^\star$. In addition, we have that the corresponding curve $(\phi_t^\cdot)_\sharp \alpha$ interpolates geodesically between α and β in $\mathbb{W}_p$.

Finally, we have that $T^\star = \phi_1^\cdot$ and we recover the transport cost in the static Monge formulation, i.e. $\mathcal{C}^{dyn}(v^\star) = \mathcal{C}(T^\star)$.

Proof. α, β and c verify the hypothesis for Santambrogio 2015, Theorem 1.17 which gives existence and uniqueness of the OT map $T^\star$. Its invertibility is justified by Remark 1.20 of the same reference.

Taking $\mu_t = (\phi_t^\cdot)_\sharp \alpha$, we have that (μ_t, v_t) solves the continuity equation:

$$\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$$

eq. S3 then becomes a problem of finding the curve of minimal length in $\mathbb{W}_p$ between α and β. This space being a geodesic one, such a curve always exists and is unique. This also justifies the equivalence of eq. S2 and eq. S3 as well as the fact that $T^\star = \phi_1^\cdot$. A more rigorous justification is given in chapters 4 and 5 of Santambrogio 2015 and a few elements are summarized in the Appendix, Section B.1.1. □ □

Here, $\mathbb{W}_p$ is the metric space of absolutely continuous probability measures with finite p-th moment where the distance between two measures μ, ν is defined as the $p-$th root of the OT cost between them. A more in-depth presentation is given in Section B.1.1 of the Appendix.

The key claim of this work, which is supported by the experiments conducted in section 4.1.2.2, is the following one: The OT map T for the quadratic cost behaves very similarly to the solution of UDT approximated by CycleGAN-like models when they behave correctly.

4.1.3.2 Regularity of OT Maps

Let us recall the definition of Hölder continuity: A function $f : \mathcal{X} \to \mathcal{Y}$ is said to be η-Hölder continuous if:

$$\forall x, y \ \|f(x) - f(y)\| \leq M\|x - y\|^{\eta}$$

for $\eta \in]0, 1]$. Moreover, the space of functions whose k-th derivative is η-Hölder continuous is denoted by $C^{k,\eta}(\mathcal{X}, \mathcal{Y})$.

Using the same notation as above and recent results obtained for OT maps X. Ma et al. 2005a, we have the following:

Proposition S2 (Regularity)**.** *$T^\star$ is everywhere differentiable, except on a set of null α measure.*

Additionally, if $T^\star$ does not have singularities, there exists $\eta > 0$ and A, respectively B, relatively closed in $\mathcal{A}$, respectively $\mathcal{B}$, of null Lebesgue measure, such that $T^\star$ is η-Hölder continuous from $\mathcal{A} \setminus A$ to $\mathcal{B} \setminus B$.

Moreover, if the densities of α and β are $C^{k,\eta}$, then $T^\star \in C^{k+1,\eta}(\mathcal{A} \setminus A, \mathcal{B} \setminus B)$.

This notion of regularity is exactly the one that one wants for UDT as the regularity of the mappings has to be linked to that of their underlying domains. Here, the recovered map is even one degree more regular than the domains themselves.

Moreover, the fact that regularity excludes a negligible set of points of the domains is also coherent with what we should expect: In the transported domains, there can be points which are close but nevertheless represent elements from different classes and thus should be transported far from each other. For example, in image-to-image translation between photographs and paintings, two images with the same background can represent different objects and thus be translated into very different paintings. Thus, this regularity result supports our claim for the transport cost to be the right measure of "complexity" for UDT mappings.

4.1.3.3 Computing the Inverse

Consider the optimal vector field of eq. S3 and the following system of differential equations, for all $x \in \mathcal{B}$:

$$\begin{cases} \partial_t \psi_t^x = -v_t^\star(\psi_t^x) \\ \psi_0^x = x \end{cases} \tag{S4}$$

Then we have the following:

Proposition S3. *The solution curve $(\psi_t^\cdot)_t$ of eq. S4 geodesically interpolates between β and α. In particular, $S^\star = \psi_1^\cdot$ is the inverse of $T^\star$, verifies $S^\star_\sharp \beta = \alpha$ and is the OT map between β and α.*

Proof. Let us consider $\nu_t = (\psi_t^\cdot)_\sharp \beta$. Then $(\nu_t, -v_t)$ solves the continuity equation. On the other hand, by a direct calculation and taking previous notations, we have that:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int f \mathrm{d}\mu_{1-t} = - \int \nabla f(x) \cdot v_t(x) \mathrm{d}\mu_{1-t}$$

for any C^1 test function f which means that $(\mu_{1-t}, -v_t)$ also solve the continuity equation with the same initial condition β. This means, by uniqueness, that we have $\nu_t = \mu_{1-t}$ which proves the result. □ □

This result shows that $(T^\star, S^\star)$ does indeed solve the UDT problem and is in the null space of the CycleGAN loss. Moreover, in order to compute $S^\star$, there is no need to parametrize it nor to solve a difficult optimization problem. It is only necessary to discretize the associated differential equation which is of the same nature as the one for the forward mapping, meaning that the same scheme can be used.

4.1.4 A Residual Instantiation from Dynamical OT

This section proposes an instantiation of our model which closely follows the CycleGAN implementation and experiments are conducted to compare both on the CelebA dataset.

4.1.4.1 Linking the Dynamical Formulation with CycleGAN

Let us show that CycleGAN corresponds to a specific implementation of our dynamic formulation with the added transport minimization.

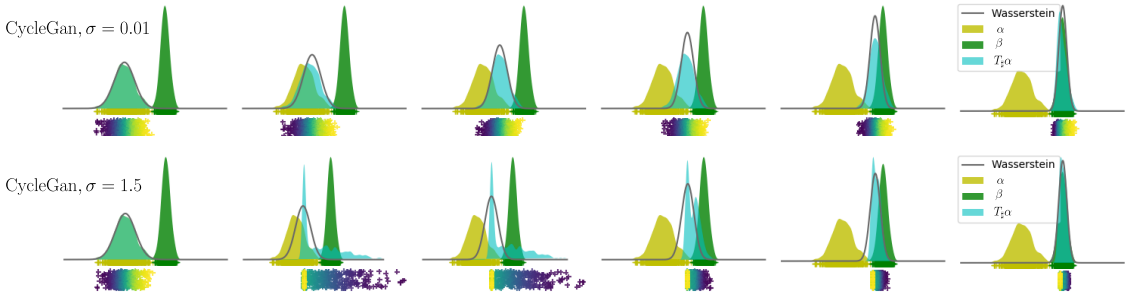


Figure S3. – Visualization of the hidden layers of CycleGAN when mapping the yellow gaussian distribution to the green one with different initializations: As shown by the colored points representing samples under the histograms, when σ increases, the mapping goes from a simple translation (Top) to a more complicated mapping (Bottom), thus inducing an increase in transport cost.

Discretization If v_k corresponds to the residual for layer k of the residual block defined by $\phi_k^x = \phi_{k-1}^x + v_k(\phi_{k-1}^x)$, then, taking the continuous time limit, one recovers the differential equation $\partial_t \phi_t^x = v_t(\phi_t^x)$ Weinan 2017 which appears as a constraint in equation S3. Thus, if we discretize the forward equation in eq. S3 using an Euler numerical scheme, we recover the forward map in the CycleGAN architecture[4].

In CycleGAN, the first boundary condition $\phi_0^\cdot = \mathrm{id}$ is satisfied by construction, while the second $(\phi_1^\cdot)_\sharp \alpha = \beta$ is enforced with the GAN loss.

Thus we recover CycleGAN as a particular implementation of this model when there is no transport cost minimization. We actually construct our instantiation in a similar fashion in order to have meaningful comparisons: The differential equations are discretized using an Euler scheme and boundary conditions are enforced using an iterative penalization of the GAN loss. More involved schemes can be used here such as any suitable parametrized solver T. Q. Chen et al. 2018b.

The fully discretized optimization problem is then the following:

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \mathcal{C}_d(\theta) = \sum_{k=1}^{K} \sum_{x \in \text{Data}_\alpha} \|v^{\theta_k}(\phi_k^x)\|_p^p \\ \text{subject to} \quad & \forall x,\ \forall k,\ \phi_{k+1}^x = \phi_k^x + \Delta t\, v^{\theta_k}(\phi_k^x), \\ & \phi_0^\cdot = \mathrm{id},\ (\phi_1^\cdot)_\sharp \alpha = \beta \end{aligned} \tag{S5}$$

4. Other schemes could be used, which would lead to other architectures, and could arguably be more suited for stability reasons but this is beyond the scope of this work.

Let us also notice that using small initialization gains for the network (See 4.1.2.2) tends to bias the $\|v^\theta\|$s to small values, linking latent trajectories of residual networks with minimal length ones as in Figure S3. It remains to be proven that this fact is indeed stable after training via gradient descent and we consider this to be an interesting problem to analyze in the future.

Enforcing boundary conditions The constraint $(\phi_1^\cdot)_\sharp \alpha = \beta$ ensuring that input domain α maps to the target domain β isn't straightforward to implement. We do so by optimizing an iterative Lagrangian relaxation associated to eq. S5, introducing a measure of discrepancy D between output and target domains:

$$\underset{\theta}{\text{minimize}} \quad \mathcal{C}_d(\theta) + \frac{1}{\lambda_i} D((\phi_1^\cdot)_\sharp \alpha,\, \beta) \tag{S6}$$

where the sequence of Lagrange multipliers $(\lambda_i)_i$ converges linearly to 0 during optimization. At the limit, as the sequence of multipliers converges to 0, the constraint is satisfied.

Each λ_i induces an optimization problem which is solved using stochastic gradient based techniques. As in most approaches for UDT, D may be implemented using generative adversarial networks, or any other appropriate measure of discrepancy between measures, such as kernel distances. Moreover, in order to stabilize the adversarial training which enforces boundary conditions for both our model and CycleGAN, we use an auto-encoder to a lower dimensional latent space. This limits the sharpness of output images but produces consistent and reproducible results, thus allowing meaningful comparisons which is the objective here.

Algorithm Training is done only for the forward equation and the reverse is obtained by iterating $y_{k-1} = y_k - \Delta t\, v^{\theta_k}(y_k)$, starting from a sample y_K from β, as Section 4.1.3.3 allows to. Algorithm 1 gives all necessary details of the procedure.

Architectures. Implementation is performed via DCGAN and ResNet architectures as described below. For the Encoder, we use a standard DCGAN architecture [5], augmenting it with 2 self-attention layers, mapping the images to a fixed, 128 dimensional latent vector. For the Decoder, we use residual up-convolutions, also augmented with 2 self-

5. https://github.com/pytorch/examples/tree/master/dcgan

Algorithm 3 Training Procedure

Input: Dataset of unpaired images $(I_{\mathcal{A}}, I_{\mathcal{B}})$ sampled from (α, β),
initial coefficient λ_0, decay parameter d, initial parameters θ, minimal penalization ϵ

Pretrain Encoder E and decoder D

Make dataset of encodings $(x = E(I_{\mathcal{A}}), y = E(I_{\mathcal{B}}))$

for $i = 1, \ldots, M$ **do**

Randomly sample a mini-batch of x, y

Solve forward equation $\phi^x_{k+1} = \phi^x_k + \Delta t\, v^{\theta_k}(\phi^x_k)$, starting from $\phi^x_0 = x$

Estimate loss $\mathcal{L} = \mathcal{C}_d(\theta) + \frac{1}{\lambda_i} D((\phi^{\cdot}_1)_\sharp \alpha,\, \beta)$ on mini-batch

Compute gradient $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\theta}$ backpropagating through forward equation

Update θ in the steepest descent direction

$\lambda_{i+1} \leftarrow \max(\lambda_i - d, \epsilon)$

end for

Output: Learned parameters θ.

attention layers. We use 9 temporal steps, corresponding to as many residual blocks which consist of a linear layer, batch normalization, a non-linearity, and a final linear layer. The discrepancy D is implemented using generative adversarial networks with the discriminator being a simple MLP architecture of depth 3, consisting of linear layers with spectral normalization, and LeakyReLU($p = 0.2$).

Moreover, in the experiments below, our dataset is the CelebA dataset, resizing images to 128×128 pixels, without any additional transformation. The initial coefficient is $\lambda_0 = 1$, and the decay factor is set depending on the number of total iterations M, so as to be ϵ on the final iteration. Throughout all the experiments, we use the Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$.

4.1.4.2 A Typical UDT Task

Taking the CelebA dataset, we consider the male to female task where the objective is to change the gender of the input image while keeping other characteristics of the image unchanged as much as possible.

Figure S4 illustrates how our model works for Male to Female translation (forward) and back (reverse) on the CelebA dataset, displaying intermediate images as the input distribution gradually trans-

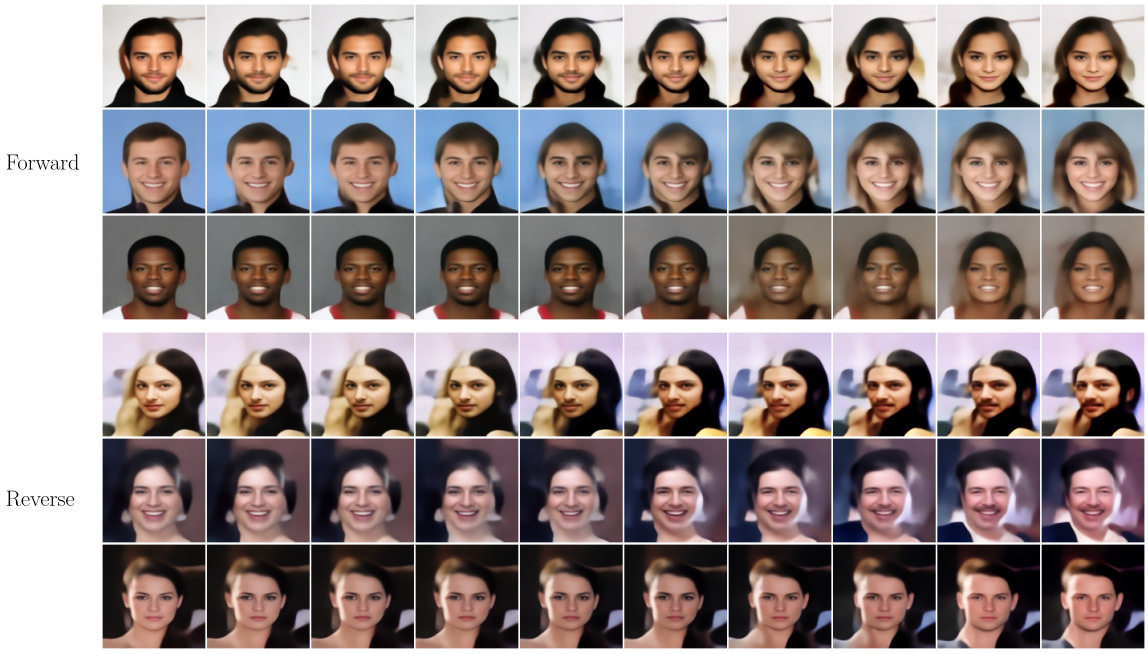


Figure S4. – Male to Female translation (top) and the inverse (bottom). Intermediate images are the interpolations provided by the network's intermediate layers. The reverse mapping is not trained. Additional samples are available in the Appendix, B.1.4.

forms into the target distribution. **No cycle-consistency is being explicitly enforced** here and the **reverse is not directly parametrized nor trained** but still performs well. The model changes relevant high-level attributes when progressively aligning the distributions but doesn't change non-relevant features (hair or skin color, background,...) which is coherent to what is expected for an optimal map w.r.t. an attractive cost function (here the squared Euclidean one). Additional samples are available in Section B.1.4 of the Appendix. All the experiments conducted in this work with our proposed OT framework have been implemented using this dynamical formulation.

Figure S5 shows that for a low initialization gain, both our method and CycleGAN give satisfying and similar solutions. When changing the value of this hyper-parameter, the CycleGAN mapping becomes unstable, producing outputs very different from the inputs.

The non-uniqueness of the solution of CycleGAN's optimization problem is highlighted here by the multiple mappings found for different initializations. It is also worth noting that, for CycleGAN, using a large σ made convergence of the optimization harder. As already observed before, the chaotic behavior of the CycleGAN model correlates with an increase in the transport cost of the obtained mappings. This validates

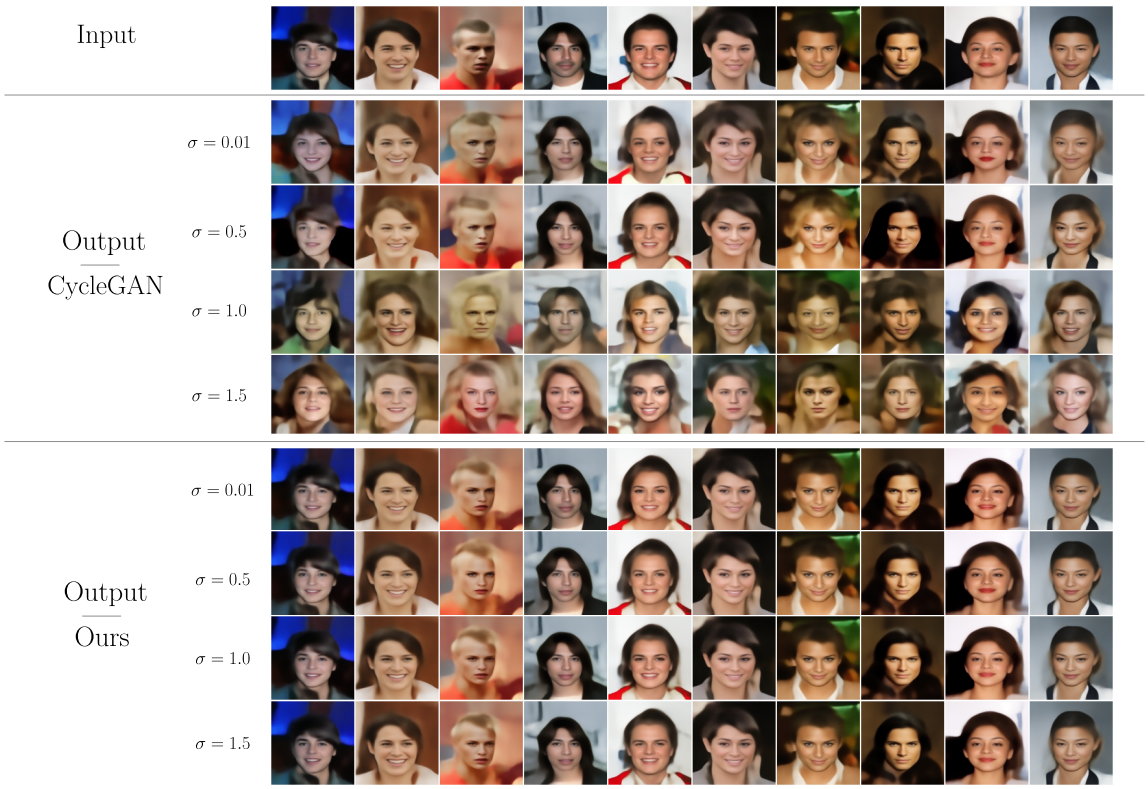


Figure S5. – Each column associates one input image to its outputs for different models: CycleGAN and our model with different initial gain parameters. We have ensured convergence of all models to the same fit to the target distribution.

the L^2 OT bias of CycleGAN, showing that this model only works as an implicit OT mapping for a quadratic cost given a certain architecture, initialized and trained in a certain way. For this example, the prior induced by the quadratic transport cost is the right one and correctly captures the geometry of the task, as one wants to preserve as much as possible the characteristics of the input. By explicitly enforcing optimality w.r.t. the quadratic cost, the model becomes robust to changes in the initialization as the OT problem admits a unique solution for this cost.

4.1.4.3 Imbalanced CelebA Task

Here, we tackle the case of a corrupted dataset where **structural bias** is present in the target domain, which can be an important use case of UDT when fairness of the datasets is an issue Dwork et al. 2012: samples from the target dataset are systematically corrupted. We consider a subset of the CelebA dataset, where domains correspond, respectively,

Figure S6. – Results for Imbalanced CelebA Task. We wish to map faces that have the **Non-Smiling** and **Black Hair** attributes to **Smiling**, **Black-Hair** faces, while only accessing **Smiling**, **Blond Hair** faces for the target domain.

to female faces with **black hair** which are **non-smiling** for α, and **smiling** for β. However, we only have access to biased samples from β, where female faces have **blond** instead of **black** hair.

In Fig. S6, we report results with CycleGAN and our approach with the quadratic cost: the hair color is modified along with the *smile* feature, and black-haired non smiling faces are mapped to blond smiling ones as should be expected from both. This highlights a particular case where CycleGAN's implicit bias fails.

Using our presented formulation, we are able to solve this task by changing the cost function: We use a a non-standard cost which is more suited to the geometry of the problem:

$$c(x, y) = \|H(x) - H(y)\|_2^2 \tag{S7}$$

where $H(I)$ is a histogram function of the image I. More precisely, H is computed as a soft histogram over the colors of the image of 20 bins, using a Gaussian kernel with $\sigma = 0.05$ for the smoothing. This cost allows to take into account the texture of the image, thus helping to find an OT map which preserves hair color in this case and re-balances the dataset as needed.

This task is an example of a case where a simple cost may help achieve non-trivial results when appropriate information is injected into it. In other words, by using prior knowledge on the corruption of the dataset, a cost function can be tailored to correct it.

More generally, it is not difficult to prove that a cost can almost always be designed to find the right solution for a given task between two distributions α and β among the infinity of candidates in the kernel of CycleGAN's loss. This is shown in Section B.1.3 of the Appendix.

4.1.5 Related Work

As discussed before, our work is motivated by the observations of works such as Galanti et al. 2018; Benaim et al. 2018 which have linked well-behaved UDT models with a notion of simplicity which we tried here to frame in a more rigorous and more useful formulation, making it task dependant. Moreover, similarly to us, Galanti et al. 2018; Benaim et al. 2017 show that learning a one-sided mapping is possible but do not directly obtain the inverse mapping as we do. Others have tried a hybrid approach between paired and unpaired translation Tripathy et al. 2018, which still doesn't solve the problem of ill-posedness as there generally still are infinitely many possible mappings. Also similar to us, Gong et al. n.d. uses a progressive interpolation. In the domain adaptation field, using Optimal Transport to help a classifier extrapolate has been around for some years, e.g. Courty et al. 2015; Damodaran et al. 2018 use a transport cost to align two distributions. The task, although related, is clearly different and so are the methods they develop. Finally, G. Lu et al. 2018 also try to regularize CycleGAN through OT but use barycenters from the optimal plan obtained in the discrete, static setting in order to guide the mapping instead of seeing it directly as an OT map (or as biased towards it), thus not explaining why CycleGAN works in practice.

4.1.6 Discussion and Conclusion

We start by formalizing what should be expected of a UDT mapping, namely that it should be conservative and regular. We then show empirically that CycleGAN works well when highly biased towards a particular form of conservation, which is unexpected as this is not enforced explicitly during training. We believe this is in particular due to gradient descent using residual architectures with small initializations.

A very interesting avenue for research would be to prove this theoretically, potentially making use of recent developments in the implicit regularization effect of gradient descent Ji et al. 2019; Soudry et al. 2018; Arora et al. 2019a.

We believe the proposed OT formulation is particularly adapted to UDT and allows us to leverage the plethora of theoretical and practical tools developed in this community. Typically, we were able to guarantee not only the existence and uniqueness of the solution, but also provide fine grained regularity results for the solution map. Moreover, we have also adapted practical algorithms from OT and have made analogies between residual networks and Dynamical OT which have resulted in an improved UDT model which is more robust and can be useful in settings where CycleGAN's biases fail.

4.2 Analysis of Classification Networks

abstract

Neural networks have been achieving high generalization performance on many tasks despite being highly over-parameterized. Since classical statistical learning theory struggles to explain this behaviour, much effort has recently been focused on uncovering the mechanisms behind it, in the hope of developing a more adequate theoretical framework and having a better control over the trained models. In this work, we adopt an alternative perspective, viewing the neural network as a dynamical system displacing input particles over time. We conduct a series of experiments and, by analyzing the network's behaviour through its displacements, we show the presence of a low kinetic energy bias in the transport map of the network, and link this bias with generalization performance. From this observation, we reformulate the learning problem as follows: find neural networks that solve the task while transporting the data as efficiently as possible. This offers a novel formulation of the learning problem which allows us to provide regularity results for the solution network, based on Optimal Transport theory. From a practical viewpoint, this allows us to propose a new learning algorithm, which automatically adapts to the complexity of the task, and leads to networks with a high generalization ability even in low data regimes.

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

Skander Karkar, Ibrahim Ayed, Emmanuel de Bézenac, and Patrick Gallinari (2020). "A Principle of Least Action for the Training of Neural Networks". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part II*. ed. by Frank Hutter, Kristian Kersting, Jefrey Lijffijt, and Isabel Valera. Vol. 12458. Lecture Notes in Computer Science. Springer, pp. 101–117. URL: https://doi.org/10.1007/978-3-030-67661-2%5C_7

4.2.1 Introduction

Deep neural networks (DNNs) have repeatedly shown their ability to solve a wide range of challenging tasks, while often having many more parameters than there are training samples. Such a performance of over-parametrized models is counter-intuitive. They seem to adapt their complexity to the given task,

systematically achieving a low training error without suffering from over-fitting as could be expected Belkin et al. 2019; Nakkiran et al. 2020; C. Zhang et al. 2017. This is in contradiction with the classical statistical practice of selecting a class of functions complex enough to represent the coherent patterns in the data, and simple enough to avoid spurious correlations Belkin et al. 2018; Hastie et al. 2001. Although this behavior has sparked much recent work towards explaining neural networks' success De Palma et al. 2019; Jacot et al. 2018a; R. Novak et al. 2018; Rahaman et al. 2019, it still remains poorly understood. Among the factors to consider are the implicit biases present in the choices made for the parametrization, the architecture, the parameter initialization and the optimization algorithm, and that contribute all to this success. Our aim in this work is to uncover some of these hidden biases and highlight their link with generalization performance through the lens of dynamical systems.

We will focus on residual networks (ResNets) K. He et al. 2016; Kaiming He et al. 2016a, now ubiquitous in applications. This family of models has made it possible to learn very complex non-linear functions by improving the trainability of very deep networks, and has thus improved generalization. Links have been derived between these networks and dynamical systems: a ResNet can be seen as a forward Euler scheme discretization of an associated ordinary differential equation (ODE) Weinan 2017:

$$\beta x_{k+1} = \beta x_k + v_k(\beta x_k) \iff \partial_t \beta x_t = v_t(\beta x_t) \tag{S8}$$

This link has yielded many exciting results, *e.g.* new architectures Y. Lu et al. 2018 and reversible networks Chang et al. 2018. Here, we make use of this analogy and analyze the behavior of residual networks by studying their associated differential flows. Adopting this dynamical point of view allows us to leverage the theories and mathematical tools developed to study, approximate and apply differential equations.

More specifically, we conduct experiments to observe how neural networks displace their inputs–seen as particles–through time. We measure a strong empirical correlation between good test performance and neural networks with low kinetic energy along their transport flow. From this, we reformulate the training problem as follows: retrieve the network which solves the task using the principle of least action, *i.e.* expending as little kinetic energy as possible. This problem, in its probabilistic formulation, is tightly linked with and inspired by the well-known problem of finding an optimal transportation map Santambrogio 2015. This yields new insights into neural networks' generalization capabilities, and provides a novel algorithm that automatically adapts to the complexity of the data and robustly improves the network's performance, including in low data

regimes, without slowing down the training. To summarize, our contributions are the following:

- Through the dynamic viewpoint, we highlight the *low-energy bias* of ResNets.
- We formulate a Least Action Principle for the training of Neural Networks.
- We prove existence and regularity results for networks with minimal energy.
- We provide an algorithm for retrieving minimal energy networks compatible with different architectures, which leads to better generalization performance on different classification tasks, without complexifying the architecture.

We introduce in Section 4.2.2 some background on Optimal Transport (OT) and highlight the link between the dynamical formulation of OT and ResNets. We describe in Section 4.2.3 the general setting of our analysis. Section 4.2.4 provides empirical evidence illustrating our point. The formal framework of networks trained with minimized energy and a practical algorithm are described in Section 4.2.5. Experiments on standard classification tasks are provided in Section 4.2.6. The code is available online at `github.com/skander-karkar/LAP`.

4.2.2 Background

This section outlines the main elements of the formalism and reasoning of our work. Supplementary Material A gives more details about Optimal Transport.

4.2.2.1 Optimal Transport

The principle of least action is central to many fields in physics, mathematics and economics. It is found in classical and relativistic mechanics, thermodynamics, quantum mechanics Feynman 2005; Garcia-Morales et al. 2008; Gray 2009, etc.. It broadly states that the dynamical trajectory of a system between an initial and final configuration is one that makes a certain action associated with the system locally stationary Gray 2009. One mathematical theory which can be associated with this general idea is the theory of Optimal Transport which was initially introduced as a way of finding a transportation map minimizing the cost of displacing mass from one configuration to another Santambrogio 2015.

Formally, let α and β be absolutely continuous distributions compactly supported in $\mathcal{R}^d$, and $c : \mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}$ a cost function. Consider a transportation map $T : \mathcal{R}^d \to \mathcal{R}^d$ that satisfies $T_\sharp \alpha = \beta$, *i.e.* that pushes[6] α to β. The total cost of the

6. $T_\sharp \alpha$ is the *push-forward measure*: $T_\sharp \alpha(B) = \alpha(T^{-1}(B))$ for any measurable set B.

transportation then depends on all the individual contributions of costs of transporting (infinitesimal) mass from each point x to $T(x)$, and finding the optimal transportation map amounts to solving:

$$\begin{aligned} \underset{T}{\text{minimize}} \quad & \mathcal{C}^{\text{stat}}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{subject to} \quad & T_\sharp \alpha = \beta \end{aligned} \tag{S9}$$

A standard choice for c is the p-th power of a norm of $\mathcal{R}^d$, *i.e.* $c(x, y) = \|x - y\|^p$, but other costs can be used, defining different variants of the problem. This cost induces, through the p-th root of the minimal value of eq. S9, a distance W_p between any two distributions α and β of finite $p-$th moment, called the p-Wasserstein distance Peyre et al. 2018.

In Benamou et al. 2000, the link between Optimal Transport and the principle of least action was made by showing that the static transportation can equivalently be viewed as a dynamical one that minimizes an action as it gradually displaces particles of mass in time. In other words, instead of directly pushing samples of α to β in $\mathbb{R}^d$ using T, we can displace mass from α according to a continuous flow with velocity $v_t : \mathcal{R}^d \to \mathcal{R}^d$. This implies that the density μ_t at time t satisfies the *continuity equation* $\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$, assuming that initial and final conditions are given respectively by $\mu_0 = \alpha$ and $\mu_1 = \beta$. In this case, the optimal displacement is the one that minimizes the action $\|v_t\|^p_{L^p(\mu_t)}$:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \mathcal{C}^{\text{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{d}t \\ \text{subject to} \quad & \partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0, \mu_0 = \alpha, \mu_1 = \beta \end{aligned} \tag{S10}$$

where $\|v_t\|^p_{L^p(\mu_t)} = \int_{\mathcal{R}^d} \|v_t(x)\|^p \mathrm{d}\mu_t(x)$ for costs $c(x, y) = \|x - y\|^p$ with $p > 1$. In this case, minimizers exist and the two transport costs are the same, *i.e.* $\mathcal{C}^{\text{stat}}(T) = \mathcal{C}^{\text{dyn}}(v)$ at the optimums. For $p = 2$ and the Euclidean norm, the dynamical cost $\mathcal{C}^{\text{dyn}}(v)$ corresponds to the *kinetic energy*.

4.2.2.2 Link with Residual Networks

The dynamical formulation in eq. S10 explicitly describes the evolution in time of the density μ_t, starting from an input distribution α. In this form, the link between deep residual networks and dynamical Optimal Transport is not clear. However, it is possible to adopt an alternate viewpoint which helps make it immediate. Instead of explicitly describing the density's evolution, we describe the paths $\phi^x \colon [0, 1] \to \mathcal{R}^d,\ t \mapsto \phi^x_t$ taken by particles from α at position x, when

displaced along the flow v. The continuity equations can then equivalently be written as:

$$\partial_t \phi_t^x = v_t(\phi_t^x) \tag{S11}$$

See chapters 4 and 5 of Santambrogio 2015 for details. We can now note the resemblance between the residual network eq. S8 and equation eq. S11. Rewriting the conditions as necessary, the dynamical formulation eq. S10 can equivalently be represented by:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \mathcal{C}^{\text{lag}}(v) = \int_0^1 \|v_t\|^p_{L^p((\phi_t^\cdot)_\sharp \alpha)} \, \mathrm{d}t \\ \text{subject to} \quad & \partial_t \phi_t^x = v_t(\phi_t^x), \\ & \phi_0^\cdot = \mathrm{id}, \\ & (\phi_1^\cdot)_\sharp \alpha = \beta \end{aligned} \tag{S12}$$

where $\phi_t^\cdot : x \in \mathcal{R}^d \mapsto \phi_t^x \in \mathcal{R}^d$ corresponds to the transport map induced by the flow, up until time t. As both formulations are equivalent, we have that for any flow v, $\mathcal{C}^{\text{lag}}(v) = \mathcal{C}^{\text{dyn}}(v)$. Moreover, optimal transportation plans in the static eq. S9 and dynamical eq. S12 cases coincide: if T and $\phi_t^\cdot$, are respectively solutions to eq. S9 and eq. S12, we have that $T = \phi_1^\cdot$.

This link allows us to associate residual networks with a local action for each layer, which induces a global transportation cost $\mathcal{C}^{\text{lag}}$, and taking $p = 2$ and the Euclidean norm allows us to refer to the network's kinetic energy.

4.2.3 General Setting

In order to better understand the inner workings of a DNN, it is essential to adopt a viewpoint in which the different driving mechanisms become apparent and are decoupled.

Decomposing a DNN We consider the following model of a deep neural network f where computations are separated into the three steps, *i.e.* $f = F \circ T \circ \varphi$ (this is similar to Q. Li et al. 2018 and corresponds to the general structure of recent deep models or to the structure of components of a deep model Kaiming He et al. 2016a; Xie et al. 2017; Zagoruyko et al. 2016):

1. **Dimensionality change:** Starting from an input distribution $\mathcal{D}$ in $\mathcal{R}^n$, a transformation φ is applied, transforming it into $\alpha = \varphi_\sharp \mathcal{D}$, a distribution in $\mathcal{R}^d$. This corresponds to the first few layers present in most recent architectures and represents a change of dimensionality. φ is known as the *encoder*.

2. **Data Transport:** Then α is transformed by a mapping $T : \mathcal{R}^d \to \mathcal{R}^d$, which we see as a transport map. Here, the dimensionality doesn't change and, if this part of the network is a sequence of residual blocks, T can be written as the discretized flow of an ODE.

3. **Task-specific final layers:** A final function $F : \mathcal{R}^d \to \mathcal{Y}$ is applied to $T_\sharp\alpha$ in order to compute the loss $\mathcal{L}$ associated with the task at hand, *e.g.* F could be a perceptron classifier. Like φ, F is typically made up of a few layers.

The focus of this work is on analyzing the second phase, Data Transport, and we assume that the encoder φ is pretrained and fixed (this will be relaxed in some experiments later). To solve a complex non-linear task for which a DNN is needed, the data has to be transformed in a non-trivial way, meaning that this is an essential phase, *e.g.* in the case of classification, $T_\sharp\alpha$ needs to be linearly separable if F is linear. This model is quite general, as many ResNet-based architectures Xie et al. 2017; Zagoruyko et al. 2016 alternate modules that change the dimensionality (step 1) and transport modules that keep the dimensionality fixed (step 2) and according to Jastrzebski et al. 2018, the transport modules have similar behaviour. The model can then be considered as a simplified ResNet, sometimes called a *single representation* ResNet. Note that Sandler et al. 2019 finds that networks that keep the same resolution remain competitive.

The set of admissible targets As recent neural architectures have systematically achieved near-zero training error Belkin et al. 2019; Belkin et al. 2018; Jacot et al. 2018a; C. Zhang et al. 2017, we place ourselves in this regime, which makes it possible to model this as a hard constraint. For some tasks, this constraint over T is obvious: in a generative setting for example, $T_\sharp\alpha$ must be equal to some prescribed distribution β which is the target of the generation process. But in general, T is less strictly constrained and the condition depends on F and $\mathcal{L}$. This leads us to define a *set of admissible targets* for the task:

$$S_{F,\mathcal{L}} = \{\beta \in \mathcal{P}(\mathcal{R}^d) \mid \mathcal{L}(F, \beta) = 0\} \tag{S13}$$

with $\beta = T_\sharp\alpha$. In general, $\mathcal{L}$ is fixed while F is learned jointly with T. This set is supposed to be non-empty for some F and, in general, it will contain many distributions. The goal of the learning task can then be reformulated as:

$$\text{Find } (T, F) \text{ such that } T_\sharp\alpha \in S_{F,\mathcal{L}} \tag{S14}$$

An important observation is that, even when $S_{F,\mathcal{L}}$ is reduced to a singleton, the problem is still strongly under-constrained and it is possible to obtain many such (T, F) that lead to poor generalization. One can then ask why this is not the case in practice, as good generalization performance is usually achieved.

The case of classification Even though our framework is general, we focus our experiments on classification tasks, with $\mathcal{L}$ being the cross entropy loss. The task consists in separating N classes. Let us denote α_i the class distributions which are supposed to be distributions in $\mathbb{R}^d$ of mutually disjoint supports, meaning that there is no ambiguity in the class of data points, and such that $\alpha = \sum_i \alpha_i/N$. One wants to find a transformation T of these distributions such that all transported distributions can be correctly classified by a classifier F. When F is linear, $S_{F,\mathcal{L}}$ is the set of distributions which have N components that are linearly separated by F. Note that we place ourselves in a noiseless ideal setting where perfect classification is possible. The question we examine in this work is then twofold:

- What are the properties characterizing mappings reached by standard residual architectures with common hyper-parameters?

- Can we find a criteria to *automatically select* mappings with desirable properties in order to improve performance and robustness?

4.2.4 Empirical Analysis of Transport Dynamics in ResNets

Before introducing our framework, we conduct an exploratory analysis of the impact of the network's inner dynamics on generalization. We present below two experiments. The first one highlights how good generalization performance is closely related to low transport cost for classification tasks on MNIST and CIFAR10. This cost therefore appears as a natural characterisation of the complexity and disorder of a network. The second experiment, performed on a toy 2D dataset, visualizes the transport induced by the blocks of a ResNet.

We consider ResNets where, after encoding, a data point x_0 is transported by applying $x_{k+1} = x_k + v_k(x_k)$ for K residual blocks and then classified using F. We measure the disorder/complexity of a network by its transport cost which is the sum of the displacements induced by its residual blocks: $\mathcal{C}(v) = \sum_k \|v_k(x_k)\|_2^2$. This quantity corresponds to the kinetic energy of the total displacement.

Transportation cost and generalization on MNIST and CIFAR10. In order to study the correlation between the transport cost of a residual network and its generalization ability on image data, we train convolutional 9-block ResNets with different initializations (orthogonal and normal with different gains), for 10-class classification tasks MNIST and CIFAR10. In Figure S8, each point represents a trained network and gives the transport cost $\mathcal{C}$ as a function of the test accuracy of the network. This experiment clearly highlights the strong negative correlation between transport cost and good generalization. This illustrates the importance of

the implicit initialization bias and motivates initialization schemes which favour a low kinetic energy. We believe a number of factors contribute to this low energy bias: small initialization gains tend to bias $\|v_k(x_k)\|_2^2$ towards small values, and training using gradient descent does not change this much.

Visualizing network dynamics on 2D toy data. This experiment provides a 2D visualization of the transport dynamics inside a network. The task is 2-class classification of a non-linearly separable dataset (two concentric circles, from `sklearn`) that contains 1000 points with a train-test split of 80%-20%, see Figure S7 top left. The network is a ResNet containing 9 residual blocks, followed by a fixed linear classifier. Each residual block contains two fully connected layers separated by a batch normalization and a ReLU activation.

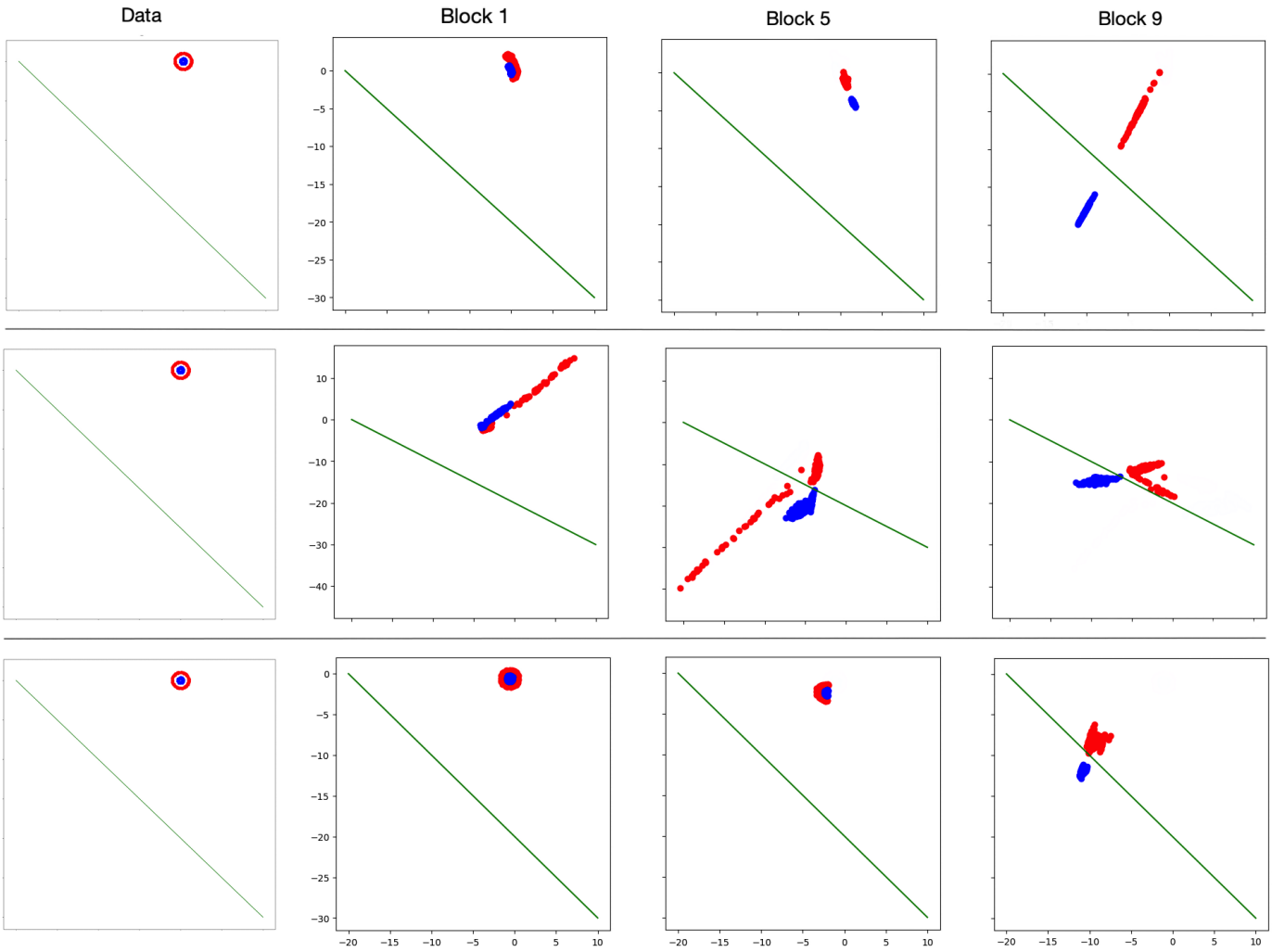


Figure S7. – Transformed circles test set by a ResNet9 after blocks 1, 5 and 9 after training; first row with good initialization; second row with a $\mathcal{N}(0, 5)$ initialization; third row with a $\mathcal{N}(0, 5)$ initialization and the transport cost added to the loss

With the cross-entropy loss alone, the behaviour of a well trained and carefully initialized network achieving 100% test accuracy is illustrated in the first row of Figure S7. With a $\mathcal{N}(0, 5)$ initialization, significantly bigger than an "optimal" initialization, the test accuracy drops to 98% (average of 100 runs) and the transport becomes chaotic (Figure S7, second row). Adding the transport cost to the loss improves the test accuracy (99.7% on average) of this badly initialized network and the movement becomes more controlled (third row of Figure S7). Thus, controlling transport improves the behavior and generalization ability of the network. This allows to explicitly control the network whereas implicit biases such as "good" initialization rely on heuristics. In Supplementary Material C.4, more experiments show that in other situations that deviate from the ideal setting

where the task is perfectly solved, e.g. when using a network which is too large or too small, or a small training set, controlling the transport cost also improves generalization.

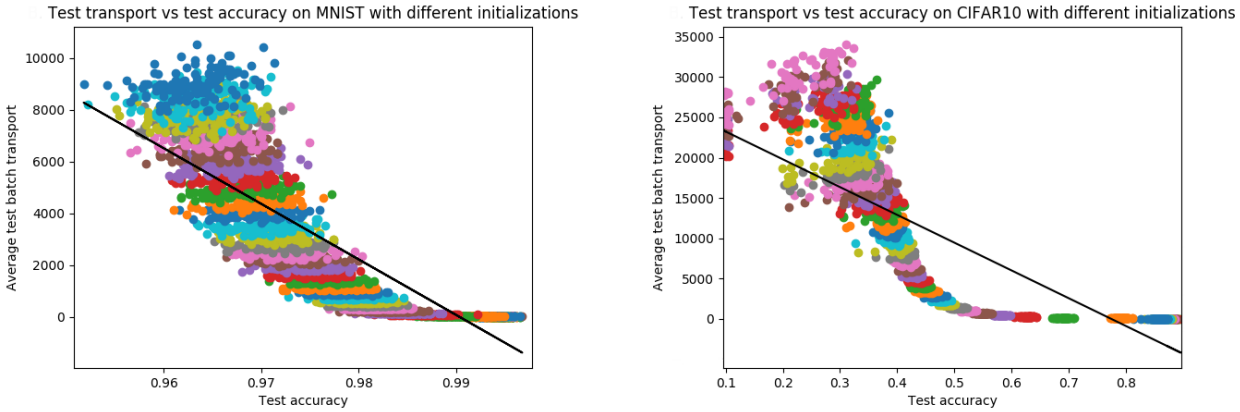


Figure S8. – Test transport against test accuracy of ResNet9 models on MNIST (left) and CIFAR10 (right) with fitted linear regressions, where each color indicates a different initialization (either orthogonal or normal with varying gains)

4.2.5 Least Action Principle for Training Neural Networks

The previous section has shed some light on the low energy bias of networks as well as on its potential benefits on test accuracy. In this section, we take a step further and make this implicit bias explicit by considering a formulation for training that enforces minimal kinetic energy, closely related to the problem of Optimal Transport. This allows us to prove the existence of minimizers, and exhibit interesting regularity properties of the minimal energy neural networks which may explain good generalization performance.

4.2.5.1 Formulation

We consider costs $c(x, y) = \|x - y\|^p$ (where $\|.\|$ is a norm of $\mathcal{R}^d$), with $p > 1$, and suppose that $\alpha \in \mathcal{P}_p(\mathcal{R}^d)$ (the set of absolutely continuous measures on $\mathcal{R}^d$ with finite p-th moment). We assume that the space of classifiers is compact, that the loss $\mathcal{L}$ is continuous, that the set $\cup_{F\in\mathcal{F}} S_{F,\mathcal{L}}$ is at a finite p-Wasserstein distance W_p from α (in particular, it is non-empty) and that all its bounded subsets are totally bounded (*i.e.* can be covered by finitely many subsets of any fixed size). These properties depend on the choice of the loss $\mathcal{L}$ and of a class of functions $\mathcal{F}$ for the classifier F.

Returning to the transport problem as defined in Section 4.2.2.1, a natural way to select a robust model, given the empirical observations of Section 4.2.4, is to select, among the maps which transport α to $S_{F,\mathcal{L}}$ and thus solve the task, one with a minimal transport cost. This gives us the following optimization problem:

$$\begin{aligned} \inf_{T,F} \quad & \mathcal{C}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{subject to} \quad & T_\sharp \alpha \in S_{F,\mathcal{L}} \end{aligned} \tag{S15}$$

The equivalent dynamical version for $c(x, y) = \|x - y\|^p$ is, as per Section 4.2.2.2,

$$\begin{aligned} \inf_{v,F} \quad & \int_0^1 \|v_t\|^p_{L^p((\phi_t^\cdot)_\sharp \alpha)} \, \mathrm{d}t \\ \text{subject to} \quad & \partial_t \phi_t^x = v_t(\phi_t^x) \\ & \phi_0^\cdot = \mathrm{id} \\ & (\phi_1^\cdot)_\sharp \alpha \in S_{F,\mathcal{L}} \end{aligned} \tag{S16}$$

where $\|v_t\|^p_{L^p((\phi_t^\cdot)_\sharp \alpha)} = \int_{\mathbb{R}^d} \|v_t\|^p \mathrm{d}(\phi_t^\cdot)_\sharp \alpha$. The result below shows that these two problems are equivalent and that the infima are realized as minima:

Theorem S4. *The infima of eq. S15 and eq. S16 are finite and are realized through a map T which is (or a velocity field v which induces) an optimal transportation map. When $c(x, y) = \|x - y\|^p$, then eq. S15 and eq. S16 are equivalent.*

Proof. From the hypothesis above, there exists $\beta \in S_{F,\mathcal{L}}$ at a finite distance from α. Taking any transport map between α and β, we see that the infima are finite.

Consider eq. S15 and take a minimizing sequence $(T_i, F_i)_i$. Set $\beta_i = (T_i)_\sharp \alpha$. Then $(\mathcal{C}(T_i))_i$ converges to the infimum which is strictly bounded by $M > 0$. Then, by definition, for i large enough, $W_p^p(\alpha, \beta_i) \leq \mathcal{C}(T_i) \leq M$. So that $(\beta_i)_i$ is a bounded sequence in $\cup_F S_{F,\mathcal{L}}$. By the hypothesized total boundedness of bounded subsets and as $\mathcal{P}_p(\mathcal{R}^d)$ endowed with W_p is a complete metric space (see Bolley 2008 for a proof), up to an extraction, $(\beta_i)_i$ converges to $\beta^\star$ in the closure of $\cup_F S_{F,\mathcal{L}}$. Moreover, up to an extraction, $(F_i)_i$ also converges to $F^\star$ by compactness of the class of classifiers. Taking $T^\star$ the OT map between α and $\beta^\star$ (see Supplementary Material A for existence of OT maps), we then have, by continuity of $\mathcal{L}$,

$$T^\star_\sharp \alpha = \beta^\star \in S_{F^\star,\mathcal{L}}$$

and $\mathcal{C}(T^\star) \leq \lim \mathcal{C}(T_i)$ by optimality of $T^\star$, which means, since $(\mathcal{C}(T_i))_i$ is a minimizing sequence, that $\mathcal{C}(T^\star)$ minimizes eq. S15. So $(T^\star, F^\star)$ is a minimizer and $T^\star$ is an OT map.

Finally, there exists, by dynamical OT theory (Supplementary Material A), a velocity field $v_t^\star$ inducing the OT map between α and $\beta^\star$ which then gives a minimizer $(v^\star, F^\star)$ for eq. S16. By the same reasoning, taking a minimizing sequence $(v^{(i)}, F_i)_i$ and the induced maps T_i shows that both problems are equivalent. □ □

Note that uniqueness doesn't hold anymore, as the constraint $T_\sharp\alpha \in S_{F,\mathcal{L}}$ in eq. S16 is looser than in standard OT. However, as we show in the following section, the fact that the optimization problems are solved by OT maps will give regularity properties for the models induced by these optimization problems.

4.2.5.2 Regularity

Intuitively, the fact that we minimize the energy of the transport map transforming the data is akin to the core idea of Occam's razor: among all the possible networks that correctly solve the the task, the one transforming the data in the simplest way is selected. Moreover, it is possible to show that this optimal transformation is regular: our formulation provides an alternate view on generalization for modern deep learning architectures in the overparametrized regime.

Optimal maps can be as irregular as needed in order to fit the target distribution, however in much the same way as successfully trained DNNs, optimal maps are still surprisingly regular. In a way, they are as regular as possible given the constraints which is exactly the type of flexibility needed. However, the constraints in eq. S15 and eq. S16 are looser than in the standard definitions of Optimal Transport. Still, supposing that the input data distribution has a nicely behaved density, namely bounded and of compact support, with the same hypothesis as above, we have the following, which is mainly a corollary of Theorem S4:

Proposition S5. *Consider $T^\star$ the OT map induced by eq. S15 (or eq. S16) given by Theorem S4. Take X, respectively Y, an open neighborhood of the support of α, respectively of $T_\sharp^\star\alpha$, then $T^\star$ is differentiable, except on a set of null α measure.*

Additionally, if $T^\star$ doesn't have singularities, there exists $\eta > 0$ and A, respectively B, relatively closed in X, respectively Y, such that $T^\star$ is η-Hölder continuous from $X \setminus A$ to $Y \setminus B$. Moreover, if the two densities are smooth, $T^\star$ is a diffeomorphism from $X \setminus A$ to $Y \setminus B$.

Proof. This is a consequence of Theorem S4, the hypothesis made in this section and the regularity theorems stated in Supplementary Material B. □ □

There are two main results in Proposition S5: the first gives α-$a.e.$ differentiability. This is already as strong as might be expected from a classifier: there are

necessarily discontinuities at the frontiers between different classes. The second is even more interesting: it gives Hölder continuity over as large a domain as possible, and even a diffeomorphism if the data distribution is well-behaved enough. We recall that a function f is η-Hölder continuous for $\eta \in]0, 1]$ if $\exists\ M > 0$ such that $\|f(x) - f(y)\| \leq M\|x - y\|^\eta$ for all x, y. η measures the smoothness of f, the higher its value, the better. In particular, in the case of classification, this means that the Hausdorff dimension along the frontiers between the different classes is scaled by less than a factor of $1/\eta$ in the transported domain. If the densities are smooth, the dimension even becomes provably smaller by this result.

Intuitively, this means that, in these models, the data is transported in a way that preserves and simplifies the patterns in the input distribution. In the following, we propose a practical algorithm implementing these models and use it for standard classification tasks, showing an improvement over standard models.

4.2.5.3 Practical Algorithm

We propose an algorithm for training ResNets using the least action principle by minimizing the kinetic energy. Starting from problem eq. S16 with $p = 2$ and the Euclidean norm, we first discretize the differential equation via a forward Euler scheme, which yields $\phi^x_{k+1} = \phi^x_k + v_k(\phi^x_k)$. The discretized flow v_k is parameterized by a residual block, giving a standard residual architecture. The residual blocks, along with a classifier F, are parametrized by θ. Next, the constraint $(\phi_1^\cdot)_\sharp \alpha \in S_{F,\mathcal{L}}$ is rewritten as $\mathcal{L}(F, (\phi_1^\cdot)_\sharp \alpha) = 0$, denoted $\mathcal{L}(\theta) = 0$ below. Finally, as we only have access to a finite set $\mathcal{X}$ of samples x from α, we use a Monte-Carlo approximation of the integral *w.r.t* the distributions $(\phi_t^\cdot)_\sharp \alpha$, to obtain:

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \mathcal{C}(\theta) = \sum_{x \in \mathcal{X}} \sum_{k=0}^{K-1} \|v_k(\phi^x_k)\|_2^2 \\ \text{subject to} \quad & \phi^x_{k+1} = \phi^x_k + v_k(\phi^x_k), \\ & \phi^x_0 = x,\ \forall\ x \in \mathcal{X}, \\ & \mathcal{L}(\theta) = 0 \end{aligned} \tag{S17}$$

Is is easy to see that the min-max problem $\min_\theta \max_{\lambda > 0}\ \mathcal{C}(\theta) + \lambda\, \mathcal{L}(\theta)$ yields the same solution, as the first two constraints are satisfied trivially. If the constraint $\mathcal{L}(\theta) = 0$ corresponding to solving the task, which includes the classifier F, is not verified, this will cause the second term to grow unbounded, and the solution will thus be avoided by the minimization. This min-max problem can be solved using an iterative approach, starting from some initial λ_0 and θ_0:

$$\begin{cases} \theta_{i+1} = \arg\min_{\theta} \ \mathcal{C}(\theta) + \lambda_i \ \mathcal{L}(\theta) \\ \lambda_{i+1} = \lambda_i + \tau \ \mathcal{L}(\theta_{i+1}) \end{cases} \tag{S18}$$

The minimization is done via SGD for a number of steps s, where a step means a batch, starting from the previous parameter value θ_i. This algorithm is similar to Uzawa's algorithm used in convex optimization Santambrogio 2015. In practice, it is more stable to divide the minimization objective in eq. S18 by λ_i, yielding:

Algorithm 4 Training neural networks with Least Action Principle (LAP-Net)

Input: Training samples, step size τ, number of steps s, initial weight λ_0
Initialization: Initialize the parameters θ_0 and set $i = 0$
while not converged **do**
1. Starting from θ_i, perform s steps of stochastic gradient descent:
   1.1. $\theta_{i+1}^0 = \theta_i$
   1.2. $\theta_{i+1}^l = \theta_{i+1}^{l-1} - \epsilon(\nabla \mathcal{C}(\theta_{i+1}^{l-1})/\lambda_i + \nabla \mathcal{L}(\theta_{i+1}^{l-1}))$ for l from 1 to s
   1.3. $\theta_{i+1} = \theta_{i+1}^s$
2. Update the weight $\lambda_{i+1} = \lambda_i + \tau \ \mathcal{L}(\theta_{i+1})$ and increment $i \leftarrow i + 1$
end while
Output: Learned parameters θ

While the high non-convexity makes it difficult to ensure exact optimality, we can still have some induced regularity when reaching a "good" local minimum:

Proposition S6. *Suppose $(F^{\theta^\star}, T^{\theta^\star})$ is reached by the optimization algorithm such that $T^{\theta^\star}$ is an $\epsilon-$OT map between α and its push-forward*[7]*. Then we have, with the same notations as in Proposition S5,*

$$\forall x, y \in X \setminus A, \|T^{\theta^\star}(x) - T^{\theta^\star}(y)\| \leq O(\epsilon + \|x - y\|^\eta)$$

Proof. We simply write the decomposition:

$$T^{\theta^\star}(x) - T^{\theta^\star}(y) = T^{\theta^\star}(x) - T^\star(x) + T^\star(x) - T^\star(y) + T^\star(y) - T^{\theta^\star}(y)$$

and use the triangular inequality: the first and third terms are smaller than ϵ by hypothesis while Hölder continuity applies for the second by Proposition S5. □ □

This shows that minimizing the transport cost still endows the model with some regularity, even in situations where the global minimum is not reached.

7. By this, we mean that $\|T^{\theta^\star} - T^\star\|_\infty \leq \epsilon$ where $T^\star$ is the OT map.

4.2.6 Experiments

MNIST Experiments The base model is a ResNet with 9 residual blocks. Two convolutional layers first encode the image of shape $1 \times 28 \times 28$ into shape $32 \times 14 \times 14$. A residual block contains two convolutional layers, each preceded by a ReLU activation and batch normalization. The classifier is made up of two fully connected layers separated by batch normalization and a ReLU activation. We use an orthogonal initialization Saxe et al. 2014 with gain 0.01. This and all vanilla models and their training regimes are implemented by following closely the cited papers that first introduced them and our method is added over these training regimes. More implementation details are in Supplementary Material C.3.

When using the entire training set, the task is essentially solved (99.4% test accuracy). We penalize the transport cost as presented in Section 4.2.5.3, using $\lambda_0 = 5$, $\tau = 1$ and $s = 5$. The performance barely drops (99.3% test accuracy), and we can visualise the preservation of information from the point of view of a pretrained autoencoder (see Supplementary Material C.1). From the experiments in two dimensions, we suspect that adding the transport cost helps when the training set is small. For performance comparisons, we average the highest test accuracy achieved over 30 training epochs (over random orthogonal weight initializations and random subsets of the complete training set). We find that adding the transport cost improves generalization when the training set is very small (Table S1). We see that the improvement becomes more important as the training set becomes smaller and reaches an increase of almost 14 percentage points in the average test accuracy.

Training set size	ResNet	LAP-ResNet (Ours)
500	90.8, [90.4, 91.2]	**90.9**, [90.7, 91.1]
400	88.4, [88.0, 88.8]	**88.4**, [88.0, 88.8]
300	83.5, [83.0, 84.1]	**86.2**, [85.8, 86.6]
200	74.9, [73.9, 75.9]	**82.0**, [81.5, 82.5]
100	56.4, [54.9, 58.0]	**70.0**, [69.0, 71.0]

Table S1. – Average highest test accuracy and 95% confidence interval of ResNet9 over 50 instances on MNIST with training sets of different sizes (in %)

CIFAR10 Experiments We run the same experiments on CIFAR10. The architecture is exactly the same except that the encoder transforms the input which is of shape $3 \times 32 \times 32$ into shape $100 \times 16 \times 16$. For our method, we use $\lambda_0 = 0.1$, $\tau = 0.1$ and $s = 50$. We average the highest test accuracy achieved over 200 training epochs over random orthogonal weight initializations and random subsets of

the complete train set. Here, we find that adding the transport cost helps for all sizes of the train set (which has 50 000 images in total). The increase in average precision becomes more important as the train set becomes smaller (Table S2).

Training set size	ResNet	LAP-ResNet (Ours)
50 000	91.49, [91.40, 91.59]	**91.94**, [91.84, 92.04]
30 000	88.61, [88.47, 88.75]	**89.41**, [89.31, 89.50]
20 000	85.73, [85.59, 85.87]	**86.74**, [86.61, 86.87]
10 000	79.25, [79.00, 79.49]	**80.90**, [80.74, 81.06]
5 000	70.32, [70.00, 70.63]	**72.58**, [72.36, 72.79]
4 000	67.80, [67.55, 68.07]	**70.12**, [69.81, 70.42]

Table S2. – Average highest test accuracy and 95% confidence interval of ResNet9 over 20 instances on CIFAR10 with training sets of different sizes (in %)

CIFAR100 experiments On CIFAR100, results using a ResNet are in Supplementary Material C.2. We also used the ResNeXt Xie et al. 2017 architecture: the residual block of a ResNeXt applies $x + \sum_i w_i(x)$ with the functions w_i having the same architecture but independent weights, followed by a ReLU activation. We used the ResNeXt-50-32×4d architecture detailed in Xie et al. 2017. This is a much bigger and state-of-the-art network, as compared with the single representation ResNet used so far. It also extends the experimental results beyond the theoretical framework in three ways: the embedding dimension changes between the residual blocks, a block applies $x_{k+1} = \texttt{ReLU}(x_k + \sum_i w_{k,i}(x_k))$ and the encoder is no longer fixed. We found that penalizing $\sum_i w_{k,i}(x_k)$ or $x_{k+1} - x_k$ is essentially equivalent. Table S3 shows consistent accuracy gains as our method (with $\lambda_0 = 1$, $\tau = 0.1$ and $s = 5$) corrects a slight overfitting of the bigger ResNeXt compared to ResNet.

Training set size	ResNeXt	LAP-ResNeXt (Ours)
50 000	72.97, [71.79, 74.14]	**76.11**, [75.32, 76.89]
25 000	62.55, [60.18, 64.92]	**64.11**, [62.25, 65.96]
12 500	45.90, [43.16, 48.67]	**48.23**, [46.39, 50.07]

Table S3. – Average highest test accuracy and 95% confidence interval of ResNeXt50 over 10 instances on CIFAR100 with training sets of different sizes (in %)

An important observation is that adding the transport cost significantly reduces the variance in the results. This is expected as the model becomes more

constrained and can be seen as an advantage, especially in cases where the results vary more with the initialization (*e.g.* transfer learning). This is illustrated by the width of the 95% confidence intervals in the tables above often becoming narrower when the transport cost is penalized. Finally, we could also have considered a relaxation of the optimization program by considering a fixed weight λ, which provides a simpler and quite competitive benchmark (see Supplementary Material C.2). The training's progress is shown there as well, and we see that the training is not slowed down by our method.

4.2.7 Related Work

That ResNets (K. He et al. 2016; Kaiming He et al. 2016a) are naturally biased towards minimally transforming their input, especially for later blocks and deeper networks, is already shown in (Jastrzebski et al. 2018), which found that earlier blocks learn new representations while later blocks only slowly refine those representations. (Hauser 2019) found that the deeper the network the more its blocks minimally move their input. Both were inspirations for this work. The ODE point of view of ResNets has inspired new architectures (Chang et al. 2018; Haber et al. 2019; Y. Lu et al. 2018; Ruthotto et al. 2020). Others were inspired by numerical schemes to improve stability, e.g. (Chang et al. 2018) add a penalty term that encourages the weights to vary smoothly from layer to layer and (J. Zhang et al. 2019) replicate an Euler scheme and study the effect of diminishing the discretization step-size. More recently, (Yan et al. 2020) accelerate the training of (R. Chen et al. 2018)'s model for generative tasks using the link with dynamical transport. But most often, regularization is achieved by penalization of the weights (e.g. spectral norm regularization (Yoshida et al. 2017), smoothly varying weights (Chang et al. 2018)).

OT theory was used in (Sonoda et al. 2019) to analyse deep gaussian denoising autoencoders (not necessarily implemented through residual networks) as transport systems. In the continuous limit, they are shown to transport the data distribution so as to decrease its entropy. Closer to this work, the dynamical formulation of OT is used in (Bézenac et al. 2019) for the problem of unsupervised domain translation.

4.2.8 Discussion and Conclusion

In this work, we have studied the behavior of ResNets by adopting a dynamical systems perspective. This viewpoint leverages the vast literature in this field.

More specifically, we have analyzed ResNets' complexity through the lens of the transport cost induced by the data displacement across the model's blocks. We find that due to a certain number of factors, this transport cost is biased towards small values. Moreover, this cost is negatively correlated to test accuracy, which has brought us to consider explicitly minimizing it. This leads us to present a novel generic formulation for training neural networks, based on the least action principle, closely related to the problem of Optimal Transport: amongst all the neural networks that correctly solve the task, select the one that transforms the data with the lowest cost. Note that even though we have only considered residual networks as they induce an ODE flow, this framework can be applied to any architecture by considering the static formulation eq. S15 of the problem.

We have proven general results of existence and regularity for models trained within our framework, studied their behaviour in low-dimensional settings when compared to vanilla models and shown their efficiency on standard classification tasks. We also found that the training is stabilized in an adaptive fashion without being slowed down.

An important property of our method which is yet to be tested and is hinted at by the regularity results and by the lower variance in the performances is the robustness of the models, more specifically in adversarial contexts. This will be one important venue of future work. Another interesting avenue of research would be to experiment with alternative transportation costs.

CHAPTER 5

ADDITIONAL WORK

5.1 Unsupervised Image Reconstruction

abstract

We address the problem of recovering an underlying signal from lossy, inaccurate observations in an unsupervised setting. Typically, we consider situations where there is little to no background knowledge on the structure of the underlying signal, no access to signal-measurement pairs, nor even unpaired signal-measurement data. The only available information is provided by the observations and the measurement process statistics. We cast the problem as finding the maximum a posteriori estimate of the signal given each measurement, and propose a general framework for the reconstruction problem. We use a formulation of generative adversarial networks, where the generator takes as input a corrupted observation in order to produce realistic reconstructions, and add a penalty term tying the reconstruction to the associated observation. We evaluate our reconstructions on several image datasets with different types of corruptions. The proposed approach yields better results than alternative baselines, and comparable performance with model variants trained with additional supervision.

The work in this section has led to the publication of a conference paper: Arthur Pajot, Emmanuel de Bezenac, and Patrick Gallinari (Sept. 2018). "Unsupervised Adversarial Image Reconstruction". In: URL: ICLR%202019%20:%20https://openreview.net/forum?id=BJg4Z3RqF7

5.1.1 Introduction

Many real world applications require acquiring information about the state of some physical system from incomplete and inaccurate measurements. For example, in infrared satellite imagery, one has to deal with the presence of clouds and a variety of other external factors perturbing the acquisition of temperature maps. This raises questions on how to recover the correct information and eliminate the contribution of external factors hindering the overall signal acquisition.

In this context, signal recovery does not usually yield a unique solution, meaning that multiple signal reconstructions could trivially explain the measurements. For the above example, different missing temperature values could accurately explain the observations. To cope with this indeterminacy, one usually relies on prior information on the structure of the true signal in order to constrain the reconstruction to plausible solutions (Stuart 2010). A common approach is to use handcrafted, analytically tractable priors (Candès et al. 2005, Mota et al. 2017).

This approach is limited to situations for which the underlying signal structure can be easily described, which are rarely observed in the wild.

Recent developments in generative models parameterized by neural networks (I. J. Goodfellow et al. 2014b, Kingma et al. 2013, Dinh et al. 2016) offer a promising statistical approach to signal recovery, for which priors on the signal are not handcrafted anymore, but learned from large amounts of data. Despite exhibiting interesting results (Bora et al. 2017, Mardani et al. 2017, Ledig et al. 2016), these methods all require some form of supervision, either observation measurement-signal pairs, or at least unpaired samples from observations and underlying signals. For many practical problems, obtaining these samples is too expensive and/or impractical, which makes these approaches not suitable for such situations.

We address the problem of image reconstruction in an unsupervised setting, when only corrupted observations are available, together with some prior information on the nature of the measurement process.

The learning problem is formulated as finding the maximum a posteriori estimate of signals given their measurements on the training set. We derive a natural objective for our reconstruction network, composed of a linear combination of an adversarial loss for recovering realistic signals, and a reconstruction loss to tie the reconstruction to its associated observation (Section 5.1.2.2). This model is evaluated and compared to baselines on 3 image datasets, CelebA (Z. Liu et al. 2015), LSUN Bedrooms (Yu et al. 2015), Recipe-1M (Marin et al. 2018), where we experiment with different types of measurement processes corrupting the images.

Our contributions are:

- A novel, computationally efficient framework for dealing with large scale signal recovery in an unsupervised context, applicable to a wide range of situations,
- A model and a new way of training a deep learning architecture for implementing this framework,
- Extensive evaluations on a number of image datasets with different measurement processes.

5.1.2 Preliminaries

Notations. We use capital letters (*e.g.* X) for random variables, and lower-case letters (*e.g.* x) for their values. $\mathrm{p}_X(x)$ denotes the distribution (or its density in the appropriate context) of X evaluated at x.

5.1.2.1 Problem setting.

Suppose there exists a signal $X \sim \mathrm{p}_X$ we wish to acquire, but we only have access to this signal through lossy, inaccurate measurements $Y \sim \mathrm{p}_Y$. The measurement process is modeled through a stochastic operator F mapping signals X to their associated observations Y. We will refer to F as the *measurement process*, which *corrupts* the input signal. F is parameterized by a random variable $\Theta \sim \mathrm{p}_\Theta$ following an underlying distribution p_Θ we can sample from, which represents the factors of corruption. Thus, given a specific signal x, we can simulate its measurement by first sampling θ from p_Θ, and then computing $F(x;\theta)$. Additional sources of uncertainty, *e.g.* due to unknown factors, can be modeled using additive *i.i.d.* Gaussian noise $\mathcal{E} \sim \mathcal{N}(0, \sigma^2 I)$, so that the overall observation model becomes:

$$Y = F(X;\Theta) + \mathcal{E} \tag{S1}$$

F is assumed to be differentiable *w.r.t.* its first argument X, and Θ and X to be independent (denoted $X \perp\!\!\!\perp \Theta$). Different instances of F will be considered (refer to Section 5.1.4.2), like random occlusions, information acquisition from a sparse subset of the signal, overly smoothing out and corrupting the original distribution with additive noise, etc. In such cases, the factors of corruption Θ might respectively represent the position of the occlusion, the coordinates of the acquired information, or simply the values of the additive noise.

5.1.2.2 Approach

Given an observation y, our objective is to find a signal $\hat{x}$ as close as possible to the associated true signal x. From a probabilistic viewpoint, it is natural to formulate the problem as finding the *maximum a posteriori* (MAP) estimate, which consists in selecting the most probable signal x^* under the posterior distribution $\mathrm{p}_{X|Y}(\cdot|y)$:

$$x^* = \arg\max_x \log \mathrm{p}_{X|Y}(x|y) \tag{S2}$$

or equivalently:

$$x^* = \arg\max_x \log \mathrm{p}_{Y|X}(y|x) + \log \mathrm{p}_X(x) \tag{S3}$$

where $\mathrm{p}_{Y|X}(y|x)$ is the likelihood of the signal x given observation y, and $\mathrm{p}_X(x)$ is the prior probability evaluated at x. Therefore, a good reconstruction must be likely to have generated the data, *i.e.* yield high likelihood, and look realistic, *i.e.* yield high probability under the prior.

In the general case, calculating the likelihood term $\mathrm{p}_{Y|X}(y|x)$ requires marginalizing over noise parameters Θ and this does not yield an analytic form. As for the prior $\mathrm{p}_X(x)$, it is unknown, and we have no access to samples from X since we are in an unsupervised setting: there is then no direct way to estimate p_X either. In the general case considered here, with no assumption on the form of the distributions, solving Equation (S3) is up to our knowledge an open problem.

In the following sections, we will introduce an approach to deal with the likelihood term (Section 5.1.3.1), and the unknown prior term (Section 5.1.3.2) in order to provide an approximate solution to equation (S3) (Section 5.1.3.3). For that, we will formulate the problem as learning a mapping $G : \mathcal{Y} \to \mathcal{X}$ that links each measurement y to its associated MAP estimate x^* on the training set. The associated objective is then:

$$G^* = \arg\max_G \mathbb{E}_{\mathrm{p}_Y} \left\{ \log \mathrm{p}_{Y|X}(y|G(y)) + \log \mathrm{p}_X(G(y)) \right\} \tag{S4}$$

Which is obtained by plugging $G(y) = x$ into equation S3 and taking the expectation w.r.t. the distribution of observations p_Y.

5.1.3 Method

From Equation (S4), we see that a valid reconstruction mapping G must yield high probability for the likelihood and the prior. This will guide the design of an appropriate objective during the following section, where the reconstruction mapping G will be implemented using a neural network.

5.1.3.1 Handling the Likelihood term

In the general case, evaluating the likelihood $\mathrm{p}_{Y|X}(y|x)$ in equation (S3) requires marginalizing on the unobserved noise variable Θ: $\mathrm{p}_{Y|X}(y|x) = \mathbb{E}_{\mathrm{p}_\Theta} \mathrm{p}_{Y|X,\Theta}(y|x,\theta)$, which involves computing an intractable integral. Most probabilistic model for image denoising make assumptions on the structure of the measurement operator $F(.,\Theta)$ and on the distribution of Θ in order to obtain an analytic form for the expectation (Boyat et al. 2015, Alkinani et al. 2017). Here, we consider more general measurement operators which do not necessarily lead to such a simplification and therefore proceed in a different way.

We outline below, the main steps of the method for handling the likelihood term $\mathrm{p}_{Y|X}(y|x)$ in equation (S4). The complete derivation is provided in Appendix C.1.1.

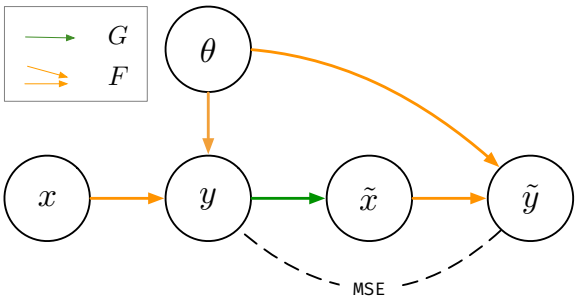


Figure S1. – The Figure illustrates the dependencies between the variables considered for handling the likelihood term when solving (S4). The likelihood term in (S4) can be replaced by the expectation of $\frac{1}{2\sigma^2}\|y - F(G(y);\theta)\|_2^2$ (see Equation S7). To compute this expectation, one first simulates an observation y from a signal x using $F(x;\theta)$, then generates $\tilde{x} = G(y)$ and $\tilde{y} = F(\tilde{x};\theta)$, as in the Figure. This allows us to compute the MSE term in the above expression.

1. Making use of the independence between X and Θ, the expectation term $\mathbb{E}_{\mathrm{p}_Y} \log \mathrm{p}_{Y|X}(y|G(y))$ in equation (S4) can be rewritten as :

$$\mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X \mathrm{p}_{Y|X,\Theta}} \log \mathrm{p}_{Y|X,\Theta}(y|G(y),\theta) + c_1 \tag{S5}$$

, with c_1 constant w.r.t. G.

2. The general measurement process described in equation (S1) induces $\log \mathrm{p}_{Y|X,\Theta}(y|G(y),\theta)$ to yield a simple analytic expression:

$$\log p(y|G(y),\theta) = -\frac{1}{2\sigma^2}\|y - F(G(y);\theta)\|_2^2 + c_2 \tag{S6}$$

with c_2 a constant.

3. The likelihood term $\mathbb{E}_{\mathrm{p}_Y} \log \mathrm{p}_{Y|X}(y|G(y))$ can then be replaced in objective (S4) by

$$-\mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X \mathrm{p}_{Y|X,\Theta}} \frac{1}{2\sigma^2}\|y - F(G(y);\theta)\|_2^2 \tag{S7}$$

Equation (S7) shows that the likelihood term can be evaluated by first sampling a measurement y conditioned on a corruption parameter θ and signal x, and then constrain G such that $\|y - F(G(y);\theta)\|_2^2$ is close to zero. Note that in this expression, the same parameter θ is used for simulating $\tilde{y}$ from $\tilde{x}$ and y from x (see Figure S1 and section 5.1.3.3 for more details). Unfortunately, this requires first sampling x from the signal distribution p_X which is unknown. In the following sections, we will see how we work around this problem.

5.1.3.2 Handling the Prior term

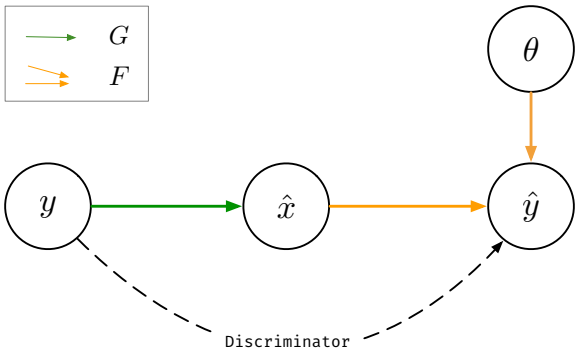


Figure S2. – The figure illustrates the dependencies of the variables used for dealing with the prior term in (S4). An observation y is sampled, and then transformed by the generative network into a reconstructed signal $\hat{x} = G(y)$. One then simulates a measurement $\hat{y} := F(\hat{x}; \theta)$ from this reconstruction. We then enforce the distributions of observations p_Y and simulated measurements p_Y^G to be similar using an adversarial loss. In order to produce indistinguishable distributions, the generator G has to *remove* the corruption and recover a sample $\hat{x}$ from p_X.

Maximizing w.r.t. the prior term $\mathrm{p}_X(G(y))$ in equation (S4) is similar to learning a mapping G, such that the distribution induced by $G(y)$, $\mathbb{E}_{\mathrm{p}_Y}\mathrm{p}_X(G(y))$ is close to the distribution p_X. The prior p_X being unknown, the only sources of information are the lossy measurements y and the known prior p_Θ on the measurement process. In order to learn an approximation of the true prior p_X, we will use a form of generative adversarial learning, and build on an idea introduced in the AmbientGAN model by Bora et al. 2018b.

AmbientGAN aims at learning an *unconditional* generative model G of the true signal distribution p_X, when only lossy measurements y of the signal are available together with a known stochastic measurement operator F. In AmbientGAN, a generator is trained to produce uncorrupted signal samples from a latent code so that the generated signals when corrupted are indistinguishable from the observation measurements. In Bora et al. 2018b, the authors show that for some families of noise distributions p_Θ, the generator's induced distribution matches the signal's true distribution. Note that even if the generation process of the observations y in AmbientGAN is similar to the one considered in this paper (see Section 5.1.2.1), the objective is however different: when the aim of AmbientGAN is to learn a distribution of the underlying signal by sampling a latent space, ours is to reconstruct corrupted signals.

In order for G to produce uncorrupted signals we will use an approach inspired from AmbientGAN, as illustrated in Figure S2. Given an observation y, one wants to reconstruct a latent signal approximation $\hat{x} = G(y)$ so that a corrupted version of this signal $\hat{y} = F(\hat{x})$ will have a distribution indistinguishable from the one of the observations y. The generator G and a discriminator D are trained on observations y and generated samples $\hat{y}$. The corresponding loss is the following [1]:

$$\mathcal{L}^{\text{prior}}(G) := \max_D \; \mathbb{E}_{Y\sim \mathrm{p}_Y, \hat{Y}\sim \mathrm{p}_Y^G} \big\{ \log D(y) + \log\big(1 - D(\hat{y})\big) \big\} \tag{S8}$$

where p_Y^G corresponds to the distribution induced by G's corrupted outputs ($\hat{y}$ in Figure S2) , *i.e.* $\mathrm{p}_Y^G(y) := \mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X^G}\big\{\mathrm{p}(y|x,\theta)\big\}$ and p_X^G denotes the marginal distribution induced by G's outputs ($\hat{x}$ in Figure S2): $\mathrm{p}_X^G(x) := \mathbb{E}_{\mathrm{p}_Y} \mathrm{p}_{X|Y}^G(x|y) = \mathbb{E}_{\mathrm{p}_Y} \delta\big(x - G(y)\big)$ [2]. This penalty enforces the marginal p_X^G to be close to the true prior distribution p_X, and thus forces G to map its input measurements onto p_X.

5.1.3.3 Putting everything together

In Section 5.1.3.1, we have shown that it is possible to maximize the average log-likelihood term in equation (S4), given that we can sample from the unknown prior distribution p_X. In Section 5.1.3.2, we have shown how it is possible to enforce the generator to produce samples from p_X, without ever having access to uncorrupted samples. The idea is then to use the distribution induced by the generator's output p_X^G as a proxy for p_X to compute an approximate value of the expectation in equation (S5): This gives us the following penalty term (see appendix C.1.1):

$$\mathcal{L}^{\text{likeli}}(G) := \mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X^G, \hat{Y}\sim \mathrm{p}_{Y|X,\Theta}} \|\hat{y} - F(G(\hat{y});\theta)\|_2^2 \tag{S9}$$

The full objective is a linear combination of penalties (S8) and (S9):

$$\arg\min_G \; \mathcal{L}^{\text{prior}}(G) + \lambda \cdot \mathcal{L}^{\text{likeli}}(G) \tag{S10}$$

As illustrated by the dependencies highlighted in Figure S2, in the process of minimizing $\mathcal{L}^{\text{prior}}$, we sample from the marginal likelihood $\mathrm{p}_Y^G(y) := \mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X^G}\big\{\mathrm{p}(y|x,\theta)\big\}$. The expectancy in the likelihood term $\mathcal{L}^{\text{likeli}}$ is precisely computed w.r.t. this distribution. We can then use the same samples in order to minimize the full objective (S10). This gives us the Algorithm 5 described below, along with the dependency structure illustrated in Figure S3.

1. the min term of the adversarial loss will be introduced later, see Equation (S10)
2. $\delta(x)$ is the Dirac delta function, which is equal to zero everywhere except in x.

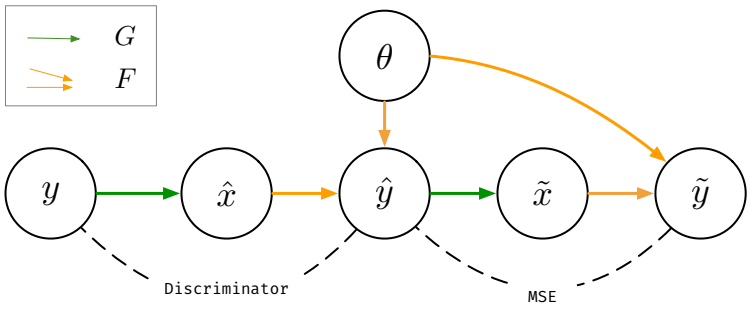


Figure S3. – General Approach. We wish to train G to recover a plausible signal from lossy measurements. As is shown in Section 5.1.2.2, this requires the reconstructions $\hat{x} := G(y)$ to have high probability under the likelihood and the prior. For simplicity, variable $\mathcal{E}$ has been omitted. *Prior :* we sample a measurement y from the data, produce a reconstruction $\hat{x}$, and sample a perturbation parameter θ. We enforce the simulated measurement $\hat{y} := F(\hat{x}; \theta)$ to be similar to measurements in the data using an adversarial penalty. Intuitively, this requires the network to *remove* the corruption. *Likelihood :* to enforce G to produce reconstructions with high likelihood, it is not possible to add a penalty to constrain the mean square error (MSE) between y and $\hat{y}$ to be small. This is because the underlying perturbation that caused y is unknown, and may be different from θ. Starting from $\hat{y}$ we generate a $\tilde{y}$ (see figure S3) using the same θ as the one used for generating $\hat{y}$. We then constrain $\|\hat{y} - \tilde{y}\|_2^2$ to be small ($\tilde{y} = F(G(\hat{y}))$.

Algorithm 5 Training Procedure.

Initialize parameters of the generator G and the discriminator D.
while (G, D) not converged **do**
  Sample $\{y_i\}_{1 \leq i \leq n}$ from data distribution p_Y
  Sample $\{\theta_i\}_{1 \leq i \leq n}$ from P_Θ
  Sample $\{\varepsilon_i\}_{1 \leq i \leq n}$ from $P_\mathcal{E}$
  Set $\hat{y}_i$ to $F(G(y_i), \theta_i) + \varepsilon_i$ for $1 \leq i \leq n$
  Update D by ascending:

$$\frac{1}{n} \sum_{i=1}^{n} \log D(y_i) + \log(1 - D(\hat{y}_i))$$

  Update G by descending:

$$\frac{1}{n} \sum_{i=1}^{n} \lambda \cdot \|\hat{y}_i - F(G(\hat{y}_i); \theta_i)\|_2^2 + \log(1 - D(\hat{y}_i))$$ [3]

end while

3. In practice we optimize $-\log D(\hat{y}_i)$ instead of $\log(1 - D(\hat{y}_i))$

5.1.4 Experiments

5.1.4.1 Model architectures and Datasets

Architectures. We will briefly describe the architectures, additional details on architectures and hyperparameters can be found in appendix C.1.2. Our network architectures are inspired by the GAN architecture in H. Zhang et al. 2018. We use the same discriminator, and we propose an image-to-image variant of their latent-to-image generator for the reconstruction network G.

Datasets. We evaluate our approach using three different image datasets :

- **CelebA.** Dataset of celebrities, containing approximately 200 000 samples. As Bora et al. 2018b, the images are center-cropped.
- **LSUN Bedrooms.** Dataset of bedrooms, containing 3 million samples.
- **Recipe-1M.** Dataset of cooked meals, containing approximately 600 000 samples.

All the images have been resized to 64×64. In order to place ourselves in the most realistic setting possible, every image has been corrupted once, *i.e.* there is never multiple occurrences of an image corrupted with different corruption parameters.

We withhold 15% of the training set for validation, selected uniformly at random for each dataset.

5.1.4.2 Corruptions

Let us present the different measurement processes F used in the experiments, also named corruptions:

Remove-Pixel. This measurement process randomly samples a fraction p of pixels uniformly and sets the associated channel values to 0. All the corresponding channel values are set to 0.

Remove-Pixel-Channel. Instead of setting to 0 a pixel for all channels as in Remove-Pixel, one samples a pixel coordinate and a channel, and sets the corresponding value to 0.

Convolve-Noise. Here $F(x;\theta) := k * x + \theta$, where $*$ is the convolution operator and k is a mean filter of size l. For each pixel, noise θ sampled from a zero-mean Gaussian of variance σ_C^2 is added to the previous result.

Patch-Band. A horizontal band of height h whose vertical position in the image is uniformly sampled from the set of possible positions. For each pixel falling inside the band, its associated value is set to 0. The resulting measurement for pixel at column i and row j can be summarized as:

$$F(x;\theta)_{i,j} := \begin{cases} 0, & \text{if } j \in \{\theta, \dots, \theta + h\} \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{S11}$$

where θ is uniformly sampled from $\{1, \dots, H - h\}$, and H is the image height. In the experiments, h is set to 20.

5.1.4.3 Baselines

Conditional AmbientGan.

This is our only unsupervised baseline. The context is the same as for our model: the measurement process F is assumed known, there is no access to samples from the uncorrupted signal distribution p_X, but only to their corrupted counterpart p_Y. This baseline is a combination of two recent techniques in the field of signal recovery: the aforementioned AmbientGan Bora et al. 2018b and CS-GAN Bora et al. 2017.

An unconditional generator G is trained using the AmbientGan framework (Bora et al. 2018b) for each type of measurement process F, in order to produce samples from p_X (see Section 5.1.4.2). The distribution induced by the generator p_X^G is an approximation of p_X (at the optimum, both distributions match, *i.e.* $\mathrm{p}_X^G = \mathrm{p}_X$). Given a specific measurement y, the reconstruction $\hat{x}$ is the signal from G that is closest to y, as in Bora et al. 2017. To find $\hat{x} = G(\hat{z})$, we search for the latent code $\hat{z}$ of G, such that $\hat{z} = \arg\min_z \|y - G(z)\|_2^2 + R(z)$. $R(z)$ is a regularizing term that enforces the latent code to stay in G's input domain. This objective is optimized using stochastic gradient descent. To train G, we use the same architectures and hyper-parameters as those provided by the authors. Because this approach may be sensitive to the initial latent code, we reiterate this approach three times and select the best resulting image.

Unpaired Variant.

This is a variant of our model where we have access to samples of the signal distribution p_X. This means that although we have no paired samples from $\mathrm{p}_{X,Y}$, we have access to unpaired samples from p_X and p_Y. This baseline is similar to our model but instead of discriminating between a measurement from the data y and a simulated measurement $\hat{y}$, we directly discriminate between samples x from the signal distribution and the output of the reconstruction network $\hat{x}$. For a diagram describing the model, refer to appendix C.1.3.1.

Paired Variant.

This is a variant of our model where we have access to signal measurement pairs (y, x) from the joint distribution $\mathrm{p}_{Y,X}$. Given input measurement y, the reconstruction is obtained by regressing y to the associated signal x using a MSE loss. In order to avoid blurry samples, we add an adversarial term in the objective in order to constrain G to produce realistic samples, as in Isola et al. 2016b. The model is trained using the same architectures as our model, and the hyperparameters have been found using cross-validation. For a diagram describing the model, refer to appendix C.1.3.2.

Measurement Specific Baselines.

We also compare our model to baselines that where designed to remove specific corruptions.

Deep Image Prior (Ulyanov et al. 2017). Given a generator G_ϕ parametrized by randomly initialized weights ϕ and a measurement y, this method seeks to find a reconstruction from G_ϕ that is *close* to the measurement. For corruptions processes Patch-Band, Remove-Pixel and Remove-Pixel-Channel, we assume the access to the θ associated to the observations in the data (*i.e.* in this case, the mask). For more details, please refer to appendix C.1.3.

Biharmonic Inpainting (Damelin et al. 2018). By considering inpainting as a smooth surface extension domain, this baseline resolves a biharmonic equation to obtain a high order approximation of the image. This approximation is then extended to the missing part of the image. This method assumes access to the θ associated to the observations in the data (*i.e.* in this case, the mask).

Total Variation Denoising (Chambolle 2004). This denoising baseline aims to minimize the total variation of an image i.e the integral of the absolute gradient

of the image. Reducing the total variation of the image removes unwanted detail, such as white noise artifacts while preserving important details such as edges and corners.

5.1.5 Results

We will now present our results. First, we compare quantitatively our model with non-measurement specific baselines on CelebA. We then present qualitative results with samples from our model and these baselines. Comparisons with measurement specific baselines are presented in appendix C.1.4 for the three datasets.

5.1.5.1 Quantitative Results

We compare our model with baselines introduced in the previous section. We report mean square error (MSE) scores between the reconstructed $\hat{x}$ and the true signal x used to generate the input y. Table S1 shows the MSE computed on the *test* set, a randomly selected subset of CelebA comprised of 40000 images. Because the Conditional AmbientGan model is too computationally expensive, we only report the MSE on 40 randomly chosen samples of the test set.

Table S1. – Average Mean Squared Error (MSE) of neural network based models on the test set of CelebA, for different measurement processes. The first two rows are models trained with no supervision, the last two rows with additional supervision.

	Remove-Pixel	Remove-Pixel-Channel	Patch-Band	Convolve-Noise
Conditional AmbientGan	0.292	0.2829	0.1421	0.0814
Our Deterministic Model	**0.0414**	**0.0409**	**0.0165**	**0.0088**
Unpaired Variant	**0.037**	**0.0336**	0.034	0.0103
Paired Variant	0.0383	0.0401	**0.0147**	**0.0084**

Quantitatively, our model performs well. Except for the Conditional Ambiant GAN, all the methods are quite similar in terms of MSE. Our unsupervised model reaches performance similar to its variants trained using additional supervision. We also note that when the aligned signal-observation pairs are not used (as in our Unpaired Variant), results are comparable – sometimes better – than when these pairs are used directly (as in our Paired Variant). This suggests that our likelihood term is sufficient to condition the reconstruction on the input signal.

5.1.5.2 Qualitative Results

However, quantitative results gives us only partial information. We now evaluate the quality of our reconstruction on three different datasets (Section 5.1.4.1). Figure S4 shows reconstructions obtained from different models on the CelebA dataset.We observe that Conditional AmbientGAN yields visually poor results, especially for the Remove-Pixel and Remove-Pixel-Channel measurement processes. We hypothesize that this is due to the large Euclidean distance between the measurements and the associated signals, and the suboptimality of the generator. Visually, the quality of our model's reconstructions are coherent with the quantitative results: they are comparable to its paired and unpaired counterparts (Section 5.1.4.3). Figures (S5), (S6), (S7), and (S8) each show reconstructions from a given measurement processe on different datasets. Our model is able to produce images with good visual quality while remaining coherent with the underlying uncorrupted images. In Figures (S3), (S4), (S5) and (S6) in appendix C.1.4, we compare our model with commonly used inpainting or denoising methods. We can see that contrary to these methods, we are able to capture semantic information from the dataset. Typically, in Figure S6, the model infers missing eyes or noses, without ever having seen them. Additional samples are available in the appendix C.1.4, refer to Figures (S7), (S8), (S9), and (S10).

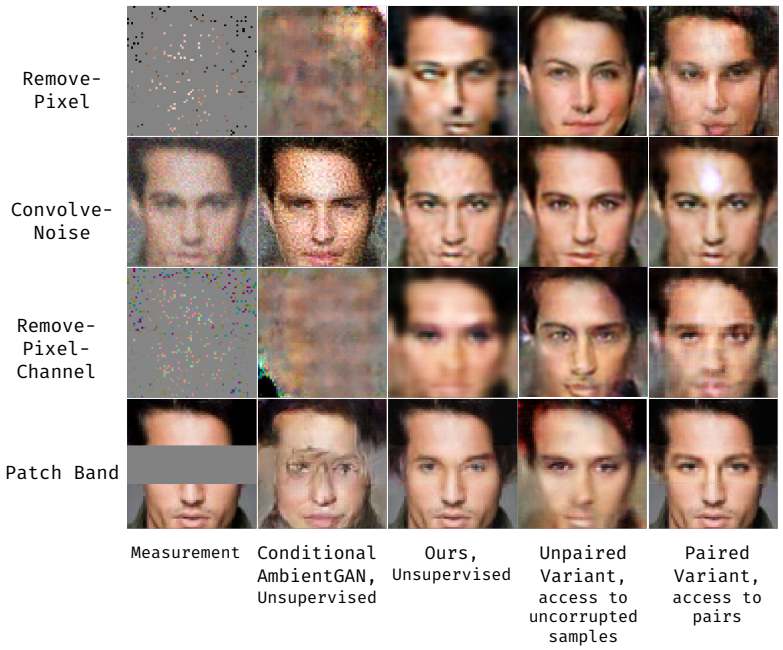


Figure S4. – Model reconstructions for different corruption processes, on CelebA. Each row corresponds to a specific corruption process, and each column to a particular model.

Figure S5. – On the top row, randomly sampled test set measurements from CelebA corrupted using Patch-Band($h = 20$), and below, our associated reconstructions.

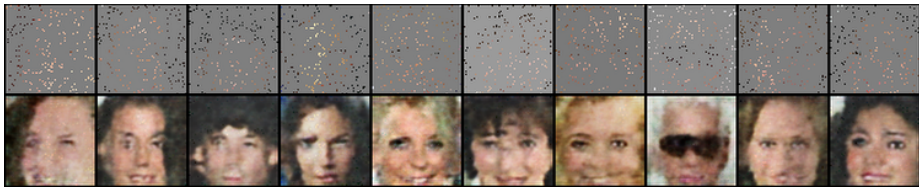


Figure S6. – On the top row, randomly sampled test set measurements from CelebA corrupted using Remove-Pixel($p = 0.95$), and below, our associated reconstructions.

Figure S7. – On the top row, randomly sampled test set measurements from LSUN corrupted using Patch-Band($h = 20$), and below, our associated reconstructions.

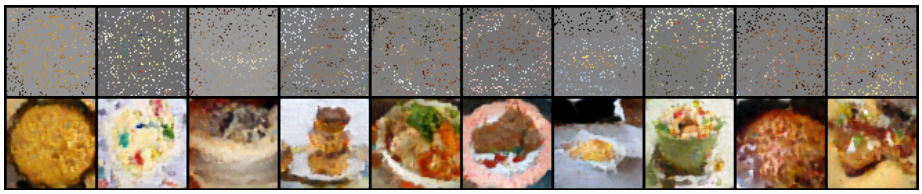


Figure S8. – On the top row, randomly sampled test set measurements from Recipe-1M corrupted using Remove-Pixel ($p = 0.9$), and below, our associated reconstructions.

5.1.6 Related Work

To our knowledge, there is no other Deep Learning approach attempting to solve the unsupervised signal reconstruction problem. However, some of the ideas developed here are close to or even inspired from recent work.

In order to enforce high likelihood, we incorporate a penalty in our objective that is similar to the Cycle Consistency loss, used in several contexts (J.-Y. Zhu et al. 2017, Lample et al. 2017, Almahairi et al. 2018). This constraint is used to learn from unpaired data sets. Moreover, they too use adversarial training to constrain the marginal distribution induced by the generator.

In the context of image super resolution, (Ledig et al. 2016, Sønderby et al. 2016, Mardani et al. 2017) attempt to retrieve *maximum a posteriori* estimates of the super resolution image conditioned on an input image. They too use a generative model of the signal trained in an adversarial fashion using samples from signal distribution to constrain their reconstructions. Their approach is fully supervised.

Other works attempt to solve ill-posed inverse problems using generative models (Bora et al. 2017, Asim et al. 2018, Tripathi et al. 2018, Van Veen et al. 2018). The general approach in all these papers consists to first train a generative model on the uncorrupted signal distribution. Then, given a measurement from which we wish to reconstruct the signal, it is *inverted* by finding the latent input code that generated the uncorrupted image, by minimizing the mean square error between the corrupted reconstruction and the measurement. This requires solving an optimization problem for each image, which takes several minutes (Ulyanov et al. 2017) on GPU, and requires random restarts to avoid falling a bad local minima. Again, the setting is fully supervised.

Finally Lehtinen et al. 2018 propose a method for denoising images without direct supervision. They train a network to regress a corrupted image to the same image with a different corruption value. Assuming the corruption has zero-mean, their network learns to remove the corruption by the conditional expectation. This setting implicitly assumes access to the distribution of uncorrupted images in order to generate different noisy versions of the same image, which is not our case.

5.1.7 Conclusion

We have proposed a general formulation to recover a signal from lossy measurements using a neural network, without having access to uncorrupted signal data. We have formulated the problem as finding a *maximum a posteriori* estimate of the signal given its observation, for all observations in the training set. This gives us a natural objective for our neural network, composed of a linear combination of an adversarial loss for recovering realistic signals, and a reconstruction loss to tie the reconstruction to its associated observation. Our approach yields results superior to the baselines, while staying competitive with other model variants that have access to higher forms of supervision.

For future work, we plan to apply our framework to different corruption processes, and evaluate our model's performance in real world settings, specifically for retrieving uncorrupted scientific data. Another interesting research direction would be to make our reconstruction network stochastic, in order to approximate the true posterior of the signal given the measurement, and to obtain uncertainty estimates.

5.2 A Neural Tangent Kernel Perspective of GANs

abstract

Theoretical analyses for Generative Adversarial Networks (GANs) generally assume an arbitrarily large family of discriminators and do not consider the characteristics of the architectures used in practice. We show that this framework of analysis is too simplistic to properly analyze GAN training. To tackle this issue, we leverage the theory of infinite-width neural networks to model neural discriminator training for a wide range of adversarial losses via its Neural Tangent Kernel (NTK). Our analytical results show that GAN trainability primarily depends on the discriminator's architecture. We further study the discriminator for specific architectures and losses, and highlight properties providing a new understanding of GAN training. For example, we find that GANs trained with the integral probability metric loss minimize the maximum mean discrepancy with the NTK as kernel. Our conclusions demonstrate the analysis opportunities provided by the proposed framework, which paves the way for better and more principled GAN models. We release a generic GAN analysis toolkit based on our framework that supports the empirical part of our study.

5.2.1 Introduction

Generative Adversarial Networks GANs; I. Goodfellow et al. 2014 have become a canonical approach to generative modeling as they produce realistic samples for numerous data types, with a plethora of variants Z. Wang et al. 2021. These models are notoriously difficult to train and require extensive hyperparameter tuning Brock et al. 2019; Karras et al. 2020; M.-Y. Liu et al. 2021. To alleviate these shortcomings, much effort has been put in gaining a better understanding of the training process, resulting in a vast literature on theoretical analyses of GANs (see related work below). A large portion of them focus on studying GAN loss functions to conclude about their comparative advantages.

Yet, empirical evaluations Lucic et al. 2018; Kurach et al. 2019 have shown that different GAN formulations can yield approximately the same performance in terms of sample quality and stability of the training algorithm, regardless of the chosen loss. This indicates that by focusing exclusively on the formal loss function, theoretical studies might not model practical settings adequately.

In particular, the discriminator being a trained neural network is not taken into account, nor are the corresponding inductive biases which might considerably alter the generator's loss landscape. Moreover, neglecting this constraint hampers the analysis of gradient-based learning of the generator on finite training sets, since the gradient from the associated discriminator is ill-defined everywhere. These limitations thus hinder the potential of theoretical analyses to explain GAN's empirical behaviour.

In this work, leveraging the recent developments in the theory of deep learning driven by Neural Tangent Kernels NTKs; Jacot et al. 2018c, we provide a framework of analysis for GANs incorporating explicitly the discriminator's architecture which comes with several advantages.

First, we prove that, in the proposed framework, under mild conditions on its architecture and its loss, the trained discriminator has strong differentiability properties; this result holds for several GAN formulations and standard architectures, thus making the generator's learning problem well-defined. This emphasizes the role of the discriminator's architecture in GANs trainability.

We then show how our framework can be useful to derive both theoretical and empirical analyses of standard losses and architectures. We highlight for instance links between Integral Probability Metric (IPM) based GANs and the Maximum Mean Discrepancy (MMD) given by the discriminator's NTK, or the role of the ReLU activation in GAN architectures. We evaluate the adequacy and practical implications of our theoretical framework and release the corresponding Generative Adversarial Neural Tangent Kernel ToolKit GAN(TK)[2].

Related Work

GAN Theory. A first line of research, started by I. Goodfellow et al. (2014) and pursued by many others Nowozin et al. 2016; Zhou et al. 2019; Sun et al. 2020, studies the loss minimized by the generator. Assuming that the discriminator is optimal and can take arbitrary values, different families of divergences can be recovered. However, as noted by Arjovsky et al. (2017a), these divergences should be ill-suited to GANs training, contrary to empirical evidence. We build up on this observation and show that under mild conditions on the discriminator's architecture, the generator's loss and gradient are actually well-defined.

Another line of work analyzes the dynamics and convergence of the generated distribution Nagarajan et al. 2017; Mescheder et al. 2017;

Mescheder et al. 2018. As the studied dynamics are highly non-linear, this approach typically requires strong simplifying assumptions, e.g. restricting to linear neural networks or reducing datasets to a single datapoint. More recently, Mroueh et al. (2021) proposed to study GANs using RKHSs, and Jacot et al. (2019) improve generator training by investigating checkerboard artifacts in the light of NTKs. The most advanced modelizations taking into account discriminator's parameterization are specialized to specific models Bai et al. 2019, such as a linear one, or random feature models S. Liu et al. 2017; Balaji et al. 2021. In contrast to these works, we are able to establish generally applicable results about the influence of the discriminator's architecture.

By taking into account the parameterization of discriminators for a wide range of architectures, our framework of analysis provides a more complete modelization of GANs.

Neural Tangent Kernel. NTKs were introduced by Jacot et al. (2018c), who showed that a trained neural network in the infinite-width regime equates to a kernel method, hereby making the dynamics of the training algorithm tractable and amenable to theoretical study. This fundamental work has been followed by a thorough line of research generalizing and expanding its initial results Arora et al. 2019b; Bietti et al. 2019a; J. Lee et al. 2019b; C. Liu et al. 2020; Sohl-Dickstein et al. 2020, developing means of computing NTKs Roman Novak et al. 2020; G. Yang 2020, further analyzing these kernels Fan et al. 2020; Bietti et al. 2021; L. Chen et al. 2021, studying and leveraging them in practice Zhou et al. 2019; Arora et al. 2020; J. Lee et al. 2020; Littwin et al. 2020b; Tancik et al. 2020, and more broadly exploring infinite-width networks Littwin et al. 2020a; G. Yang et al. 2020; Alemohammad et al. 2021. These prior works validate that NTKs can encapsulate the characteristics of neural network architectures, providing a solid theoretical basis to study the effect of architecture on learning problems.

While other works have studied the regularity of NTKs Bietti et al. 2019a; G. Yang et al. 2019; Basri et al. 2020, as far as we know, ours is the first to state general derivability results for NTKs and infinite-width networks, as well as the first to leverage the theory of NTKs to study GANs.

5.2.2 Modeling GAN's Discriminator

We present in this section the usual GAN formulation and learning procedure, illustrate the limitations of prior analyses and introduce our framework which we develop in the remaining of the paper.

First, we introduce some notations. Let $\Omega \subseteq \mathcal{R}^n$ be a closed convex set, $\mathcal{P}(\Omega)$ the set of probability distributions over Ω, and $L^2(\mu)$ the set of square-integrable functions from the support $\operatorname{supp}\mu$ of μ to $\mathcal{R}$ with respect to measure μ, with scalar product $\langle \cdot, \cdot \rangle_{L^2(\mu)}$. If $\Lambda \subseteq \Omega$, we write $L^2(\Lambda)$ for $L^2(\lambda)$, with λ the Lebesgue measure on Λ.

5.2.2.1 Generative Adversarial Networks

GAN algorithms seek to produce samples from an unknown target distribution $\beta \in \mathcal{P}(\Omega)$. To this extent, a generator function $g \in \mathcal{G} \colon \mathcal{R}^d \to \Omega$ parameterized by θ is learned to map a latent variable $z \sim p_z$ to the space of target samples such that the generated distribution α_g and β are indistinguishable for a discriminator network $f \in \mathcal{F}$ parameterized by ϑ. The generator and the discriminator are trained in an adversarial manner as they are assigned conflicting objectives.

Many GAN models consist in solving the following optimization problem, with $a, b, c \colon \mathcal{R} \to \mathcal{R}$:

$$\inf_{g \in \mathcal{G}} \Big\{ \mathcal{C}_{f^\star_{\alpha_g}}(\alpha_g) \triangleq \mathbb{E}_{x \sim \alpha_g} \Big[c_{f^\star_{\alpha_g}}(x) \Big] \Big\}, \tag{S12}$$

where $c_f = c \circ f$, and $f^\star_{\alpha_g}$ is chosen to solve, or approximate, the following optimization problem:

$$\sup_{f \in \mathcal{F}} \Big\{ \mathcal{L}_{\alpha_g}(f) \triangleq \mathbb{E}_{x \sim \alpha_g}[a_f(x)] - \mathbb{E}_{y \sim \beta}[b_f(y)] \Big\}. \tag{S13}$$

For instance, I. Goodfellow et al. (2014) originally used $a(x) = \log(1 - \sigma(x))$, $b(x) = c(x) = -\log(\sigma(x))$; in LSGAN Mao et al. 2017, $a(x) = -(x+1)^2$, $b(x) = (x-1)^2$, $c(x) = x^2$; and for Integral Probability Metrics IPMs; Müller 1997 leveraged for example by Arjovsky et al. (2017b), $a = b = c = \mathrm{id}$. Many more fall under this formulation Nowozin et al. 2016; Lim et al. 2017.

Equation (S12) is then solved using gradient descent on the generator's parameters:

$$\theta_{j+1} = \theta_j - \eta \mathbb{E}_{z \sim p_z} \Big[\nabla_\theta g_{\theta_j}(z)^T \nabla_x c_{f^\star_{\alpha_g}}(x) \Big|_{x = g_{\theta_j}(z)} \Big]. \tag{S14}$$

Since $\nabla_x c_{f^\star_\alpha}(x) = \nabla_x f^\star_\alpha(x) \cdot c'(f^\star_\alpha(x))$, and as highlighted e.g. by I. Goodfellow et al. (2014) and Arjovsky et al. (2017a), the gradient of the discriminator plays a crucial role in the convergence of training. For example, if this vector field is null on the training data when $\alpha \neq \beta$, the generator's gradient is zero and convergence is impossible. For this reason, the following sections are devoted to developing a better understanding of this gradient field when the discriminator is a neural network. In order to characterize the discriminator's gradient field, we must first study the discriminator itself.

5.2.2.2 On the Necessity of Modeling the Discriminator Parameterization

For each GAN formulation, it is customary to elucidate which loss is implemented by Equation (S13), often assuming that $\mathcal{F} = L^2(\Omega)$, i.e. the discriminator can take arbitrary values. Under this assumption, the original GAN yields the Jensen-Shannon divergence between α_g and β, and LSGAN a Pearson χ^2-divergence, for instance.

However, as pointed out by Arora et al. (2017), the discriminator is trained in practice with a finite number of samples: both fake and target distributions are finite mixtures of Diracs, which we respectively denote as $\hat{\alpha}$ and $\hat{\beta}$. Let $\hat{\gamma} = \frac{1}{2}\hat{\alpha} + \frac{1}{2}\hat{\beta}$ be the distribution of training samples.

Assumption 1. $\hat{\gamma} \in \mathcal{P}(\Omega)$ is a finite mixture of Diracs.

In this setting, the Jensen-Shannon and χ^2-divergence are constant since $\hat{\alpha}$ and $\hat{\beta}$ generally do not have the same support. This is the theoretical reason given by Arjovsky et al. (2017a) to introduce new losses, such as in WGAN Arjovsky et al. 2017b. However, this is inconsistent with empirical results showing that GANs can be trained with these losses. Actually, perhaps surprisingly, in the alternating optimization setting used in practice – as described by Equation (S14) – the constancy of $\mathcal{L}_{\hat{\alpha}}$ does not imply that $\nabla_x c_{f^\star_\alpha}$ in Equation (S14) is zero on these points; see Section 5.2.4.2 and Appendix C.2.2.2 for further discussion on this matter. Yet, in their theoretical framework where the discriminator can take arbitrary values, this gradient field is not even defined for any loss $\mathcal{L}_{\hat{\alpha}}$.

Indeed, when the discriminator's loss $\mathcal{L}_{\hat{\alpha}}(f)$ is only computed on the empirical distribution $\hat{\gamma}$ (as it is the case for most GAN formulations), the discriminator optimization problem of Equation (S13) never yields a unique optimal solution outside $\hat{\gamma}$. This is illustrated by the following straightforward result.

Proposition S9 (Ill-Posed Problem in $L^2(\Omega)$)**.** *Suppose that* $\mathcal{F} = L^2(\Omega)$, $\operatorname{supp}\hat{\gamma} \subsetneq \Omega$. *Then, for all* $f, h \in \mathcal{F}$ *coinciding over* $\operatorname{supp}\hat{\gamma}$, $\mathcal{L}_{\hat{\alpha}}(f) = \mathcal{L}_{\hat{\alpha}}(h)$ *and Equation* (S13) *has either no or infinitely many optimal solutions in* $\mathcal{F}$, *all coinciding over* $\operatorname{supp}\hat{\gamma}$.

In particular, the set of solutions, if non-empty, contains non-differentiable discriminators as well as discriminators with null or non-informative gradients. This underspecification of the discriminator over Ω makes the gradient of the optimal discriminator in standard GAN analyses ill-defined.

This signifies that the loss alone does not impose any constraint on the values that $f_{\hat{\alpha}}$ takes outside $\operatorname{supp}\hat{\gamma}$, and more particularly that there are no constraints on the gradients. Therefore, an analysis beyond the loss function is necessary to precisely define the learning problem of the generator.

5.2.2.3 Modeling Inductive Biases of the Discriminator in the Infinite-Width Limit

In practice, however, the inner optimization problem of Equation (S13) is not solved exactly. Instead, a proxy discriminator is computed using several steps of gradient ascent. For a learning rate ε, this results in the optimization procedure, from $i = 0$ to N:

$$\vartheta_{i+1} = \vartheta_i + \varepsilon \nabla_\vartheta \mathcal{L}_\alpha(f_{\vartheta_i}), \qquad f^\star_\alpha = f_{\vartheta_N} \tag{S15}$$

In the following, we show that by modeling the discriminator as the result of a gradient ascent in a set of parameterized neural networks, the problem is no longer unspecified. To facilitate theoretical analyses of discriminator training, we consider the continuous-time equivalent of Equation (S15):

$$\partial_t \vartheta_t = \nabla_\vartheta \mathcal{L}_\alpha(f_{\vartheta_t}), \tag{S16}$$

which we study in the infinite-width limit of the discriminator, making its analysis more tractable.

In the limit where the width of the hidden layers of f tends to infinity, Jacot et al. (2018c) showed that its so-called Neural Tangent Kernel (NTK) k_ϑ remains constant during a gradient ascent such as Equation (S16), i.e. there is a limiting kernel k^∞ such that:

$$\forall \tau \in \mathcal{R}_+,\ \forall x, y \in \mathcal{R}^n,\ \forall t \in [0, \tau],\ k_{\vartheta_t}(x, y) \triangleq \partial_\vartheta f_t(x)^T \partial_\vartheta f_t(y) = k^\infty(x, y). \tag{S17}$$

In particular, $k^\infty = k$ only depends on the architecture of f and the initialization distribution of its parameters. The constancy of the NTK of f_t during gradient descent holds for many standard architectures, typically without bottleneck and ending with a linear layer C. Liu et al. 2020, which is the case of most standard discriminators for GAN algorithms in the setting of Equation (S13). We discuss in details the applicability of this approximation in Appendix C.2.2.1.

The constancy of the NTK simplifies the dynamics of training in the functional space. In order to express these dynamics, we must first introduce some preliminary definitions and assumptions.

Definition S1 (Functional Gradient)**.** Whenever a functional $\mathcal{L}\colon L^2(\mu) \to \mathcal{R}$ has sufficient regularity, its gradient with respect to μ evaluated at $f \in L^2(\mu)$ is defined in the usual way as the element $\nabla^\mu \mathcal{L}(f) \in L^2(\mu)$ such that for all $\psi \in L^2(\mu)$:

$$\lim_{\epsilon \to 0} \frac{1}{\epsilon}(\mathcal{L}(f + \epsilon\psi) - \mathcal{L}(f)) = \langle \nabla^\mu \mathcal{L}(f), \psi \rangle_{L^2(\mu)}. \tag{S18}$$

Assumption 2. $k\colon \Omega^2 \to \mathcal{R}$ is a symmetric positive semi-definite kernel with $k \in L^2(\Omega^2)$.

Definition S2 (RHKS w.r.t. μ and kernel integral operator Sriperumbudur et al. 2010)**.** If k follows Assumption 2 and $\mu \in \mathcal{P}(\Omega)$ is a finite mixture of Diracs, we define the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_k^\mu$ of k with respect to μ given by the Moore–Aronszajn theorem as the linear span of functions $k(x, \cdot)$ for $x \in \operatorname{supp} \mu$. Its kernel integral operator from Mercer's theorem is defined as:

$$\mathcal{T}_{k,\mu}\colon L^2(\mu) \to \mathcal{H}_k^\mu,\ h \mapsto \int_x k(\cdot, x) h(x)\, \mathrm{d}\mu(x) \tag{S19}$$

Note that $\mathcal{T}_{k,\mu}$ generates $\mathcal{H}_k^\mu$, and elements of $\mathcal{H}_k^\mu$ are functions defined over all Ω as $\mathcal{H}_k^\mu \subseteq L^2(\Omega)$.

In this infinite-width limit, Jacot et al. (2018c) show that the discriminator $f_t \triangleq f_{\vartheta_t}$ trained by Equation (S20) obeys the following differential equation:

$$\partial_t f_t = \mathcal{T}_{k,\hat{\gamma}}\big(\nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f_t)\big). \tag{S20}$$

In the following, we will rely on this differential equation to gain a better understanding of the discriminator during training, and its implications for training the generator.

5.2.3 General Analysis of The Discriminator and its Gradients

In the previous section, we highlighted the indeterminacy issues arising in common analysis settings, showing the need for a theory beyond the loss function. From this, we proposed a novel analysis framework by considering the discriminator trained using gradient descent in the NTK regime. In this section, under mild assumptions on the discriminator loss function, we prove that Equation (S20) admits a unique solution for a given initial condition, thereby solving the indeterminacy issues. We then study differentiability of neural networks in this regime, a necessary condition for trainability of GANs. The results presented in this section are not specific to GANs but generalize to all neural networks trained under empirical losses of the form of Equation (S13), e.g. any pointwise loss such as binary classification and regression. All results, presented in the following in the context of a discrete distribution $\hat{\gamma}$ but that generalize to other distributions, are proved in Appendix C.2.1.

5.2.3.1 Existence, Uniqueness and Characterization of the Discriminator

The following is a positive result on the existence and uniqueness of the discriminator that also characterizes the general form of the discriminator, amenable to theoretical analysis.

Assumption 3. a and b from Equation (S13) are differentiable with Lipschitz derivatives over $\mathcal{R}$.

Theorem S3 (Solution of gradient descent)**.** *Under Assumptions 1 to 3, Equation* (S20) *with initial value* $f_0 \in L^2(\Omega)$ *admits a unique solution* $f_\cdot : \mathcal{R}_+ \to L^2(\Omega)$*. Moreover, the following holds:*

$$\forall t \in \mathcal{R}_+, f_t = f_0 + \int_0^t \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f_s)\right) \mathrm{d}s = f_0 + \mathcal{T}_{k,\hat{\gamma}}\left(\int_0^t \nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f_s)\, \mathrm{d}s\right). \tag{S21}$$

As for any given training time t, there exists a unique $f_t \in L^2(\Omega)$, defined over all of Ω and not only the training set, the aforementioned issue in Section 5.2.2.2 of determining the discriminator associated to $\hat{\gamma}$ is now resolved. It is now possible to study the discriminator in its general form thanks to Equation (S21). It involves two terms: the neural network at initialization f_0, as well as the kernel operator of an integral. This integral is a function that is undefined outside $\operatorname{supp} \hat{\gamma}$, as by definition $\nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f_s) \in L^2(\hat{\gamma})$. Fortunately, the kernel operator behaves

like a smoothing operator, as it not only defines the function on all of Ω but embeds it in a highly structured space.

Corollary 1. Under Assumptions 1 to 3, $f_t - f_0$ belongs to the RKHS $\mathcal{H}_k^{\hat{\gamma}}$ for all $t \in \mathcal{R}_+$.

In the GAN setting, this space is generated from the NTK k, which only depends on the discriminator architecture, and not on the considered loss function. This highlights the crucial role of the discriminator's implicit biases, and enables us to characterize its regularity for a given architecture.

5.2.3.2 Differentiability of the Discriminator and its NTK

We study in this section the smoothness, i.e. infinite differentiability, of the discriminator. It mostly relies on the differentiability of the kernel k, by Equation (S21). Therefore, we prove differentiability of NTKs of standard architectures, and then conclude about the differentiability of f_t.

Assumption 4 (Discriminator architecture). The discriminator is a standard architecture (fully connected, convolutional or residual) with activations that are smooth everywhere except on a closed set D of null Lebesgue measure.

This last assumption covers in particular the sigmoid and all ReLU-like activations.

Assumption 5 (Discriminator regularity). $0 \notin D$, or linear layers have non-null bias terms.

We first prove the differentiability of the NTK under these assumptions.

Proposition S10 (Differentiability of k)**.** *The NTK of an architecture following Assumption 4 is smooth everywhere over Ω^2 except on points of the form $(0, x)$ or $(x, 0)$. If Assumption 5 is also assumed, the NTK is then smooth everywhere.*

Remark. This result contradicts Bietti et al. (2019a) about the non-Lipschitzness of the bias-free ReLU kernel, that we prove to be incorrect. We further discuss this matter in Appendix C.2.2.3.

From Prop. S10, NTKs satisfy Assumption 2. We can thus use Theorem S3 and conclude about the differentiability of f_t.

Theorem S4 (Differentiability of f_t, informal)**.** *Suppose that k is an NTK of a network following Assumption 4. Then f_t has the same regularity as k over Ω.*

Remark. ReLU networks with two or more layers and no bias are not differentiable at 0. However, by introducing non-zero bias, the NTK and the infinite-width discriminator become differentiable everywhere. This observation explains some experimental results in Section 5.3.4.

This result demonstrates that, for a wide range of GAN formulations, e.g. vanilla GAN and LSGAN, the optimized discriminator indeed admits gradients almost everywhere, making the gradient flow given to the generator well-defined. This supports our motivation to bring the theory closer to empirical evidence indicating that many GAN models do work in practice where their theoretical interpretation until now has been stating the opposite Arjovsky et al. 2017a.

5.2.4 Fined-Grained Study for Specific Losses

Further assumptions on the loss function are needed to enhance our understanding of the discriminator in Equation (S21). Hence, we restrict our study to more specific cases. Proofs are detailed in Appendix C.2.1.

5.2.4.1 The IPM as an MMD with the NTK as its Kernel

We study the case of the IPM loss for the discriminator, with the following remarkable solutions.

Proposition S11 (IPM Discriminator)**.** *Under Assumptions 1 and 2, the solutions of Equation* (S20) *for* $a = b = id$ *are the functions of the form* $f_t = f_0 + t f^*_{\hat{\alpha}}$*, where* $f^*_{\hat{\alpha}}$ *is the unnormalized MMD witness function, yielding:*

$$f^*_{\hat{\alpha}} = \mathbb{E}_{x\sim\hat{\alpha}}[k(x,\cdot)] - \mathbb{E}_{y\sim\hat{\beta}}[k(y,\cdot)], \quad \mathcal{L}_{\hat{\alpha}}(f_t) = \mathcal{L}_{\hat{\alpha}}(f_0) + t \cdot \mathrm{MMD}^2_k\Big(\hat{\alpha}, \hat{\beta}\Big). \tag{S22}$$

Suppose that $f_0 = 0$; this is possible with the initialization scheme of Yaoyu Zhang et al. (2020). In the IPM case, the loss for the optimized discriminator is then proportional to the squared MMD distance Gretton et al. 2012 with the NTK as kernel between the empirical generated and target distributions.

This connection is especially interesting as the MMD has thoroughly been studied in the literature Muandet et al. 2017. If k is characteristic (a hypothesis discussed in Appendix C.2.2.5), then it defines a distance between distributions. Moreover, the statistical properties of the loss induced by the discriminator directly follow from those of the MMD: it is an unbiased estimator with a squared sample complexity that is independent of the dimension of the samples Gretton et al. 2007.

Remark (Link with Instance Smoothing). It is possible to show for IMPs that modeling the discriminator's architecture amounts to smoothing out the input distribution using the kernel integral operator $\mathcal{T}_{k,\hat{\gamma}}$ and

can thus be seen as a generalization of the regularization technique for GANs called instance noise Kaae Sønderby et al. 2017. This is discussed in further details in Appendix C.2.2.4.

Moreover, as $c = \mathrm{id}$, the spatial gradient of f_t received by the generator in Equation (S14) is proportional to $\nabla_x f^\star_{\hat{\alpha}}$. As following the gradient field induced by the discriminator has been shown to be a proxy to describe the evolution of the generated samples Mroueh et al. 2019, it is possible to analyze the evolution of the generated samples in this context through the gradient flow induced by the MMD (see Section 5.3.4). To this extent, results from Arbel et al. (2019) are directly transferable, including convergence guarantees and discretization properties. This is, to the best of our knowledge, the first work considering the use of NTKs as kernels for the MMD. A more in-depth study of this use case is out of the scope of this paper, but appears relevant considering this connection with GANs and its application to our empirical analysis in Section 5.3.4.

5.2.4.2 LSGAN, Convergence, and Emergence of New Divergences

Optimality of the discriminator can be proved when assuming that its loss function is well-behaved. In particular, when it is concave and bounded from above as it is the case e.g. for vanilla GAN and LSGAN, it is possible to study the convergence of the discriminator for large training times. Consider as an example the case of LSGAN, for which Equation (S20) can be solved by slightly adapting the results from Jacot et al. (2018c) in the context of regression.

Proposition S12 (LSGAN Discriminator)**.** *Under Assumptions 1 and 2, the solutions of Equation (S20) for $a = -(id + 1)^2$ and $b = -(id - 1)^2$ are the functions defined for all $t \in \mathcal{R}_+$ as:*

$$f_t = \exp(-4t\mathcal{T}_{k,\hat{\gamma}})(f_0 - \rho) + \rho, \qquad \rho = \frac{d\left(\hat{\beta} - \hat{\alpha}\right)}{d\left(\hat{\beta} + \hat{\alpha}\right)}. \tag{S23}$$

In the previous result, ρ is the optimum of $\mathcal{L}_{\hat{\alpha}}$ over $L^2(\hat{\gamma})$. When k is positive definite over $\hat{\gamma}$ (see Appendix C.2.2.5 for more details), f_t tends to the optimum for $\mathcal{L}_{\hat{\alpha}}$ as f_t tends to ρ over $\operatorname{supp}\hat{\gamma}$. Nonetheless, unlike the discriminator with arbitrary values of Section 5.2.2.2, f_∞ is defined over all Ω thanks to the integral operator $\mathcal{T}_{k,\hat{\gamma}}$. It is also the solution to the minimum norm interpolant problem in the RKHS Jacot et al. 2018c, therefore explaining why the discriminator tends to not

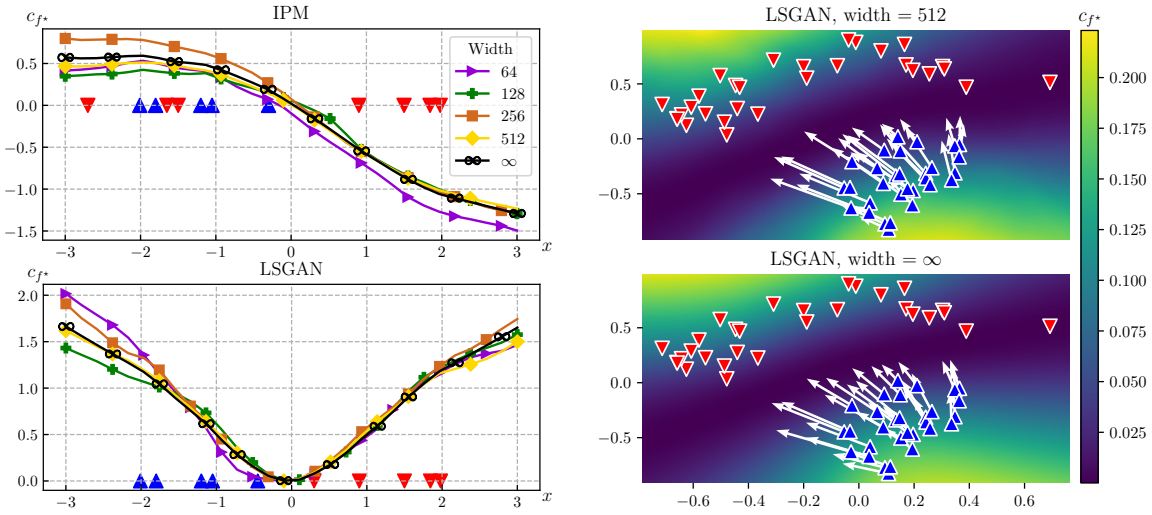


Figure S9. – Values of $c_{f^\star}$ for LSGAN and IPM, where $f^\star$ is a 3-layer ReLU MLP with bias and varying width trained on the dataset represented by ▼ (fake) and ▲ (real) markers, initialized at $f_0 = 0$. The infinite-width network is trained for a time $\tau = 1$ and the finite-width networks using 10 gradient descent steps with learning rate $\varepsilon = 0.1$, to make training times correspond. The gradients $\nabla_x c_{f^\star}$ are shown with white arrows on the two-dimensional plots for the fake distribution.

overfit in scarce data regimes (see Section 5.3.4), and consequently to have bounded gradients despite large training times, assuming its NTK is well-behaved. We also prove a more detailed generalization of this result for concave bounded losses in Appendix C.2.1.4, where the same optimality conclusion holds.

Following the discussion initiated in Section 5.2.2.2, and applying it to the case of LSGAN, similarly to the Jensen-Shannon, the resulting generator loss on discrete training data is constant. However, the previous result implies that the gradients received by the generator have no a priori reason to be null; see for instance the empirical analysis of Section 5.3.4. This observation raises the question of determining the actual loss minimized by the generator, which could be retrieved by analyzing the spatial gradient that it receives from the discriminator at each step. We were able to make the connection for the IPM loss in Section 5.2.4.1 thanks to Arbel et al. (2019) who leveraged the theory of Ambrosio et al. (2008), but this connection remains to be established for other adversarial losses. Furthermore, the same problem arises for gradients obtained from incompletely trained discriminators f_t. This is beyond the scope of this paper, but we believe that our work motivates such a study of actual divergences between distributions minimized by GANs.

5.2.5 Empirical Study

In this section, we present a selection of empirical results for different losses and architectures and evaluate the adequacy and practical implications of our theoretical framework in different settings; see Appendix C.2.3 for more results. All experiments were obtained with our Generative Adversarial Neural Tangent Kernel ToolKit GAN(TK)², available in the supplementary material and released upon publication in the hope that the community leverages and expands it for principled GAN analyses. It is based on the JAX Neural Tangents library Roman Novak et al. 2020, and is convenient to evaluate novel architectures and losses based on different visualizations and analyses.

Adequacy for fixed distributions. Firstly, we analyze the case where generated and target distributions are fixed. In this setting, we qualitatively study the similarity between the finite- and infinite-width regime of the discriminator and its gradients. Figure S9 shows $c_{f^\star}$ and its gradients on 1D and 2D data for LSGAN and IPM losses with a 3-layer ReLU MLP with varying widths. We find the behavior of finite-width discriminators to be close to their infinite-width counterpart for commonly used widths, and converges rapidly to the given limit as the width becomes larger.

In the following, we focus on the study of convergence of the generated distribution.

Experimental setting. We now consider a target distribution sampled from 8 Gaussians evenly distributed on a centered sphere (Figure S10), in a setup similar to that of Metz et al. (2017), Akash et al. (2017) and Arjovsky et al. (2017b). As for the generated distribution, instead of implementing a generator network that would complexify the analysis beyond the scope of this paper, we follow Mroueh et al. (2019) and Arbel et al. (2019), and model its evolution considering a finite number of samples – initially Gaussian – moving in a direction that minimizes the loss as fast as possible, i.e. along the flow induced by the vector field $-\nabla_x c_{f^\star_{\hat{\alpha}}}$. This setup is, in essence, similar to Equation (S14); we refer to Mroueh et al. (2019) for a more formal description. For both IPM and LSGAN losses, we evaluate the convergence of the generated distributions for a discriminator with ReLU activations in the finite and infinite width regime (see Figure S10 and Table S2). More precisely, we test a ReLU network with and without bias, and a "ReLU (reset)" network whose parameters are reinitialized at each generator

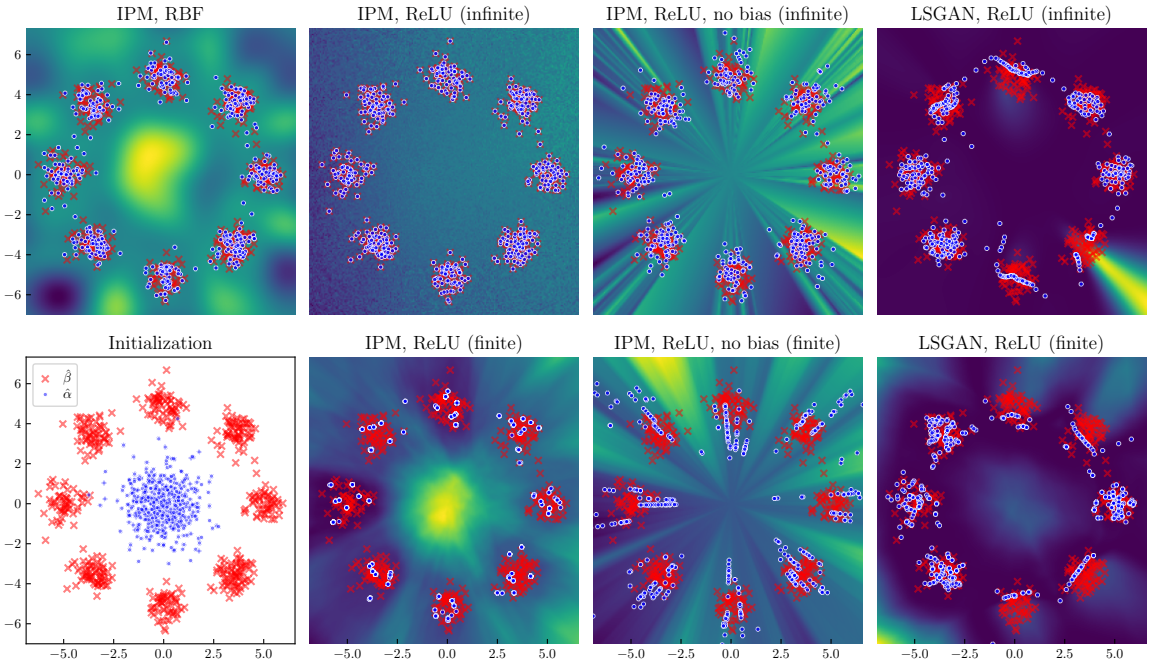


Figure S10. – Generator (●) and target (×) samples for different methods. In the background, $c_{f^\star}$.

step, corresponding to the setting of our infinite-width experiments. It is also possible to evaluate the advantages of the architecture in this setting by considering the case where the infinite-width loss is not given by an NTK, but by the popular Radial Basis Function (RBF) kernel, which is characteristic and yields attractive properties Muandet et al. 2017. Note that results for more datasets, including MNIST LeCun et al. 1998, and architectures are also available in Appendix C.2.3.

Impact of initialization. In the analysis conducted in the previous sections, we have shown in Equation (S21) that the discriminator depends on its initialization f_0 which can be non-zero and does not depend on the data. Thus, as it may bias the solution, we set $f_0 = 0$ in all the experiments, using the scheme proposed in Yaoyu Zhang et al. (2020). We have observed a global improvement in terms of speed of convergence of the samples.

Furthermore, we observe better results and more stable training when reinitializing the network at each step (see ReLU (reset)), which is closer to our theoretical framework. We believe that this is due to an inherent bias present in standard training of the discriminator as its parameters depend on old samples of the generator, and introduces a complex feedback loop. This surprising observation challenges the way we usu-

ally train discriminators and would constitute an interesting future investigation.

Adequacy. We observe that performances between the finite- and infinite-width regime are correlated, performances of ReLU Networks being considerably better in the infinite-width regime. Remarkably, in the IPM (inf.) setting, generated and target distributions perfectly match. This can be explained by the high capacity of infinite-width neural networks and their idealized setting; it has already been shown that NTKs benefit from low-data regimes Arora et al. 2020.

Impact of bias. The bias-free version of the discriminator performs worse than with bias, for both regimes and both losses. This is in line with findings for e.g. Basri et al. (2020), and can be explained in our theoretical framework by comparing their NTKs. Indeed, the NTK of a bias-free ReLU network is not characteristic, whereas its bias counterpart was proven to present powerful approximation properties Ji et al. 2020. Furthermore, results of Section 5.2.3.2 state that the ReLU NTK with bias is differentiable everywhere, whereas its bias-free version admits non-differentiability points, which can disrupt optimization based on its gradients: note in Figure S10 the abrupt streaks of the discriminator and its consequences on convergence.

NTK vs. RBF kernel. Finally, we observe the superiority of NTK w.r.t. to the RBF kernel. This highlights that the gradients of a ReLU network with bias are particularly well adapted to GANs. Visualizations of the gradients given by the ReLU architecture in the infinite-width limit are available in Appendix C.2.3 and further corroborate these findings. More generally, for the same reasons, we believe that the NTK of ReLU networks could be of particular interest for kernel methods requiring the computation of a spatial gradient, e.g. Stein Variational Gradient Descent Q. Liu et al. 2016.

5.2.6 Conclusion and Discussion

Contributions. Leveraging the theory of infinite-width neural networks, we proposed a framework of analysis of GANs explicitly modeling a large variety of discriminator architectures. We show that the proposed framework models more accurately GAN training compared to prior approaches by deriving properties of the trained discriminator.

Table S2. – Sinkhorn divergence Feydy et al. 2019, lower is better, similar to $\mathcal{W}_2$ averaged over three runs between the final generated distribution and the target dataset for the 8 Gaussians problem.

Loss	RBF kernel	ReLU	ReLU (no bias)	ReLU (reset)
IPM (inf.)	$(2.60 \pm 0.06) \times 10^{-2}$	$(9.40 \pm 2.71) \times 10^{-7}$	$(9.70 \pm 1.88) \times 10^{-2}$	—
IPM	—	$(1.21 \pm 0.14) \times 10^{-1}$	1.20 ± 0.60	$(1.97 \pm 0.31) \times 10^{-2}$
LSGAN (inf.)	$(4.21 \pm 0.10) \times 10^{-1}$	$(7.56 \pm 0.45) \times 10^{-2}$	$(1.27 \pm 0.01) \times 10^{1}$	—
LSGAN	—	3.07 ± 0.68	7.52 ± 0.01	2.14 ± 0.59

We demonstrate the analysis opportunities of the proposed modelization by further studying specific GAN losses and architectures, both theoretically and empirically, notably using our GAN analysis toolkit that we release publicly.

Limitations. Like all theoretical analyses, the proposed interpretation comes with its shortcomings, mostly tied to the NTK theory. In particular, this theory of infinite-width neural networks cannot in its current state model feature learning, which is an essential aspect of deep learning, although some progress have recently been made Geiger et al. 2020; G. Yang et al. 2020. Beyond NTK-related issues, our framework does not encompass gradient penalties in GANs, since we directly tackle the issues that led to their introduction; we leave these considerations for future work.

Perspectives and broader impact. We believe that this work and its accompanying analysis toolkit will serve as a basis for more elaborate analyses – e.g. extending results to a more general setting or taking into account the generator's architecture – thus leading to more principled, better GAN models.

As our work is mainly theoretical and does not deal with real-world data, it does not have direct broader negative impact on the society. However, the practical perspectives that it opens constitute an object of interrogation. Indeed, the developments of performant generative models can be the source of harmful manipulation Tolosana et al. 2020 and reproduction of existing biases in databases Jain et al. 2020, especially as GANs are still misunderstood. While such negative effects should be considered, attempt such as ours at explaining generative models might also lead to ways to mitigate potential harms.

5.3 Normalizing Kalman Filters for Multivariate Time Series Forecasting

abstract

This paper tackles the modelling of large, complex and multivariate time series panels in a probabilistic setting. To this extent, we present a novel approach reconciling classical state space models with deep learning methods. By augmenting state space models with normalizing flows, we mitigate imprecisions stemming from idealized assumptions in state space models. The resulting model is highly flexible while still retaining many of the attractive properties of state space models, e.g., uncertainty and observation errors are properly accounted for, inference is tractable, sampling is efficient and good generalization performance is observed, even in low data regimes. We demonstrate competitiveness against state-of-the-art deep learning methods on the tasks of forecasting real world data and handling varying levels of missing data.

The work in this section has led to the publication of a conference paper:
Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski (2020). "Normalizing Kalman Filters for Multivariate Time Series Analysis". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin. URL: https://proceedings.neurips.cc/paper/2020/hash/1f47cef5e38c952f94c5d61726027439-Abstract.html

5.3.1 Introduction

In most real world applications of time series analysis, e.g., risk management in finance, cannibalization of products in retail or anomaly detection in cloud computing environments, time series are not mutually independent and an accurate modelling approach must take these dependencies into account Salinas et al. 2019a. The classical approach Lütkepohl 2005 is to extend standard univariate models resulting in vector autoregression Silva et al. 2010, multivariate GARCH Bauwens et al. 2006 and multivariate state space models Hyndman et al. 2008; Durbin et al. 2012. Although these approaches yield useful theoret-

ical properties, they make idealized assumptions like Gaussianity, linear inter-dependencies, and are not scalable to even moderate number of time series Patton 2012 due to the number of parameters required to be estimated, which is restrictive for many modern applications involving large panels of time series. Recently, more expressive, scalable deep learning methods Salinas et al. 2019b; Rangapuram et al. 2018 were developed for forecasting applications that learn a joint global model for multiple time series; however, they still assume that these time series are mutually independent.

In this paper we propose the *Normalizing Kalman Filter* (*NKF*), a novel approach for modelling and forecasting complex multivariate time series by augmenting classical linear Gaussian state space models (*LGM*) with normalizing flows Rezende et al. 2015. The combined model allows us to leverage the flexibility of normalizing flows (NF), alleviating strong assumptions of traditional multivariate models, while still benefiting from the rich set of mathematical properties of *LGM*. In fact, we prove that despite modelling non-Gaussian data with nonlinear inter-dependencies, we can achieve exact inference since our model has closed-form expressions for filtering, smoothing and likelihood computation. We thus retain the main attractive properties of *LGM*, in contrast to related methods Fraccaro et al. 2017; Krishnan et al. 2017. Moreover, since our model is based on *LGM*, handling of missing data and integrating prior knowledge, e.g., seasonality and trend, becomes trivial. Therefore, the proposed model can be used in forecasting time series with missing or noisy data irrespective of whether the data regime is sparse (in terms of observed time points) or dense. More importantly, *LGM* directly gives us the ability to provide tractable multi-step ahead forecast distributions while accounting for all uncertainties; this is in contrast to recent deep learning-based autoregressive models Salinas et al. 2019b; Salinas et al. 2019a that do not incorporate accumulated prediction errors into forecast distributions since predictions of the model are used as lagged inputs in a multi-step forecast scenario. For the forecasting application, we show that our method scales linearly with the number of dimensions and number of time points, unlike most of the existing work that exhibits quadratic scaling with the number of dimensions. The necessary structural assumptions do not result in a loss of generality or expressiveness of the overall model.

In summary, our main contributions are as follows:

- A tractable method for modelling non-Gaussian multivariate time series data with nonlinear inter-dependencies that has Kalman-like recursive updates for filtering and smoothing.
- A scalable, robust multivariate forecasting method that handles missing data naturally and provides tractable multi-step ahead forecast distributions while

accounting for uncertainties unlike autoregressive models Salinas et al. 2019b; Salinas et al. 2019a.

- A thorough evaluation of applicability of normalizing flows in the context of high-dimensional time series forecasting for handling non-Gaussian multivariate data with nonlinear dependencies.

5.3.2 Normalizing Kalman Filters

Let $y_t \in \mathcal{R}^N$ denote the value of a multivariate time series at time t, with $y_{t,i} \in \mathcal{R}$ the value of the corresponding i-th univariate time series. Further, let $x_{t,i} \in \mathcal{R}^k$ be time varying covariate vectors associated to each univariate time series at time t, and $x_t := [x_{t,1}, \ldots, x_{t,N}] \in \mathcal{R}^{k \times N}$. Non-random and random variables are denoted by normal and bold letters, i.e., x and $\mathbf{x}$, respectively. We use the shorthand $y_{1:T}$ to denote the sequence $\{y_1, y_2, \ldots, y_T\}$.

5.3.2.1 Generative Model

The core assumption behind our Normalizing Kalman Filter (*NKF*) model is the existence of a latent state that evolves according to simple (linear) dynamics, with potentially complex and nonlinear dependencies between latent state and observations–and thus, among observations. More precisely, the dynamics of the latent state $\mathrm{l}_t \in \mathcal{R}^d$ are governed by a time-dependent *transition matrix* $F_t \in \mathcal{R}^{d \times d}$, up to additive Gaussian noise $\beta\epsilon_t$ as in (S24b). The state is then mapped into the space of observations with *emission matrix* $A_t \in \mathcal{R}^{d \times N}$, and additive Gaussian noise $\beta\varepsilon_t$ before being transformed by a potentially nonlinear function $f_t : \mathcal{R}^N \to \mathcal{R}^N$ parametrized by Λ, generating observation $\mathbf{y}_t \in \mathcal{R}^N$:

(*NKF* model)

$$\mathbf{l}_1 \sim \mathcal{N}(\mu_1, \Sigma_1) \qquad \text{(initial state)} \tag{S24a}$$

$$\mathbf{l}_t = F_t \mathbf{l}_{t-1} + \beta\epsilon_t, \quad \beta\epsilon_t \sim \mathcal{N}(0, \Sigma_t), \qquad \text{(transition dynamics)} \tag{S24b}$$

$$\mathbf{y}_t = f_t(A_t^T \mathbf{l}_t + \beta\varepsilon_t), \quad \beta\varepsilon_t \sim \mathcal{N}(0, \Gamma_t). \qquad \text{(observation model)} \tag{S24c}$$

With parameters Λ and $\Theta = (\mu_1, \Sigma_1, \{\Gamma_t, A_t\}_{t \geq 1}, \{\Sigma_t, F_t\}_{t \geq 2})$, the model is fully specified.[4] Note that the special case $f_t = id$ recovers the standard *LGM* where

4. Placing the noise before the non-linearity f_t in the observation model is important to obtain tractability for filtering and smoothing. However, this does not imply that data generated from a process where additive noise is added after the non-linear function cannot be modelled; in fact we use this particular model (nonlinear transformation with additive non-Gaussian noise) to generate

both the transition dynamics and the observation model are linear. In the following, this similarity with *LGM* will yield numerous computational benefits and it will further allow us to easily inject prior knowledge on the structural form of the dynamics (e.g., levels, trends, seasonalities Hyndman et al. 2008) for good generalization properties.

We consider a flexible nonlinear transformation for the observation model, assuming invertibility of f_t. This guarantees the conservation of probability mass and allows the evaluation of the associated density function at any given point of interest. In particular, the probability density of an observation y_t given the state l_t can be computed using the change of variables formula:

$$p(y_t|l_t;\Theta,\Lambda) = p_{\mathrm{z}}(f_t^{-1}(y_t)|l_t;\Theta)\left|\det\left[\mathrm{Jac}_{y_t}(f_t^{-1})\right]\right|, \tag{S25}$$

where the first term in $p_{\mathrm{z}}(z_t|l_t;\Theta)$ is the density of the Gaussian variable $\mathrm{z}_t := A_t l_t + \beta\varepsilon_t$ conditioned on l_t, and the second is the absolute value of the determinant of the Jacobian of f_t^{-1} given Λ, evaluated at y_t. This equation and Figure S11 illustrate the intuition behind our approach: we would like f_t^{-1} to transform the observations such that the dynamics become simple and the noise is Gaussian.

Computing the density (S25) raises several issues: (i) finding a flexible f_t while ensuring invertibility, (ii) being able to compute the inverse efficiently and (iii) tractability of the computation of the Jacobian term when the number of time series N is large. To this extent, we will take inspiration from *normalizing flows* Dinh et al. 2017; Kingma et al. 2018; Oliva et al. 2018, which are invertible neural networks that typically transform isotropic Gaussians to fit a more complex data distribution. These invertible networks are tailored to compute the Jacobian term efficiently. Moreover, they have proven to work very well for nonlinear high-dimensional data, e.g., images Kingma et al. 2018, both in terms of flexibility and generalization. In our approach, we apply these invertible neural networks to temporal data, using them to map the distribution p_{z} given by the *LGM* to the complex data distribution. This yields a powerful function f_t where the Jacobian term is computable in *linear time* in N.

Inference and Learning. With the presented generative model it is possible to do inference (e.g., filtering and smoothing) and training in a simple and tractable way. Similar to the *LGM*, computing the *filtered distribution* $p(l_t|y_{1:t};\Theta,\Lambda)$ is essential as it determines our current belief on the state having observed all the data up to time t, and it takes part in the computation of the likelihood of the model parameters as well as the forecast distribution. For general nonlinear state space models its computation is tedious as it involves integrating out previous

data and test our method in the qualitative experiments in Section 4.1 (refer to appendix C.1 in the supplementary material for details).

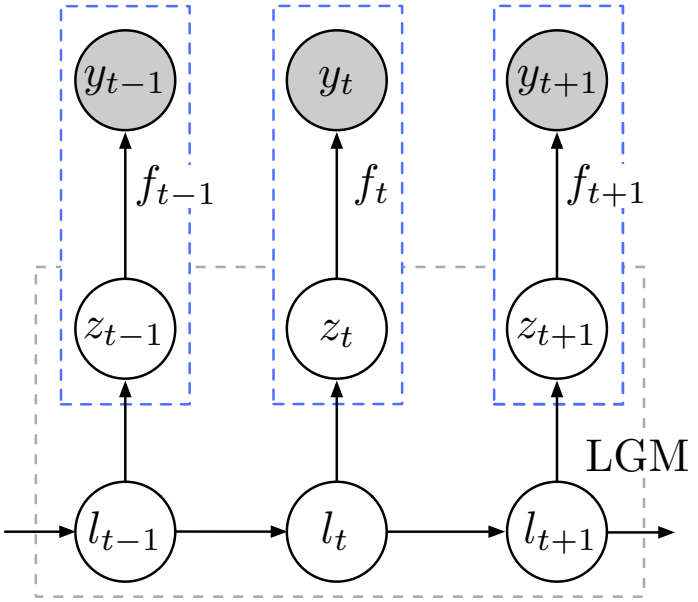


Figure S11. – Generative model of the *NKF*. States l_t and *pseudo-observations* z_t are produced from an *LGM*, which are then transformed through a normalizing flow f_t, producing observations y_t.

states. Methods such as Particle Filters H. Liu et al. 2009 resort to Monte Carlo approximations of these integrals but have difficulty scaling to high dimensions. Other methods circumvent this by locally linearizing the nonlinear transformation Zarchan et al. 2015 or by using a finite sample approximation Julier et al. 2004 in order to apply–in both cases– the techniques of the standard *LGM*, but introduce a bias. In contrast, our model allows for computing this quantity in a tractable and efficient manner, without resorting to any form of simplification or approximation. In fact, despite the nonlinear nature of f_t, the filtered distribution *remains* Gaussian and its parameters can be computed in closed form similarly to the Kalman Filter, as shown in the following proposition.

Proposition S13 (Filtering). *The filtered distributions of the NKF model are Gaussian and are given by the filtered distributions of the corresponding LGM with pseudo-observations* $z_t := f_t^{-1}(y_t),\ t \geq 1$. *That is,* $p(l_t|y_{1:t}; \Theta, \Lambda) = p_{LGM}\left(l_t|z_{1:t}; \Theta\right)$ *where* p_{LGM} *refers to the distribution given by the LGM.*

This can be proved by induction using recursive Bayesian estimation and is available in Appendix C.3.1.1 along with the exact updates. Proposition S13 shows that filtered distributions for our model are available in closed-form and have the

same computational complexity as that of the *LGM*, plus the complexity of the inverse of the nonlinear function f and the Jacobian term in (S25). [5]

Our nonlinear model (S24) is also amenable to smoothing, i.e., computing the smoothed posterior distribution $p(l_t|y_{1:T};\Theta,\Lambda)$, given past, present and future observations. Smoothing is a prevalent problem in many fields, with applications such as estimating the distribution of missing data and providing explanations in the context of offline anomaly detection. Smoothing can be obtained with a backward iterative approach using the quantities computed during a preliminary filtering pass, starting from the filtered distribution corresponding to step T, $p(l_T|y_{1:T};\Theta)$. Similar to filtering, smoothing updates also directly translate to the corresponding updates of the *LGM*.

Proposition S14 (Smoothing)**.** *The smoothed distributions of the NKF model are Gaussian and are given by the smoothed distributions of the corresponding LGM with pseudo-observations* $z_t := f_t^{-1}(y_t), t = 1, 2 \ldots, T$*. That is,* $p(l_t|y_{1:T};\Theta,\Lambda) = p_{LGM}\left(l_t|z_{1:T};\Theta\right)$.

The tractability of the computation of the filtered distribution implies that the likelihood of the model parameters can be computed efficiently by integrating out the latent state. In case of the standard *LGM*, the likelihood of its parameters Θ given the observations $z_{1:T}$ can be written as

$$\ell(\Theta) = \prod_{t=1}^{T} p_{LGM}(z_t|z_{1:t-1};\Theta), \tag{S26}$$

where $p_{LGM}(z_t|z_{1:t-1};\Theta)$ is the predictive distribution of *LGM*. This distribution is available in closed-form thanks to the filtering updates Barber 2011. We now show that the likelihood of our nonlinear model given the observations $\{y_{1:T}\}$ is essentially a reweighted version of this expression with $z_t = f_t^{-1}(y_t)$.

Proposition S15 (Likelihood)**.** *The likelihood of the parameters* (Θ,Λ) *of the NKF model given the observations* $\{y_{1:T}\}$ *can be computed as*

$$\ell(\Theta,\Lambda) = p(y_{1:T};\Theta,\Lambda) = \prod_{t=1}^{T} p_{LGM}(z_t|z_{1:t-1};\Theta)\left|\det\left[Jac_{z_t}(f_t)\right]\right|^{-1}, \tag{S27}$$

where $z_t = f_t^{-1}(y_t)$ *and* $p_{LGM}(z_t|z_{1:t-1};\Theta)$ *denotes the predictive distribution of LGM.*

Hence, we can compute the likelihood of the parameters of the *NKF* exactly (Appendix C.3.1.3 contains the closed-form expressions), and fit our model to the given data by directly maximizing the likelihood. This is done without resorting

5. In Sec. 5.3.3, the complexity of the inverse is the same as the forward map, and the Jacobian term is linear in N.

to any approximations typically required in other nonlinear extensions of *LGM* such as the Particle Filter and the Extended/Unscented Kalman Filter.

5.3.2.2 Parameter Estimation using RNNs

In Sec. 5.3.2.1, we assumed the data was generated from a given family of *NKF* state space models characterized by its parameters $\Theta_{1:T} = (\mu_1, \Sigma_1, \{\Gamma_t, A_t\}_{t=1}^T, \{\Sigma_t, F_t\}_{t=2}^T)$ along with Λ, the parameters of the normalizing flow transformation f_t (which we assume to be constant over time). However the exact values of the parameters are unknown. Similar to Rangapuram et al. 2018; Salinas et al. 2019b, we propose to predict the parameters Θ from the covariates, using a recurrent neural network (RNN) where the recurrent function Ψ is parametrized by Φ, taking into account the possibly nonlinear relationship between covariates x_t:

$$\Theta_t = \sigma(h_t; \Phi), \quad h_t = \Psi(x_t, h_{t-1}; \Phi), \quad t = 1, \dots T, \tag{S28}$$

where σ denotes the transformation mapping of the RNN output to domains of the parameters (Appendix C.3.2.1). The RNN allows our model to be more expressive by allowing time-varying parameters along with temporal dependencies in the covariates, a requirement for forecasting. The parameters of the RNN Φ and of the normalizing flow Λ are estimated by maximizing the conditional likelihood,

$$p(y_{1:T}|x_{1:T}; \Phi, \Lambda) := p(y_{1:T}; \Theta_{1:T}, \Lambda) \tag{S29}$$

where $p(y_{1:T}; \Theta_{1:T}, \Lambda)$ is given by (S27). Although the transition model in our case is linear, complex dynamics can still be taken into account as the parameters of the *LGM*, which are now the outputs of a RNN, may change in a nonlinear way.

5.3.2.3 Applications: Forecasting and Missing Values

Given a model for sequential data, we can apply it to obtain future time steps $T+1 : T+\tau$, or in other words, *forecasts* of the time series, starting from past observations $y_{1:T}$ and covariates $x_{1:T+\tau}$.

In our case such a forecast distribution $p(y_{T+1:T+\tau}|y_{1:T}, x_{1:T+\tau}; \Phi, \Lambda)$ can also be computed efficiently and is available in closed-form. From this, not only can we readily evaluate possible future scenarios (see Appendix C.3.1.4), but also draw samples from it to generate forecasts.

This is achieved by first computing the filtered distribution for the input range $1:T$, and then recursively apply the transition equation and the observation model to generate prediction samples. More precisely, starting with the RNN state h_T

and l_T sampled from $p(l_T|y_{1:T}, x_{1:T}; \Theta_{1:T})$, given by (S28) and (S29), respectively, we iteratively apply:

$$F_t, A_t, \Sigma_t, \Gamma_t = \sigma(h_t; \Phi), \qquad h_t = \Psi(x_t, h_{t-1}; \Phi), \tag{S30a}$$

$$l_t = F_t l_{t-1} + \epsilon_t, \qquad \epsilon_t \text{ sampled from } \mathcal{N}(0, \Sigma_t), \tag{S30b}$$

$$y_t = f_t(A_t^T l_t + \varepsilon_t), \qquad \varepsilon_t \text{ sampled from } \mathcal{N}(0, \Gamma_t), \quad t = T+1, \ldots, T+\tau. \tag{S30c}$$

In contrast to alternative deep learning approaches Salinas et al. 2019a; Salinas et al. 2019b; Fraccaro et al. 2017, this generative procedure is *not* autoregressive in the sense that observations y_t are never fed to the model. Instead, it is the filtering that correctly updates our beliefs based on the observed data. This has several notable consequences: we do not have to resort to large and cumbersome beam searches to obtain proper uncertainty estimates, noisy observations with varying levels of uncertainty can be properly handled, and long-term forecast computations are accelerated as intermediate observations need not be computed.

In many real world applications, observations for each time step may not be available, e.g., out-of-stock situations in the context of demand time series (no sales does not mean no demand if out-of-stock) or network failures in the case of sensor data streams. *Handling missing data* is then of central importance and there are two aspects to it: (i) learning in the presence of missing values without imputing them and (ii) imputing the missing values. Similar to *LGM* Shumway et al. 1982, our approach offers a straightforward, tractable and unbiased way for handling missing data in both of these scenarios.

In case of learning with missing values the likelihood terms corresponding to the missing entries should be ignored. This amounts to effectively dealing with them in the filtering step. Without loss of generality, let us assume that targets until time $t-1$ are observed but are missing for t. In this case, we can compute the filtered distribution $p(l_{t-1}|y_{1:t-1}; \Theta)$ with the method outlined in Sec. 5.3.2. As y_t is not observed, the filtered distribution at time t then simply corresponds to $p(l_t|y_{1:t-1}; \Theta)$, and can be obtained by applying the prediction step, starting from the filtered distribution at time $t-1$.

For the second case, we wish to impute missing values at time $t \notin \mathtt{t}^{\text{obs}}$ based on observed values at times $\mathtt{t}^{\text{obs}}$. This can be easily achieved by computing the smoothed distribution $p(l_t|y_{\mathtt{t}^{\text{obs}}}; \Theta)$ in the presence of missing data, and from this compute $p(y_t|y_{\mathtt{t}^{\text{obs}}}; \Theta)$ (full details in Appendix C.3.1.5). This distribution is available in closed form and can be readily sampled from. This allows us to use the same model for forecasting and imputation without the need for bidirectional RNNs Cao et al. 2018 as the smoothing procedure inherently handles future data.

5.3.3 Local-Global Instantiation

The number of parameters of the *NKF* scales quadratically in the dimensionality d of the state (stemming from the covariance matrices of the Gaussian distributions) in the worst case. Leveraging the strength of NF, we present an instantiation of *NKF* with an induced *local-global* structure, that displays several advantages over the general unstructured form, e.g., exhibiting linear scaling in d.

We assume that each time series is associated with latent factors that evolve *independently w.r.t.* the factors of the other time series. These factors will in turn be *mixed* together with the normalizing flow, producing *dependent* time series observations. Formally, for each univariate time series i we associate a *local LGM* with parameters $\Theta^{(i)} = (\mu_1^{(i)}, \Sigma_1^{(i)}, \{\Gamma_t^{(i)}, A_t^{(i)}\}_{t\geq 1}, \{\Sigma_t^{(i)}, F_t^{(i)}\}_{t\geq 2})$, whose dynamics are characterized by:

$$\mathbf{l}_t^{(i)} = F_t^{(i)}\mathbf{l}_{t-1}^{(i)} + \beta\epsilon_t^{(i)}, \quad \beta\epsilon_t^{(i)} \sim \mathcal{N}(0, \Sigma_t^{(i)}), \tag{S31a}$$

$$\mathbf{z}_t^{(i)} = A_t^{(i)T}\mathbf{l}_t^{(i)} + \beta\varepsilon_t^{(i)}, \quad \beta\varepsilon_t^{(i)} \sim \mathcal{N}(0, \Gamma_t^{(i)}), \tag{S31b}$$

$$\mathbf{y}_t = f_t(\mathbf{z}_t^{(1)}, \ldots, \mathbf{z}_t^{(N)}). \tag{S31c}$$

Note that here $\Gamma_t^{(i)}$ is a scalar and denotes variance of the Gaussian noise. This local-global structure has several advantages: (i) the dynamics of the local states need not be the same for each time series and thus, prior knowledge on the evolution can be readily injected *per* time series, (ii) computations can be done in parallel for each time series and finally (iii) we may benefit from the effects of amortization by predicting local parameters for each time series and sharing the same weights Φ: $\Theta_t^{(i)} = \Psi(x_t^{(i)}, h_{t-1}^{(i)}; \Phi)$, allowing the RNN to make analogies between time series. In this case, dependencies across time series are captured with the normalizing flow f_t, mixing the components together in a nonlinear fashion.

We now explain a possible instantation of the *LGM* (Eq. (S31a), (S31b)), which is an innovation-based model, similar to Rangapuram et al. 2018. Note that this is a choice and not a requirement as any other *LGM* instance may be used if that better reflects the data at hand. Nonetheless, with this form a wide range of phenomena can be captured, e.g., long term direction (trends), patterns that repeat (seasonality), cycles with different periodicity, multiplicative errors, etc.; we refer the reader to Hyndman et al. 2008. In our instantiation, we combine both level-trend and seasonality components, as described in Appendix C.3.2.2.

For f_t, we use the RealNVP architecture Dinh et al. 2017: it is a network that is composed of a series of parametrized invertible transformations with a lower triangular Jacobian structure and vector component permutations in order to capture complex dependencies. This structure has the advantage of being flexible,

while maintaining a tractable Jacobian term, computable in linear time. Moreover, as opposed to more recent architectures e.g., Kingma et al. 2018, the number of parameters scales linearly. For the RNN, we use an LSTM with 2 layers.

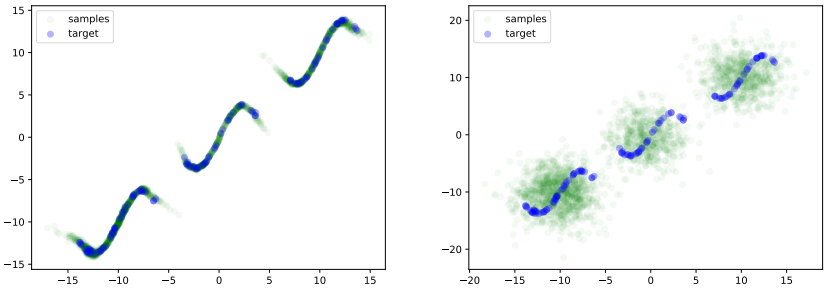
In terms of parameter complexity, this particular instantiation scales as $O(N)$ for each timestep, due to the diagonal structure of the covariance matrices (although they do not correspond to the effective number of parameters as these are predicted from the RNN). Moreover, time complexity for likelihood computation and forecasting scales as $O(N(d^2+k))$ for each timestep, where k is the number of covariates and d is the dimension of latent state; in experiments $d = 32$ (24 hourly components + 7 daily components + one level component) for hourly data and $d = 8$ for daily data.

Partial observations: The local-global instantiation model presented above could deal with partial observations (i.e., only some entries of $y_t \in \mathcal{R}^N$ are missing but not all); however it would require marginalisation over the missing dimensions, which cannot be done in closed-form. Alternatively, if one is interested in dealing with partial observations, it is possible to consider an instantiation with a global (i.e., multivariate) *LGM* with non-diagonal covariance matrix modelling *linear* dependencies among the time-series $y_t \in \mathcal{R}^N$ at any given time step t, and a normalising flow applied locally to each time-series: $y_t = f_t(z_t) = (u_t(z_{1,t}), \ldots, u_t(z_{N,t}))$, with $u_t : \mathcal{R} \to \mathcal{R}$ (see ablation study in our experiments). In this case, the marginalisation of any missing set of time series actually yields an analytic form for the filtering, smoothing and forecast distribution Shumway et al. 1982, and hence the handling of partial observations can be efficiently dealt with.

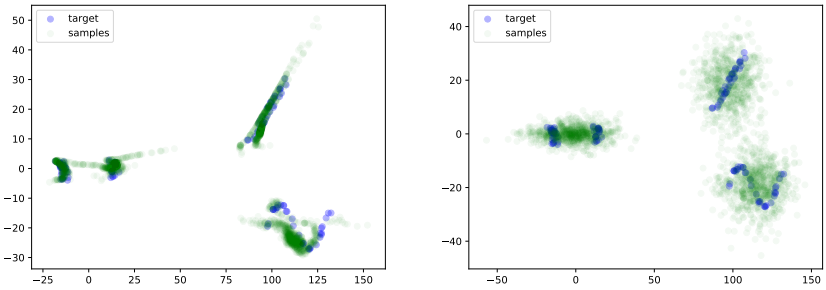
5.3.4 Experiments

5.3.4.1 Qualitative Results

We qualitatively assess our approach in Section 5.3.3 with two synthetic datasets `S1` and `S2` with increasing difficultly. The datasets are composed of 2 daily univariate time series with a weekly seasonality pattern. We compare our approach against a variant without any normalizing flow (this amounts to setting $f_t = id$ in (S31)). For dataset `S1` the daily data has three different modes and highly non-Gaussian time-*independent* observational noise. `S2` is similar, but the time series are *mixed* together in a nonlinear fashion, and the observational noise is not only non-Gaussian, but time-*dependent* (Appendix C.3.3.1 contains a detailed description). Target data (blue) and forecasts (green) are plotted in Fig. S12, where the first and second axis correspond to the first and second time series respectively, and data is aggregated over the temporal axis. Observations and forecasts can also



(a) S1 : Results with (left) and without (right) NF.



(b) S2 : Results with (left) and without (right) NF.

Figure S12. – Evaluation w/ and w/o NF. The axes correspond to the two components of the 2D time series; the temporal axis is marginalized out. Green points correspond to model samples when predicting 24 days ahead and blue points to future samples from the data-generating process.

be seen in Appendix C.3.3.1 from a viewpoint that better highlights the seasonal nature of the observations.

For both datasets, the variant with $f_t = id$ captures the daily modes correctly, but assumptions such as Gaussianity and independence between time series introduce errors. In contrast, visual inspection reveals that applying the normalizing flow allows us to fit the data better, capturing the complex dependencies between time series and the non-Gaussian noise, for both S1 and S2.

5.3.4.2 Quantitative Results

We follow the experimental set up proposed in Salinas et al. 2019a since it focusses on the same problem in the forecasting application. In particular, we evaluate on the public datasets used in Salinas et al. 2019a; see C.3.3.2 for details

and Table S4 for the summary of datasets. Similar to Salinas et al. 2019a, the forecasts of different methods are evaluated by splitting each dataset in the following fashion: all data prior to a fixed *forecast start date* compose the training set and the remainder is used as the test set. We measure the accuracy of the forecasts of various methods on all the time points of the test set. The hyperparameters have been selected using a validation set of equal size to the test set, created from the last time steps of the training data.

Our evaluation is extensive, covering relevant classical multivariate approaches as well as recent deep learning based models. In particular, we compare against *VES*, a direct generalization of univariate innovation state space model (a special case of *LGM* used in forecasting) to multivariate time series (see Chapter 17 of Hyndman et al. 2008), *VAR*, a multivariate linear autoregressive model and *GARCH* a multivariate conditional heteroskedastic model Van der Weide 2002; we include result of Lasso-regularized *VAR* as well. We also compare against the recent deep-learning based approaches *GP-Copula* Salinas et al. 2019a and *KVAE* Krishnan et al. 2017 specifically developed for handling non-Gaussian multivariate data with non-linear dependencies. *GP-Copula* builds on ideas of *VAR* and relies on RNN and low-rank Gaussian Copula process for going beyond Gaussianity and linearity. In contrast, *KVAE* uses a variational autoencoder on top of linear state space models to achieve the same. Unlike our *NKF* model, inference and likelihood computation are not tractable in *KVAE* and it relies on particle filters for their approximation. Additionally, we compare with *DeepAR* Salinas et al. 2019b an autoregressive recurrent neural network based method for univariate time series forecasting. *DeepState* Rangapuram et al. 2018 is a special case of *NKF* model and is part of the ablation study. See Table S3 for summary of the compared methods based on various parameters.

In order to evaluate forecasting models, *continuous ranked probability score* (CRPS) is generally accepted as one of the most well-founded metricsMatheson et al. 1976; Gneiting et al. 2007. However, this metric is only defined for univariate timeseries and cannot assess if dependencies across time series are accurately captured. Different generalizations to the multivariate case have been used, e.g., the energy score, or CRPS-Sum Salinas et al. 2019a. We have opted for the CRPS-Sum, as the energy score suffers from the curse of dimensionality, from both a statistical and computational viewpoint Pinson et al. 2013; Ramdas et al. 2015. The introduction of the CRPS-Sum Salinas et al. 2019a was experimentally justified. In this work, we prove it is theoretically sound as justified formally in Appendix C.3.3.4 following the proper scoring rule framework Gneiting et al. 2007. Note that, opposed to Salinas et al. 2019a, as different time series observations may have drastically different scales, we first normalize each time series by the sum of its absolute values before computing this metric (hence the '-N' suffix). We also did not

Approach	*VES*	*VAR*	*GARCH*	*DeepAR*	*GP-Copula*	*KVAE*	*NKF*(Ours)
Multivariate	✓	✓	✓	×	✓	✓	✓
Non-Linear, Non-Gaussian	×	×	×	✓	✓	✓	✓
Filtering & Smoothing	✓	NA	NA	NA	NA	✓	✓
Tractable Multi-step Forecast	✓	×	×	×	×	×	✓
Tractable Data Imputation	✓	×	×	×	×	×	✓

Table S3. – Comparative summary of competing approaches on various parameters.

	method	`exchange`	`solar`	`elec`	`wiki`	`traffic`
	VES	**0.005 ± 0.000**	0.9 ± 0.003	0.88 ± 0.0035	-	0.35 ± 0.0023
	VAR	**0.005 ± 0.000**	0.83 ± 0.006	0.039 ± 0.0005	-	0.29 ± 0.005
	VAR-Lasso	0.012 ± 0.0002	0.51 ± 0.006	0.025 ± 0.0002	3.1 ± 0.004	0.15 ± 0.002
	GARCH	0.023 ± 0.000	0.88 ± 0.002	0.19 ± 0.001	-	0.37 ± 0.0016
	DeepAR	0.006±0.001	0.336±0.014	0.023±0.001	0.127±0.042	0.055±0.003
	GP-Copula	0.007±0.000	0.363±0.002	0.024±0.000	0.092±0.012	**0.051±0.000**
	KVAE	0.014 ± 0.002	0.34 ± 0.025	0.051 ± 0.019	0.095 ± 0.012	0.1 ± 0.005
	NKF(Ours)	**0.005 ± 0.000**	**0.320±0.020**	**0.016±0.001**	**0.071±0.002**	0.10±0.002
ablation study	$f_t = id$	0.005±0.000	0.415±0.002	0.026±0.000	0.082±0.000	0.123±0.000
	f_t *Local*	0.005±0.000	0.405±0.005	0.018±0.001	0.068±0.004	0.102±0.013

Table S4. – CRPS-Sum-N (lower is better), averaged over 3 runs. The case $f_t = id$ is *DeepState* Rangapuram et al. 2018 and *VES* can be seen as part of ablation where normalizing flow and RNN are removed from *NKF*.

choose log-likelihood since not all methods yield analytical forecast distributions and is not meaningful for some methods Salinas et al. 2019a.

We report CRPS-Sum-N metric values achieved by all methods in Table S4. Classical methods, because of the Gaussianity and linear dependency assumptions, typically yield inferior results; entries marked with '-' are runs failed with numerical issues. Deep learning based models have superior performance overall. In particular, *NKF* achieves the best result in 4 out of 5 datasets. On `traffic` *NKF* is better than all methods except for *DeepAR* and *GP-Copula* which are purely data-driven autoregressive approaches with minimal modelling assumptions. Given the domain of `traffic` dataset is $(0, 1)$, it would be interesting to verify in future work if the relatively weak performance is due to a short-coming of the normalizing flow part or due to the modelling choice of adopting a state space model instead of an autoregressive process.

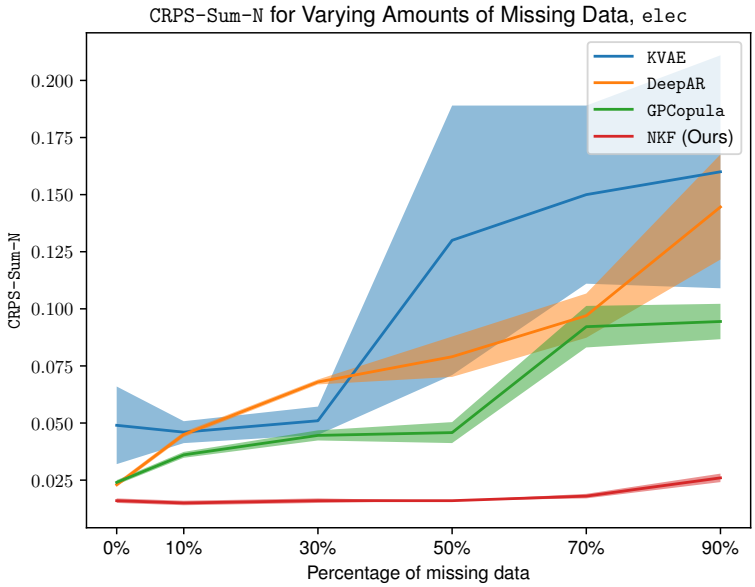


Figure S13. – Forecasting with missing data.

Ablation Study First note that *VES*, which models only Gaussian data with linear dependencies, can be seen as an ablation where RNN and normalizing flow are not used. Next, to evaluate the usefulness of the normalizing flow in accurately modelling real world data, we analyze the performance of two particular instances of *NKF* : (i) '$f_t = id$': the normalizing flow is set to be the identity function, as in Section 5.3.4.1, therefore also reducing to the *DeepState* model proposed in Rangapuram et al. 2018 (ii) 'f_t *Local*': the normalizing flow is applied *locally* for each time series, modelling non-Gaussianity and non-linearity, but not any dependencies between time series, as explained in Appendix C.3.3.2. This ablation is important in order to analyze the advantages of capturing the potential dependencies across time series in the data. Overall, we observe (bottom lines in Table S4) a significant increase in performance from the identity function to a local NF, along with another increase when applying the global NF (see *NKF* results), apart from the `wiki` dataset. While we do not have a satisfying explanation, we speculate that this is due to modelling errors, the optimization algorithm, or simply because the time series may not exhibit much dependencies between each other.

Missing Data Experiment We evaluate our model in the context of varying degrees of missing data during training and evaluation. From `elec`, we remove $p = 10+20k, k = 0\dots,4$, percent of the training data at random. This dataset, while realistic, is highly regular with clear seasonality patterns. The models are then

evaluated on rolling windows, where the input range observations are missing with the same probability. In Fig. S13 and Appendix C.3.3.3, we report results for our approach, along with *DeepAR*, *GP-Copula*, *KVAE*. We observe that not only does *NKF* outperform other approaches by a large margin, but its error also increases slower than in other methods when the percentage of missing data is increased. We believe that this is due to the proper handling of the uncertainties in our approach since our model does not directly take observations as input. Moreover, the strong results obtained for up to 90% of missing data demonstrate that our method encodes useful prior knowledge due to the structure induced in the *LGM*, rendering this method useful even in low data regimes (the same observation is made in Rangapuram et al. 2018 for this dataset).

5.3.5 Related Work

Neural networks for forecasting have seen growing attention in recent years Salinas et al. 2019b; Smyl 2020; Yuyang Wang et al. 2019; Gasthaus et al. 2019; Laptev et al. 2017; Lai et al. 2018; Oreshkin et al. 2019. We refer to Benidis et al. 2020 for an introductory overview. Most work concerns the univariate case using global models, assuming that time series are independent given the covariates and the model parameters. The family of global/local models, e.g., Sen et al. 2019; Yuyang Wang et al. 2019, provide a more explicit way of incorporating global effects into univariate time series, without attempting to estimate the covariance structure of the data. An explicit probabilistic multivariate forecasting model is proposed in Salinas et al. 2019a, which relies on Gaussian copulas to model non-Gaussian multivariate data.

Further related work combines probabilistic time series models with neural networks (e.g., point/renewal processes and neural networks A. C. Turkmen et al. 2019 or exponential smoothing based expressions Smyl 2020). We extend the approach in Rangapuram et al. 2018 which uses an RNN to parametrize a state space model to the multivariate case alleviating Gaussianity and linear dependency assumptions in the observation model.

The idea to take advantage of the appealing properties of Kalman Filters (KF) Kalman 1960 while relaxing its assumptions is not new. Prominent examples include the Extended Kalman Filter (EKF) Zarchan et al. 2015, the Unscented Kalman Filter (UKF) Julier et al. 2004 and Particle Filters (PF) J. S. Liu et al. 1998 that relax the linearity and Gaussianity assumption by approximation or sampling techniques. The Gaussian process state space model (GPSSM) Ko et al. 2011; Deisenroth et al. 2012 is a nonlinear dynamical system that extends *LGM*s by using GPs as the transition and/or observation mappings but typically assume the noise is additive Gaussian. If the noise is non-Gaussian, then these models

again have to resort to approximation techniques similar to Particle Filters. Kernel Kalman Filters Gebhardt et al. 2017 address the linearity limitation of *LGM*s by defining the state space model in the reproducing Kernel Hilbert space (RKHS). In particular, the random latent state and the observation variable are mapped to RKHS and the state dynamics and the observation model are assumed to be linear in the kernel space. Note, however, that this approach still relies on the assumption that the noise is additive Gaussian.

Similarly, combining KF with neural networks is not new. Additionally to Rangapuram et al. 2018, Fraccaro et al. 2017 proposes to combine KF with Variational Auto-Encoders (KVAE) and Krishnan et al. 2017 proposes variational approximations of the predictive distribution in nonlinear state space models. Finally, while most work on normalizing flows Dinh et al. 2017; Kingma et al. 2018; Behrmann et al. 2019; Oliva et al. 2018 was presented in the i.i.d. setting, extensions to sequential data have recently been proposed J. He et al. 2018; Xuezhe Ma et al. 2019; Ziegler et al. 2019. Concurrent independent work by Rasul et al. 2020 also addresses the multivariate time series forecasting problem by combining normalizing flows with (deep) autoregressive models instead of state space models as in our work.

5.3.6 Discussion and Conclusion

In this paper we presented a simple, tractable and scalable approach to high-dimensional multivariate time series analysis, combining classical state space models with normalizing flows. Our approach can capture non-linear dependencies in the data and non-Gaussian noise, while still inheriting important analytic properties of the linear Gaussian state space model. This model is flexible, while still retaining interesting prior structural information, paramount to good generalization in low data regimes. Experimentally, our approach achieves the best results among a wide panel of competing methods on the tasks of forecasting and missing value handling. One caveat of our approach is that we no longer have identifiability w.r.t. the state space parameters: an interesting avenue of research is to work towards identifiability, e.g. by constraining the normalizing flow's expressivity.

Chapter 6

CONCLUSION

To conclude, we give a brief recall of the contributions of this thesis, and provide directions for further research.

6.1 Learning Dynamical Systems using Deep Learning

In the recent years, deep learning has been applied to more and more new fields and is opening up new research directions and avenues to treat classical problems from a data-driven perspective. In a number of fields, e.g. machine translation, image and speech recognition, it has drastically changed the way previously complex problems where tackled, going from knowledge based approaches to more prior agnostic approaches where massive amounts of data are being leveraged. For all the aforementioned problems, prior knowledge is scarce and/or is very difficult to transcribe formally. This is not necessarily the case for physical processes, which have been studied since centuries and for which we have acquired a vast amount of knowledge along with a plethora of tools and methods which have proven their value and usefulness. In this context, the role that machine learning– and more specifically deep learning– has to play is not quite as clear; addressing this open question has been one of the aims of this thesis.

In Section 3.1, we have studied how the learning problem could be formulated, in particular in the realistic setting of partial observability, adopting a continuous time framework taking inspiration from algorithms in the dynamical systems community (this has also been the basis for the work of (R. Chen et al. 2018)). In this framework, we model the system's temporal evolution, which is different from classical deep learning methods for time-series e.g. RNNs, where it is the transition function between states that is learned. We believe that this learning problem is a particularly natural one for learning dynamical systems, has the benefit of being able to compute states for any given time, and yield favorable inductive biases.

Moreover, the spatial aspect in physical processes has also been treated. This has been achieved by viewing the system's state at a given time for all spatial coordinates as a classical image, and thus applying CNNs, which have proven their efficacy for image processing. Quantitative results are positive, although this method requires the data grid to be regular, which may not necessarily be the case. Since this work, other methods have been proposed that alleviate this problem considering generalized spatial convolutions, perhaps most notably (Z. Li et al. 2021).

Open problems in the context of partial observability still remain, as the learning problem becomes under-specified, and the recovered state may no longer have any physical meaning. To this extent semi-supervised approaches may yield promising results (Ayed et al. 2020); first approaches have been proposed using additional knowledge e.g. the system's initial state or estimates of the latter. These results are still rather preliminary and further experimental and theoretical analysis could be beneficial.

In this work, we have also attacked the problem of generalization through different angles. In many realistic scenarios we have access to some form of prior knowledge, stemming from scientists' careful analysis of the system. This knowledge often takes the form of equations constraining the state's evolution, i.e. differential equations. In Sections 3.2 and 3.3, we have seen how this knowledge could be incorporated into a deep learning framework, wether it may be incomplete (Section 3.2) or inaccurate (Section 3.3). These new approaches have demonstrated their superiority, w.r.t. to both prior-agnostic deep learning methods, and classical physical based modeling approaches. Taking a step back, this seems to partially address the initial question of the role that deep learning has to play in the context of physical processes. Knowledge acquired through extensive study of the physical processes appears essential and should by no means be disregarded; data-driven and handcrafted approaches are complementary and should be used in conjunction in order to yield the best of both worlds. We also believe that taking inspiration from the techniques in the field of dynamical systems is not only useful for developing deep learning algorithms, but is indispensable.

We have also seen that prior knowledge may not always be easily accessed, and data may be too scarce for models to generalize satisfactorily. In this case, scientists modeling natural processes may leverage their knowledge acquired from universal principles that generalize to different systems (e.g. principal of least action), or from the study of systems which are similar. From this, they may gain interesting insights, and perhaps make useful analogies. We take inspiration from this to conceive a machine learning framework in Section 3.4, learning a model by making analogies with other, similar dynamical systems. This has the potential of leveraging the data acquired from these systems, and improving the model's

generalization capabilities as it has access to more data to learn from. But adopting a classical expected risk minimization framework is not sufficient and leads to biased solutions. Instead, in Section 3.4, we provide a novel framework that yields promising results, both from a statistical theoretic and experimental point of view. More generally, this leads the way to methods to improve generalization for learned dynamical systems, without the use of prior knowledge injection. We think that approaching the modeling problem as a meta-learning problem where multiple systems are considered is a propitious research avenue when the data is scarce and prior knowledge is limited.

6.2 A Dynamical Systems Viewpoint on Deep Learning

In this chapter, we have adopted a dynamical systems view on residual deep learning models, by analyzing the network's changes along its layers from a least action perspective. We have observed empirically that residual networks tend to have a low kinetic energy, which has brought us to consider the first time the link between optimal transportation and residual networks. This low energy bias is an implicit one as it is not enforced in the loss, and its cause remains unknown. A very interesting avenue for research would be gain a better theoretical understanding of this phenomenon, potentially making use of recent developments in the implicit regularization effect of gradient descent (Ji et al. 2019; Soudry et al. 2018; Arora et al. 2019a), perhaps using infinite width arguments, as in Section 5.2.

From this link, we have successfully leveraged the theory and algorithms in this field in order to conceive algorithms for the tasks of unsupervised domain translation and classification. For example, we have proven existence and uniqueness results for a wide range of transportation costs, improved generalization and solved unsupervised domain translation tasks that cannot be solved by standard models.

More generally, we believe that adopting a dynamical systems point of view on deep learning, either considering the networks gradually changing its inputs throughout its layers, or simply considering the dynamics of the network's descent as in Section 5.2 is particularly promising.

REFERENCES

Adler, Robert J. (Dec. 1981). *The Geometry Of Random Fields*. Society for Industrial and Applied Mathematics (cit. on p. 289).

Adler, Robert J. (1990). "An Introduction to Continuity, Extrema, and Related Topics for General Gaussian Processes". In: *Lecture Notes-Monograph Series* 12, pp. i–155 (cit. on p. 290).

Akash, Srivastava, Valkov Lazar, Russell Chris, Gutmann Michael U., and Sutton Charles (2017). "VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning". In: *Neural Information Processing Systems* (cit. on p. 158).

Alemohammad, Sina, Zichao Wang, Randall Balestriero, and Richard Baraniuk (2021). "The Recurrent Neural Tangent Kernel". In: *International Conference on Learning Representations* (cit. on p. 148).

Alexandrov, Alexander, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C Maddix, Syama Rangapuram, David Salinas, and Jasper Schulz (2020). "GluonTS: Probabilistic Time Series Models in Python". In: *Journal of Machine Learning Research* 21.116, pp. 1–6 (cit. on p. 324).

Alkinani, Monagi H. and Mahmoud R. El-Sakka (Aug. 2017). "Patch-based models and algorithms for image denoising: a comparative review between patch-based images denoising methods for additive noise reduction". In: *EURASIP Journal on Image and Video Processing* 2017.1, p. 58. URL: https://doi.org/10.1186/s13640-017-0203-4 (cit. on p. 133).

Allen-Zhu, Zeyuan, Yuanzhi Li, and Zhao Song (June 2019). "A Convergence Theory for Deep Learning via Over-Parameterization". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 242–252 (cit. on p. 306).

Almahairi, Amjad, Sai Rajeswar, Alessandro Sordoni, Philip Bachman, and Aaron Courville (2018). "Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data". In: *arXiv preprint arXiv:1802.10151* (cit. on p. 144).

Alvarez, Mauricio A., David Luengo, and Neil D. Lawrence (Nov. 2013). "Linear Latent Force Models Using Gaussian Processes". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.11, pp. 2693–2705 (cit. on p. 15).

Ambrosio, Luigi, Nicola Gigli, and Giuseppe Savare (2005). *Gradient Flows in Metric Spaces and in the Space of Probability Measures* (cit. on pp. 20, 260, 264).

Ambrosio, Luigi, Nicola Gigli, and Giuseppe Savaré (2008). *Gradient Flows. In Metric Spaces and in the Space of Probability Measures*. Birkhäuser Basel (cit. on p. 157).

Amit, Daniel J. (1989). *Modeling Brain Function: The World of Attractor Neural Networks*. Cambridge University Press (cit. on p. 20).

Arbel, Michael, Anna Korba, Adil SALIM, and Arthur Gretton (2019). "Maximum Mean Discrepancy Gradient Flow". In: *Advances in Neural Information Processing Systems*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Vol. 32. Curran Associates, Inc., pp. 6481–6491 (cit. on pp. 156–158).

Arjovsky, Martin and Léon Bottou (2017a). "Towards principled methods for training generative adversarial networks". In: *International Conference on Learning Representations* (cit. on pp. 147, 150, 155).

Arjovsky, Martin, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz (Mar. 2020). "Invariant Risk Minimization". In: *arXiv:1907.02893 [cs, stat]*. arXiv: 1907.02893 (cit. on pp. 16, 90).

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (Aug. 2017b). "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 214–223 (cit. on pp. 149, 150, 158, 302, 303).

Arora, Sanjeev, Nadav Cohen, Wei Hu, and Yuping Luo (2019a). "Implicit Regularization in Deep Matrix Factorization". In: ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (cit. on pp. 110, 180).

Arora, Sanjeev, Simon S. Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang (2019b). "On Exact Computation with an Infinitely Wide Neural Net". In: *Advances in Neural Information Processing Systems*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Vol. 32. Curran Associates, Inc., pp. 8139–8148 (cit. on pp. 148, 291, 301).

Arora, Sanjeev, Simon S. Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu (2020). "Harnessing the Power of Infinitely Wide Deep Nets on Small-data Tasks". In: *International Conference on Learning Representations* (cit. on pp. 148, 160).

Arora, Sanjeev, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang (Aug. 2017). "Generalization and Equilibrium in Generative Adversarial Nets (GANs)". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 224–232 (cit. on p. 150).

Asim, Muhammad, Fahad Shamshad, and Ali Ahmed (2018). "Solving Bilinear Inverse Problems using Deep Generative Priors". In: *CoRR* abs/1802.04073. arXiv: 1802.04073. URL: http://arxiv.org/abs/1802.04073 (cit. on p. 144).

Ayed, Ibrahim, Emmanuel de Bézenac, Arthur Pajot, Julien Brajard, and Patrick Gallinari (2019a). "Learning Dynamical Systems from Partial Observations". In: *CoRR* abs/1902.11136. arXiv: 1902.11136 (cit. on pp. 76, 86).

Ayed, Ibrahim, Emmanuel de Bézenac, Arthur Pajot, Julien Brajard, and Patrick Gallinari (2019b). "Learning dynamical systems from partial observations". In: *arXiv preprint arXiv:1902.11136* (cit. on pp. 61, 66).

Ayed, Ibrahim, Emmanuel de Bézenac, Arthur Pajot, and Patrick Gallinari (2020). "Learning the Spatio-Temporal Dynamics of Physical Processes from Partial Observations". In: *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, pp. 3232–3236. URL: https://doi.org/10.1109/ICASSP40776.2020.9053035 (cit. on pp. 5, 24, 179).

Ayed, Ibrahim, Nicolas Cedilnik, Patrick Gallinari, and Maxime Sermesant (2019c). "EP-Net: Learning Cardiac Electrophysiology Models for Physiology-Based Constraints in Data-Driven Predictions". In: *Functional Imaging and Modeling of the Heart - 10th International Conference, FIMH 2019, Bordeaux, France, June 6-8, 2019, Proceedings*. Ed. by Yves Coudière, Valéry Ozenne, Edward J. Vigmond, and Nejib Zemzemi. Vol. 11504. Lecture Notes in Computer Science. Springer, pp. 55–63 (cit. on p. 61).

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). "Neural Machine Translation by Jointly Learning to Align and Translate". In: URL: http://arxiv.org/abs/1409.0473 (cit. on p. 93).

Bai, Yu, Tengyu Ma, and Andrej Risteski (2019). "Approximability of Discriminators Implies Diversity in GANs". In: *International Conference on Learning Representations* (cit. on p. 148).

Balaji, Yogesh, Mohammadmahdi Sajedi, Neha Mukund Kalibhat, Mucong Ding, Dominik Stöger, Mahdi Soltanolkotabi, and Soheil Feizi (2021). "Understanding Over-parameterization in Generative Adversarial Networks". In: *International Conference on Learning Representations* (cit. on p. 148).

Barber, D. (2011). *Bayesian Reasoning and Machine Learning*. Cambridge University Press (cit. on pp. 167, 321).

Bartlett, Peter L, Dylan J Foster, and Matus J Telgarsky (2017). "Spectrally-normalized margin bounds for neural networks". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., pp. 6240–6249. URL: https://proceedings.neurips.cc/paper/2017/file/b22b257ad0519d4500539da3c8bcf4Paper.pdf (cit. on pp. 82, 242).

Basri, Ronen, Meirav Galun, Amnon Geifman, David Jacobs, Yoni Kasten, and Shira Kritchman (July 2020). "Frequency Bias in Neural Networks for Input of Non-Uniform Density". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 685–694 (cit. on pp. 148, 160).

Bauwens, Luc, Sébastien Laurent, and Jeroen VK Rombouts (2006). "Multivariate GARCH models: a survey". In: *Journal of applied econometrics* 21.1, pp. 79–109 (cit. on p. 162).

Baxter, Jonathan (Mar. 2000). "A Model of Inductive Bias Learning". In: *J. Artif. Int. Res.* 12.1, pp. 149–198 (cit. on pp. 80, 238, 240, 241).

Becker, Philipp, Harit Pandya, Gregor Gebhardt, Cheng Zhao, James Taylor, and Gerhard Neumann (2019). "Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 15, 62).

Behrmann, Jens, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen (2019). "Invertible Residual Networks". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, pp. 573–582 (cit. on pp. 177, 324, 327).

Belkin, Mikhail, Daniel Hsu, Siyuan Ma, and Soumik Mandal (2019). "Reconciling modern machine-learning practice and the classical bias–variance trade-off". In: *PNAS* (cit. on pp. 112, 116).

Belkin, Mikhail, Siyuan Ma, and Soumik Mandal (2018). "To Understand Deep Learning We Need to Understand Kernel Learning". In: *ICML* (cit. on pp. 112, 116).

Benaim, Sagie, Tomer Galanti, and Lior Wolf (2018). "Estimating the Success of Unsupervised Image to Image Translation". In: (cit. on pp. 21, 109).

Benaim, Sagie and Lior Wolf (2017). "One-Sided Unsupervised Domain Mapping". In: (cit. on pp. 21, 109).

Benamou, JD. and Brenier (2000). "A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem". In: (cit. on pp. 99, 114, 261).

Benidis, Konstantinos, Syama Sundar Rangapuram, Valentin Flunkert, Bernie Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. (2020). "Neural forecasting: Introduction and literature overview". In: *arXiv preprint arXiv:2004.10240* (cit. on p. 176).

Béréziat, Dominique and Isabelle Herlin (2015). "Coupling Dynamic Equations and Satellite Images for Modelling Ocean Surface Circulation". In: *Computer Vision, Imaging and Computer Graphics - Theory and Applications: International Joint Conference, VISIGRAPP 2014, Lisbon, Portugal, January 5-8, 2014, Revised Selected Papers*. Ed. by Sebastiano Battiato, Sabine Coquillart, Julien Pettré, Robert S. Laramee, Andreas Kerren, and José Braz. Cham: Springer International Publishing, pp. 191–205 (cit. on pp. 12, 15, 52, 54, 55).

Bernstein, R. L. (1982). "Sea surface temperature estimation using the NOAA 6 satellite advanced very high resolution radiometer". In: *Journal of Geophysical Research: Oceans* 87 (C12), pp. 9455–9465. URL: http://dx.doi.org/10.1029/JC087iC12p09455 (cit. on pp. 43, 49).

Bertsekas, Dimitri P. (1996). *Constrained Optimization and Lagrange Multiplier Methods (Optimization and Neural Computation Series)*. 1st ed. Athena Scientific (cit. on p. 67).

Bezenac, Emmanuel de, Arthur Pajot, and Patrick Gallinari (Feb. 2018). "Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge". In: URL: https://openreview.net/forum?id=By4HsfWAZ (cit. on pp. 5, 42).

Bézenac, Emmanuel de, Ibrahim Ayed, and Patrick Gallinari (2019). "Optimal Unsupervised Domain Translation". In: *CoRR* abs/1906.01292. arXiv: 1906.01292. URL: http://arxiv.org/abs/1906.01292 (cit. on pp. 6, 22, 127).

Bézenac, Emmanuel de, Arthur Pajot, and Patrick Gallinari (2018a). "Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. URL: https://openreview.net/forum?id=By4HsfWAZ (cit. on p. 76).

Bézenac, Emmanuel de, Arthur Pajot, and Patrick Gallinari (2018b). "Deep learning for physical processes: Incorporating prior scientific knowledge". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 61).

Bézenac, Emmanuel de, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski (2020). "Normalizing Kalman Filters for Multivariate Time Series Analysis". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin. URL: https://proceedings.neurips.cc/paper/2020/hash/1f47cef5e38c952f94c5d61726027439-Abstract.html (cit. on pp. 7, 162).

Bietti, Alberto and Francis Bach (2021). "Deep Equals Shallow for ReLU Networks in Kernel Regimes". In: *International Conference on Learning Representations* (cit. on p. 148).

Bietti, Alberto and Julien Mairal (2019a). "On the Inductive Bias of Neural Tangent Kernels". In: *Advances in Neural Information Processing Systems*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Vol. 32. Curran Associates, Inc., pp. 12873–12884 (cit. on pp. 148, 154, 303, 306).

Bietti, Alberto, Grégoire Mialon, Dexiong Chen, and Julien Mairal (Sept. 2019b). "A Kernel Perspective for Regularizing Deep Neural Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika

Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 664–674 (cit. on p. 90).

Bjorck, Nils et al. (2018). "Understanding Batch Normalization". In: *NIPS* (cit. on pp. 269, 273).

Bolley, François (2008). "Separability and Completeness for the Wasserstein distance". In: *Séminaire de Probabilités XLI*. Springer (cit. on p. 121).

Bora, Ashish, Ajil Jalal, Eric Price, and Alexandros G. Dimakis (Mar. 2017). "Compressed Sensing using Generative Models". In: *arXiv:1703.03208 [cs, math, stat]*. URL: http://arxiv.org/abs/1703.03208 (cit. on pp. 131, 139, 144).

Bora, Ashish, Eric Price, and Alexandros G. Dimakis (2018a). "AmbientGAN: Generative models from lossy measurements". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Hy7fDog0b (cit. on p. 8).

Bora, Ashish, Eric Price, and Alexandros G. Dimakis (2018b). "AmbientGAN: Generative models from lossy measurements". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Hy7fDog0b (cit. on pp. 135, 138, 139).

Boyat, Ajay Kumar and Brijendra Kumar Joshi (2015). "A Review Paper: Noise Models in Digital Image Processing". In: *CoRR* abs/1505.03489 (cit. on p. 133).

Bradbury, James, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang (2018). *JAX: composable transformations of Python+NumPy programs*. Version 0.2.12. URL: http://github.com/google/jax (cit. on p. 314).

Brock, Andrew, Jeff Donahue, and Karen Simonyan (2019). "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: *International Conference on Learning Representations* (cit. on pp. 146, 302).

Brunton, Steven L., Joshua L. Proctor, and J. Nathan Kutz (2016). "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". In: *Proceedings of the National Academy of Sciences* 113.15, pp. 3932–3937 (cit. on pp. 61, 66, 76).

Candès, Emmanuel J., Justin K. Romberg, and Terence Tao (2005). "Stable signal recovery from incomplete and inaccurate measurements". In: *Communications on Pure and Applied Mathematics* 59.8, pp. 1207–1223. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.20124. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.20124 (cit. on p. 130).

Cao, Wei, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li (2018). "BRITS: Bidirectional Recurrent Imputation for Time Series". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-

Bianchi, and Roman Garnett, pp. 6776–6786. URL: http://papers.nips.cc/paper/7911-brits-bidirectional-recurrent-imputation-for-time-series (cit. on p. 169).

Carrassi, Alberto, Marc Bocquet, Laurent Bertino, and Geir Evensen (2018). "Data assimilation in the geosciences: An overview of methods, issues, and perspectives". In: *Wiley Interdisciplinary Reviews: Climate Change* 9.5, e535. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/wcc.535 (cit. on pp. 12, 14, 15, 29).

Chambolle, Antonin (Jan. 2004). "An Algorithm for Total Variation Minimization and Applications". In: *Journal of Mathematical Imaging and Vision* 20.1, pp. 89–97. URL: https://doi.org/10.1023/B:JMIV.0000011325.36760.1e (cit. on p. 140).

Chang, Bo et al. (2018). "Reversible Architectures for Arbitrarily Deep Residual Neural Networks". In: *AAAI* (cit. on pp. 21, 22, 112, 127).

Chen, Lin and Sheng Xu (2021). "Deep Neural Tangent Kernel and Laplace Kernel Have the Same RKHS". In: *International Conference on Learning Representations* (cit. on pp. 148, 306).

Chen, R.T.Q, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud (2018). "Neural Ordinary Differential Equations". In: *NIPS* (cit. on pp. 15, 21, 127, 178).

Chen, Ricky T. Q., Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud (2018). "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., pp. 6571–6583 (cit. on pp. 76, 87).

Chen, Tian Qi, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud (June 2018a). "Neural Ordinary Differential Equations". In: *arXiv:1806.07366 [cs, stat]*. URL: http://arxiv.org/abs/1806.07366 (cit. on pp. 14, 29, 37).

Chen, Tian Qi, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud (2018b). "Neural Ordinary Differential Equations". In: *CoRR* abs/1806.07366 (cit. on p. 103).

Chen, Tian Qi, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud (2018c). "Neural ordinary differential equations". In: *Advances in neural information processing systems (NeurIPS)*, pp. 6571–6583 (cit. on pp. 61, 63, 69, 228, 230, 234, 235).

Chen, Wen-Hua (2004). "Disturbance observer based control for nonlinear systems". In: *IEEE/ASME transactions on mechatronics* 9.4, pp. 706–710 (cit. on pp. 16, 62).

Chen, Zhengdao, Jianyu Zhang, Martin Arjovsky, and Léon Bottou (2020). "Symplectic Recurrent Neural Networks". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 61).

Chizat, Lénaïc and Francis Bach (2018). "On the Global Convergence of Gradient Descent for Over-parameterized Models using Optimal Transport". In:

Advances in Neural Information Processing Systems. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., pp. 3036–3046. URL: https://proceedings.neurips.cc/paper/2018/file/a1afc58c6ca9540d057299ec3016d726-Paper.pdf (cit. on p. 244).

Cho, Kyunghyun, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (Oct. 2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1724–1734. URL: https://www.aclweb.org/anthology/D14-1179 (cit. on pp. 87, 90).

Choi, Edward, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart (2016). "RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism". In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3504–3512 (cit. on p. 60).

Choi, Yunjey, Min-Je Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo (2018). "StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation". In: (cit. on p. 93).

Choromanska, Anna, Mikael Henaff, Michaël Mathieu, Gérard Ben Arous, and Yann LeCun (2014). "The Loss Surface of Multilayer Networks". In: *CoRR* abs/1412.0233. arXiv: 1412.0233. URL: http://arxiv.org/abs/1412.0233 (cit. on p. 20).

Chung, Yu-An, Wei-Hung Weng, Schrasing Tong, and James Glass (2018). "Unsupervised Cross-Modal Alignment of Speech and Text Embedding Spaces". In: (cit. on p. 93).

Corless, Rob M., Gaston H. Gonnet, D. E. G. Hare, David J. Jeffrey, and Donald E. Knuth (Dec. 1996). "On the Lambert W function". In: *Advances in Computational Mathematics* 5.1, pp. 329–359 (cit. on p. 300).

Corless, Robert M., Hui Ding, Nicholas J. Higham, and David J. Jeffrey (2007). "The Solution of $S\exp(S) = A$ is Not Always the Lambert W Function of A". In: *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*. ISSAC '07. Waterloo, Ontario, Canada: Association for Computing Machinery, pp. 116–121 (cit. on p. 301).

Courtier, Philippe, J-N Thépaut, and Anthony Hollingsworth (1994). "A strategy for operational implementation of 4D-Var, using an incremental approach". In: *Quarterly Journal of the Royal Meteorological Society* 120.519, pp. 1367–1387 (cit. on pp. 15, 62).

Courty, Nicolas, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy (2015). "Optimal Transport for Domain Adaptation". In: *CoRR* abs/1507.00504 (cit. on pp. 21, 109).

Cranmer, Miles, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho (2020). "Lagrangian neural networks". In: *ICLR 2020 Deep Differential Equations Workshop* (cit. on pp. 66, 231).

Crutchfield, James P. and Bruce S. Mcnamara (1987). "Equations of motion from a data series". In: *Complex Systems*, p. 452 (cit. on p. 15).

Damelin, SB and NS Hoang (2018). "On surface completion and image inpainting by biharmonic functions: Numerical aspects". In: *International Journal of Mathematics and Mathematical Sciences* 2018 (cit. on p. 140).

Damodaran, Bharath Bhushan, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty (2018). "DeepJDOT: Deep Joint distribution optimal transport for unsupervised domain adaptation". In: (cit. on pp. 21, 109).

De Palma, Giacomo, Bobak Kiani, and Seth Lloyd (2019). "Random deep neural networks are biased towards simple functions". In: *NIPS* (cit. on p. 112).

Deisenroth, M. P., R. D. Turner, M. F. Huber, U. D. Hanebeck, and C. E. Rasmussen (2012). "Robust Filtering and Smoothing with Gaussian Processes". In: *IEEE Transactions on Automatic Control* 57.7, pp. 1865–1871 (cit. on p. 176).

Dheeru, Dua and Efi Karra Taniskidou (2017). *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml (cit. on p. 326).

DIMET, FRANÇOIS-XAVIER LE and OLIVIER TALAGRAND (1986). "Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects". In: *Tellus A* 38A.2, pp. 97–110. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1600-0870.1986.tb00459.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1600-0870.1986.tb00459.x (cit. on p. 12).

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio (2016). "Density estimation using Real NVP". In: *CoRR* abs/1605.08803. arXiv: 1605.08803. URL: http://arxiv.org/abs/1605.08803 (cit. on p. 131).

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio (2017). "Density estimation using Real NVP". In: *5th International Conference on Learning Representations, ICLR 2017* (cit. on pp. 165, 170, 177, 324).

Donà, Jérémie, Jean-Yves Franceschi, Sylvain Lamprier, and Patrick Gallinari (2020). "PDE-driven spatiotemporal disentanglement". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 61).

Dormand, J.R. and P.J. Prince (1980). "A family of embedded Runge-Kutta formulae". In: *Journal of Computational and Applied Mathematics* 6.1, pp. 19–26. URL: http://www.sciencedirect.com/science/article/pii/0771050X80900133 (cit. on p. 85).

Dormand, John R and Peter J Prince (1980). "A family of embedded Runge-Kutta formulae". In: *Journal of computational and applied mathematics* 6.1, pp. 19–26 (cit. on p. 228).

Durbin, James and Siem Jan Koopman (2012). *Time series analysis by state space methods*. Vol. 38. Oxford University Press (cit. on p. 162).

Dwork, Cynthia, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel (2012). "Fairness through awareness". In: (cit. on p. 107).

E, Weinan (Mar. 2017). "A Proposal on Machine Learning via Dynamical Systems". In: *Communications in Mathematics and Statistics* 5.1, pp. 1–11. URL: https://doi.org/10.1007/s40304-017-0103-z (cit. on pp. 1, 5).

Fablet, Ronan, Said Ouala, and Cédric Herzet (2017). "Bilinear residual Neural Network for the identification and forecasting of dynamical systems". In: *CoRR* abs/1712.07003. URL: http://arxiv.org/abs/1712.07003 (cit. on p. 15).

Fablet, Ronan, Said Ouala, and Cédric Herzet (2018). "Bilinear Residual Neural Network for the Identification and Forecasting of Geophysical Dynamics". In: *26th European Signal Processing Conference, EUSIPCO 2018, Roma, Italy, September 3-7, 2018*. IEEE, pp. 1477–1481. URL: https://doi.org/10.23919/EUSIPCO.2018.8553492 (cit. on p. 15).

Fan, Zhou and Zhichao Wang (2020). "Spectra of the Conjugate Kernel and Neural Tangent Kernel for linear-width neural networks". In: *Advances in Neural Information Processing Systems*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria Florina Balcan, and Hsuan-Tien Lin. Vol. 33. Curran Associates, Inc., pp. 7710–7721 (cit. on pp. 148, 306).

Farkas, Bálint and Sven-Ake Wegner (2016). "Variations on Barbălat's Lemma". In: *The American Mathematical Monthly* 123.8, pp. 825–830 (cit. on p. 294).

Felix, Rafael, Vijay B. G. Kumar, Ian Reid, and Gustavo Carneiro (2018). "Multi-modal Cycle-consistent Generalized Zero-Shot Learning". In: (cit. on p. 93).

Feydy, Jean, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré (Apr. 2019). "Interpolating between Optimal Transport and MMD using Sinkhorn Divergences". In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 2681–2690 (cit. on pp. 161, 315, 316).

Feynman, Richard P. (2005). "The Principle of Least Action in Quantum Mechanics". In: *Feynman's Thesis - A New Approach to Quantum Theory*. World Scientific Publishing (cit. on pp. 17, 113).

Figalli, A. (2017). *The Monge-Ampère Equation and Its Applications*. Zurich lectures in advanced mathematics. European Mathematical Society. URL: https://books.google.fr/books?id=e45aMQAACAAJ (cit. on pp. 20, 260, 264).

Figalli, Alessio (2008). *Optimal transportation and action-minimizing measures*. Tesi 8. Pisa: Ed. della normale. URL: http://www.sudoc.fr/128156937 (cit. on p. 99).

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *CoRR* abs/1703.03400.

arXiv: 1703.03400. URL: http://arxiv.org/abs/1703.03400 (cit. on pp. 17, 90).

Finn, Chelsea, Ian J. Goodfellow, and Sergey Levine (2016). "Unsupervised Learning for Physical Interaction through Video Prediction". In: *CoRR* abs/1605.07157. URL: http://arxiv.org/abs/1605.07157 (cit. on p. 48).

Fletcher, James and Warren Moors (Apr. 2014). "Chebyshev Sets". In: *Journal of the Australian Mathematical Society* 98, pp. 161–231 (cit. on p. 223).

Fraccaro, Marco, Simon Kamronn, Ulrich Paquet, and Ole Winther (2017). "A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning". In: *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., pp. 3601–3610 (cit. on pp. 163, 169, 177).

Franceschi, Jean-Yves, Emmanuel de Bézenac, Ibrahim Ayed, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari (2021). "A Neural Tangent Kernel Perspective of GANs". In: *CoRR* abs/2106.05566. arXiv: 2106.05566. URL: https://arxiv.org/abs/2106.05566 (cit. on p. 7).

Fresca, Stefania, Andrea Manzoni, Luca Dedè, and Alfio Quarteroni (2020). *Deep learning-based reduced order models in cardiac electrophysiology*. Vol. 15. 10 October, pp. 1–32 (cit. on p. 76).

Galanti, Tomer, Lior Wolf, and Sagie Benaim (2018). "The Role of Minimal Complexity Functions in Unsupervised Learning of Semantic Mappings". In: (cit. on pp. 21, 94–96, 109).

Garcia-Morales, V., J. Pellicer, and J. Manzanares (2008). "Thermodynamics based on the principle of least abbreviated action". In: *Annals of Physics* (cit. on pp. 17, 113).

Gasthaus, Jan, Konstantinos Benidis, Yuyang Wang, Syama S. Rangapuram, David Salinas, Valentin Flunkert, and Tim Januschowski (2019). "Probabilistic Forecasting with Spline Quantile Function RNNs". In: *AISTATS* (cit. on pp. 176, 326).

Gebhardt, Gregor H. W., Andras Kupcsik, and Gerhard Neumann (2017). "The Kernel Kalman Rule — Efficient Nonparametric Inference with Recursive Least Squares". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI'17, pp. 3754–3760 (cit. on p. 177).

Geiger, Mario, Stefano Spigler, Arthur Jacot, and Matthieu Wyart (Nov. 2020). "Disentangling feature and lazy training in deep neural networks". In: *Journal of Statistical Mechanics: Theory and Experiment* 2020.11, p. 113301 (cit. on pp. 161, 302).

Gentine, P., M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis (2018). "Could Machine Learning Break the Convection Parameterization Deadlock?" In: *Geophysical Research Letters* 45.11, pp. 5742–5751. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2018GL078202 (cit. on p. 61).

Gneiting, Tilmann and Adrian E Raftery (2007). "Strictly proper scoring rules, prediction, and estimation". In: *Journal of the American Statistical Association* 102.477, pp. 359–378 (cit. on pp. 173, 329).

Gong, Rui, Wen Li, Yuhua Chen, and Luc Van Gool (n.d.). "DLOW: Domain Flow for Adaptation and Generalization". In: () (cit. on pp. 21, 109).

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger. Vol. 27. Curran Associates, Inc., pp. 2672–2680 (cit. on pp. 146, 147, 149, 150).

Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014a). "Generative Adversarial Nets". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. MIT Press, pp. 2672–2680. URL: http://dl.acm.org/citation.cfm?id=2969033.2969125 (cit. on p. 21).

Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014b). "Generative Adversarial Nets". In: (cit. on p. 131).

Gray, C. G. (2009). "Principle of Least Action". In: *Scholarpedia* (cit. on pp. 17, 113).

Gretton, Arthur, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola (2007). "A Kernel Method for the Two-Sample-Problem". In: *Advances in Neural Information Processing Systems*. Ed. by B. Schölkopf, J. Platt, and T. Hoffman. Vol. 19. MIT Press, pp. 513–520 (cit. on p. 155).

Gretton, Arthur, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola (2012). "A Kernel Two-Sample Test". In: *Journal of Machine Learning Research* 13.25, pp. 723–773 (cit. on p. 155).

Greydanus, Samuel, Misko Dzamba, and Jason Yosinski (2019). "Hamiltonian neural networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 15353–15363 (cit. on pp. 61, 66, 69, 228, 231, 234).

Gunzburger, Max D. (2002). *Perspectives in Flow Control and Optimization*. USA: Society for Industrial and Applied Mathematics (cit. on p. 13).

Haber, Eldad, Keegan Lensink, Eran Treister, and Lars Ruthotto (2019). "IMEXnet A Forward Stable Deep Neural Network". In: *ICML* (cit. on pp. 21, 127).

Hamming, Richard W. (1973). *Numerical Methods for Scientists and Engineers*. USA: McGraw-Hill, Inc. (cit. on p. 11).

Harris, Charles R., K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'io, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy,

Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant (Sept. 2020). "Array programming with NumPy". In: *Nature* 585.7825, pp. 357–362. URL: https://doi.org/10.1038/s41586-020-2649-2 (cit. on p. 85).

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2001). *The Elements of Statistical Learning*. Springer-Verlag (cit. on p. 112).

Hauser, M. (2019). "On Residual Networks Learning a Perturbation from Identity". In: *arXiv* (cit. on pp. 21, 127).

Haussler, David (1992). "Decision theoretic generalizations of the PAC model for neural net and other learning applications". In: *Information and Computation* 100.1, pp. 78–150. URL: http://www.sciencedirect.com/science/article/pii/089054019290010D (cit. on pp. 83, 244).

He, Junxian, Graham Neubig, and Taylor Berg-Kirkpatrick (2018). "Unsupervised Learning of Syntactic Structure with Invertible Neural Projections". In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)* (cit. on p. 177).

He, K., X. Zhan, S. Ren, and J. Sun (2016). "Identity mappings in deep residual networks". In: *ECCV* (cit. on pp. 21, 112, 127, 268).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (Dec. 2015). "Deep Residual Learning for Image Recognition". In: *arXiv:1512.03385 [cs]*. URL: http://arxiv.org/abs/1512.03385 (cit. on pp. 32, 51).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016a). "Deep Residual Learning for Image Recognition". In: *CVPR* (cit. on pp. 21, 112, 115, 127).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016b). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778 (cit. on p. 273).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (June 2016c). "Deep Residual Learning for Image Recognition". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778 (cit. on p. 301).

Higham, Nicholas J. (2008). *Functions of matrices: theory and computation*. Society for Industrial and Applied Mathematics (cit. on p. 295).

Hinton, Geoffrey E. and Terrence J. Sejnowski (1983). "Analyzing cooperative computation". In: *Proceedings of the Fifth Annual Conference of the Cognitive Science Society, Rochester NY* (cit. on p. 20).

Hopfield, J J (1982). "Neural networks and physical systems with emergent collective computational abilities". In: *Proceedings of the National Academy of Sciences* 79.8, pp. 2554–2558. eprint: https://www.pnas.org/content/79/8/2554.full.pdf. URL: https://www.pnas.org/content/79/8/2554 (cit. on p. 20).

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5, pp. 359–366 (cit. on p. 306).

Hron, Jiri, Yasaman Bahri, Jascha Sohl-Dickstein, and Roman Novak (July 2020). "Infinite attention: NNGP and NTK for deep attention networks". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 4376–4386 (cit. on p. 302).

Huang, Kaixuan, Yuqing Wang, Molei Tao, and Tuo Zhao (2020). "Why Do Deep Residual Networks Generalize Better than Deep Feedforward Networks? — A Neural Tangent Kernel Perspective". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 2698–2709 (cit. on p. 291).

Hyndman, Rob, Anne Koehler, Keith Ord, and Ralph Snyder (2008). "Forecasting with exponential smoothing. The state space approach". In: (cit. on pp. 162, 165, 170, 173).

Iacono, Roberto and John P. Boyd (2017). "New approximations to the principal real-valued branch of the Lambert W-function". In: *Advances in Computational Mathematics* 43.6, pp. 1403–1436 (cit. on p. 301).

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros (2016a). "Image-to-Image Translation with Conditional Adversarial Networks". In: *arxiv* (cit. on p. 93).

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros (2016b). "Image-to-Image Translation with Conditional Adversarial Networks". In: *CoRR* abs/1611.07004. arXiv: 1611.07004. URL: http://arxiv.org/abs/1611.07004 (cit. on pp. 140, 275).

Jacot, Arthur, Franck Gabriel, François Ged, and Clément Hongler (2019). "Order and Chaos: NTK views on DNN Normalization, Checkerboard and Boundary Artifacts". In: *arXiv preprint arXiv:1907.05715* (cit. on p. 148).

Jacot, Arthur, Franck Gabriel, and Clement Hongler (2018a). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *NIPS* (cit. on pp. 8, 112, 116).

Jacot, Arthur, Franck Gabriel, and Clement Hongler (2018b). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf (cit. on p. 21).

Jacot, Arthur, Franck Gabriel, and Clement Hongler (2018c). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by Samy Bengio, Hanna Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. Vol. 31. Curran Associates, Inc., pp. 8580–8589 (cit. on pp. 147, 148, 151, 152, 156, 289, 291, 305, 306, 316).

Jaderberg, Max, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu (2015). "Spatial Transformer Networks". In: *CoRR* abs/1506.02025. URL: http://arxiv.org/abs/1506.02025 (cit. on p. 48).

Jain, Niharika, Alberto Olmo, Sailik Sengupta, Lydia Manikonda, and Subbarao Kambhampati (2020). "Imperfect imaGANation: Implications of GANs exacerbating biases on facial data augmentation and Snapchat selfie lenses". In: *arXiv preprint arXiv:2001.09528* (cit. on p. 161).

Janner, Michael, Justin Fu, Marvin Zhang, and Sergey Levine (2019). "When to trust your model: Model-based policy optimization". In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 12519–12530 (cit. on pp. 16, 62).

Jastrzebski, Stanislaw et al. (2018). "Residual connections encourage iterative inference". In: *ICLR* (cit. on pp. 21, 116, 127).

Ji, Ziwei and Matus Telgarsky (2019). "Gradient descent aligns the layers of deep linear networks". In: (cit. on pp. 110, 180).

Ji, Ziwei, Matus Telgarsky, and Ruicheng Xian (2020). "Neural tangent kernels, transportation mappings, and universal approximation". In: *International Conference on Learning Representations* (cit. on pp. 160, 306).

Johnson, Gordon G (1987). "A nonconvex set which has the unique nearest point property". In: *Journal of Approximation Theory* 51.4, pp. 289–332 (cit. on p. 223).

Julier, Simon J and Jeffrey K Uhlmann (2004). "Unscented filtering and nonlinear estimation". In: *Proceedings of the IEEE* 92.3, pp. 401–422 (cit. on pp. 166, 176).

Kaae Sønderby, Casper, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár (2017). "Amortised MAP Inference for Image Super-resolution". In: *International Conference on Learning Representations* (cit. on pp. 156, 304).

Kalinicheva, Ekaterina, Jérémie Sublime, and Maria Trocan (2019). "Change Detection in Satellite Images Using Reconstruction Errors of Joint Autoencoders". In: *Artificial Neural Networks and Machine Learning – ICANN 2019: Image Processing*. Ed. by Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis. Cham: Springer International Publishing, pp. 637–648 (cit. on p. 14).

Kalman, Rudolph Emil (1960). "A new approach to linear filtering and prediction problems". In: *Journal of basic Engineering* 82.1, pp. 35–45 (cit. on pp. 15, 62, 176).

Karkar, Skander, Ibrahim Ayed, Emmanuel de Bézenac, and Patrick Gallinari (2020). "A Principle of Least Action for the Training of Neural Networks". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part II*. Ed. by Frank Hutter, Kristian Kersting, Jefrey Lijffijt, and Isabel Valera. Vol. 12458. Lecture Notes in Computer Science. Springer, pp. 101–117. URL: https://doi.org/10.1007/978-3-030-67661-2%5C_7 (cit. on pp. 6, 111).

Karpatne, Anuj, Gowtham Atluri, James H. Faghmous, Michael S. Steinbach, Arindam Banerjee, Auroop R. Ganguly, Shashi Shekhar, Nagiza F. Samatova, and Vipin Kumar (2017). "Theory-Guided Data Science: A New Paradigm for

Scientific Discovery from Data". In: *IEEE Trans. Knowl. Data Eng.* 29.10, pp. 2318–2331. URL: https://doi.org/10.1109/TKDE.2017.2720168 (cit. on p. 14).

Karras, Tero, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila (June 2020). "Analyzing and Improving the Image Quality of StyleGAN". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8107–8116 (cit. on p. 146).

Kidger, Patrick and Terry Lyons (Sept. 2020). "Universal Approximation with Deep Narrow Networks". In: *Proceedings of Thirty Third Conference on Learning Theory*. Ed. by Jacob Abernethy and Shivani Agarwal. Vol. 125. Proceedings of Machine Learning Research. PMLR, pp. 2306–2327. URL: http://proceedings.mlr.press/v125/kidger20a.html (cit. on p. 243).

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980. arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980 (cit. on p. 273).

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun (cit. on p. 90).

Kingma, Diederik P. and Prafulla Dhariwal (2018). "Glow: Generative Flow with Invertible 1x1 Convolutions". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pp. 10236–10245 (cit. on pp. 165, 171, 177, 324).

Kingma, Diederik P. and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *CoRR* abs/1312.6114. arXiv: 1312.6114. URL: http://arxiv.org/abs/1312.6114 (cit. on p. 131).

Klaasen, Gene A. and William C. Troy (1984). "Stationary Wave Solutions of a System of Reaction-Diffusion Equations Derived from the FitzHugh–Nagumo Equations". In: *SIAM Journal on Applied Mathematics* 44.1, pp. 96–110 (cit. on pp. 68, 229).

Ko, J. and D. Fox (2011). "Learning GP-Bayes Filters via Gaussian process latent variable models". In: *Autonomous Robots* 30, pp. 3–23 (cit. on p. 176).

Krishnan, Rahul G, Uri Shalit, and David Sontag (2017). "Structured inference networks for nonlinear state space models". In: *31st AAAI Conference on Artificial Intelligence* (cit. on pp. 163, 173, 177).

Krueger, David, Ethan Caballero, Jörn-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Rémi Le Priol, and Aaron C. Courville (2020). "Out-of-Distribution Generalization via Risk Extrapolation (REx)". In: *CoRR* abs/2003.00688. arXiv: 2003.00688. URL: https://arxiv.org/abs/2003.00688 (cit. on pp. 16, 90).

Kurach, Karol, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly (June 2019). "A Large-Scale Study on Regularization and Normalization in GANs". In: *Proceedings of the 36th International Conference on Machine Learning*.

Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 3581–3590 (cit. on p. 146).

Lai, Guokun, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu (2017). "Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks". In: *CoRR* abs/1703.07015. arXiv: 1703.07015 (cit. on p. 326).

Lai, Guokun, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu (2018). "Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks". In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. Association for Computing Machinery, pp. 95–104 (cit. on p. 176).

Lamb, Alex, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio (Oct. 2016). "Professor Forcing: A New Algorithm for Training Recurrent Networks". In: *arXiv:1610.09038 [cs, stat]*. arXiv: 1610.09038. (Visited on 08/20/2020) (cit. on p. 90).

Lample, Guillaume, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato (2018). "Unsupervised Machine Translation Using Monolingual Corpora Only". In: (cit. on p. 93).

Lample, Guillaume, Ludovic Denoyer, and Marc'Aurelio Ranzato (2017). "Unsupervised Machine Translation Using Monolingual Corpora Only". In: *CoRR* abs/1711.00043. arXiv: 1711.00043. URL: http://arxiv.org/abs/1711.00043 (cit. on p. 144).

Laptev, Nikolay, Jason Yosinsk, Li Li Erran, and Slawek Smyl (2017). "Time-series Extreme Event Forecasting with Neural Networks at Uber". In: *ICML Time Series Workshop* (cit. on p. 176).

Large, William and Stephen Yeager (May 2004). *Diurnal to Decadal Global Forcing for Ocean and Sea-Ice Models: The Data Sets and Flux Climatologies* (cit. on p. 61).

Le Guen, Vincent and Nicolas Thome (2020a). "A Deep Physical Model for Solar Irradiance Forecasting with Fisheye Images". In: *CVPR 2020 OmniCV workshop* (cit. on p. 231).

Le Guen, Vincent and Nicolas Thome (2020b). "Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction". In: *Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 231).

Lecun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep learning". In: *Nature* 521.7553, pp. 436–444 (cit. on p. 1).

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (Nov. 1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324 (cit. on pp. 159, 309, 315).

LeCun, Yann, D Touresky, G Hinton, and T Sejnowski (1988). "A theoretical framework for back-propagation". In: *Proceedings of the 1988 connectionist models summer school*. Vol. 1. CMU, Pittsburgh, Pa: Morgan Kaufmann, pp. 21–28 (cit. on pp. 14, 30).

Ledig, Christian, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi (2016). "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *CoRR* abs/1609.04802. arXiv: 1609.04802. URL: http://arxiv.org/abs/1609.04802 (cit. on pp. 131, 144).

Lee, Jaehoon, Samuel S. Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein (2020). "Finite Versus Infinite Neural Networks: an Empirical Study". In: *Advances in Neural Information Processing Systems*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Vol. 33. Curran Associates, Inc., pp. 15156–15172 (cit. on p. 148).

Lee, Jaehoon, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington (2019a). "Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, pp. 8570–8581. URL: https://proceedings.neurips.cc/paper/2019/hash/0d1a9651497a38d8b1c3871c84528bd4-Abstract.html (cit. on p. 21).

Lee, Jaehoon, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington (2019b). "Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent". In: *Advances in Neural Information Processing Systems*. Ed. by Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Roman Garnett. Vol. 32. Curran Associates, Inc., pp. 8570–8581 (cit. on p. 148).

Lee, Seungjun, Haesang Yang, and Woojae Seong (2021). "Identifying Physical Law of Hamiltonian Systems via Meta-Learning". In: *CoRR* abs/2102.11544. arXiv: 2102.11544. URL: https://arxiv.org/abs/2102.11544 (cit. on pp. 17, 90).

Lehtinen, Jaakko, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila (Mar. 2018). "Noise2Noise: Learning Image Restoration without Clean Data". In: *arXiv:1803.04189 [cs, stat]*. arXiv: 1803.04189. URL: http://arxiv.org/abs/1803.04189 (visited on 07/13/2018) (cit. on p. 144).

Leipnik, Roy B. and Charles E. M. Pearce (2007). "The multivariate Faà di Bruno formula and multivariate Taylor expansions with explicit integral remainder term". In: *The ANZIAM Journal* 48.3, pp. 327–341 (cit. on p. 290).

Leshno, Moshe, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken (1993). "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural Networks* 6.6, pp. 861–867 (cit. on p. 306).

Li, Qianxiao, Long Chen, Cheng Tai, and Weinan E (2018). "Maximum Principle Based Algorithms for Deep Learning". In: *JMLR* (cit. on p. 115).

Li, Shihua, Jun Yang, Wen-Hua Chen, and Xisong Chen (2014). *Disturbance observer-based control: methods and applications*. CRC press (cit. on pp. 16, 62).

Li, Zongyi, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar (2021). "Fourier Neural Operator for Parametric Partial Differential Equations". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=c8P9NQVtmnO (cit. on pp. 84–86, 90, 91, 179, 244, 250).

Lim, Jae Hyun and Jong Chul Ye (2017). "Geometric GAN". In: *arXiv preprint arXiv:1705.02894* (cit. on p. 149).

Ling, Julia, Andrew Kurzawski, and Jeremy Templeton (Nov. 2016). "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance". In: *Journal of Fluid Mechanics* 807, pp. 155–166 (cit. on p. 49).

Littwin, Etai, Tomer Galanti, Lior Wolf, and Greg Yang (2020a). "On Infinite-Width Hypernetworks". In: *Advances in Neural Information Processing Systems*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria Florina Balcan, and Hsuan-Tien Lin. Vol. 33. Curran Associates, Inc., pp. 13226–13237 (cit. on p. 148).

Littwin, Etai, Ben Myara, Sima Sabah, Joshua Susskind, Shuangfei Zhai, and Oren Golan (2020b). "Collegial Ensembles". In: *Advances in Neural Information Processing Systems*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria Florina Balcan, and Hsuan-Tien Lin. Vol. 33. Curran Associates, Inc., pp. 18738–18748 (cit. on p. 148).

Liu, Chaoyue, Libin Zhu, and Misha Belkin (2020). "On the linearity of large non-linear models: when and why the tangent kernel is constant". In: *Advances in Neural Information Processing Systems*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria Florina Balcan, and Hsuan-Tien Lin. Vol. 33. Curran Associates, Inc., pp. 15954–15964 (cit. on pp. 148, 152, 301).

Liu, Han, John Lafferty, and Larry Wasserman (2009). "The nonparanormal: Semiparametric estimation of high dimensional undirected graphs". In: *Journal of Machine Learning Research* 10.Oct, pp. 2295–2328 (cit. on p. 166).

Liu, Jun S and Rong Chen (1998). "Sequential Monte Carlo methods for dynamic systems". In: *Journal of the American statistical association* 93.443, pp. 1032–1044 (cit. on p. 176).

Liu, Ming-Yu, Xun Huang, Jiahui Yu, Ting-Chun Wang, and Arun Mallya (2021). "Generative Adversarial Networks for Image and Video Synthesis: Algorithms and Applications". In: *Proceedings of the IEEE* 109.5, pp. 839–862 (cit. on p. 146).

Liu, Qiang and Dilin Wang (2016). "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm". In: *Advances in Neural Information*

Processing Systems. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett. Vol. 29. Curran Associates, Inc. (cit. on p. 160).

Liu, Shuang, Olivier Bousquet, and Kamalika Chaudhuri (2017). "Approximation and Convergence Properties of Generative Adversarial Learning". In: *Advances in Neural Information Processing Systems*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Vol. 30. Curran Associates, Inc., pp. 5545–5553 (cit. on p. 148).

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)* (cit. on p. 131).

Long, Yun, Xueyuan She, and Saibal Mukhopadhyay (2018). "HybridNet: integrating model-based and data-driven learning to predict evolution of dynamical systems". In: *Conference on Robot Learning (CoRL)* (cit. on pp. 16, 63).

Long, Zichao, Yiping Lu, and Bin Dong (2018a). "PDE-Net 2.0: Learning PDEs from Data with A Numeric-Symbolic Hybrid Deep Network". In: *CoRR* abs/1812.04426. arXiv: 1812.04426. URL: http://arxiv.org/abs/1812.04426 (cit. on p. 91).

Long, Zichao, Yiping Lu, Xianzhong Ma, and Bin Dong (2018b). "PDE-Net: Learning PDEs from Data". In: *ICML*, pp. 3214–3222 (cit. on p. 15).

Long, Zichao, Yiping Lu, Xianzhong Ma, and Bin Dong (2018c). "PDE-Net: Learning PDEs from Data". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 16, 61, 63).

Long, Zichao, Yiping Lu, Xianzhong Ma, and Bin Dong (2018d). "PDE-Net: Learning PDEs from Data". In: *International Conference on Machine Learning*, pp. 3214–3222 (cit. on p. 76).

Lorenc, Andrew (1986). "Analysis methods for numerical weather prediction". In: *Quarterly Journal of the Royal Meteorological Society* 112, pp. 1177–1194 (cit. on p. 15).

Lotka, Alfred J. (1926). "ELEMENTS OF PHYSICAL BIOLOGY". In: *Science Progress in the Twentieth Century (1919-1933)* 21.82, pp. 341–343. URL: http://www.jstor.org/stable/43430362 (cit. on p. 84).

Lu, Guansong, Zhiming Zhou, Yuxuan Song, Kan Ren, and Yong Yu (2018). "Guiding the One-to-one Mapping in CycleGAN via Optimal Transport". In: (cit. on pp. 21, 109).

Lu, Yiping, Aoxiao Zhong, Quanzheng Li, and Bin Dong (2018). "Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations". In: *ICML* (cit. on pp. 21, 112, 127).

Lucic, Mario, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet (2018). "Are GANs Created Equal? A Large-Scale Study". In: *Advances in Neural Information Processing Systems*. Ed. by Samy Bengio, Hanna Wallach,

Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. Vol. 31. Curran Associates, Inc., pp. 698–707 (cit. on p. 146).

Lütkepohl, Helmut (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media (cit. on p. 162).

Ma, X., N. Trudinger, and X. Wang (2005a). "Regularity of Potential Functions of the Optimal Transportation Problem". In: *Archive for Rational Mechanics and Analysis* (cit. on pp. 20, 101, 260).

Ma, X., N. Trudinger, and X. Wang (2005b). "Regularity of Potential Functions of the Optimal Transportation Problem". In: *Archive for Rational Mechanics and Analysis* (cit. on p. 264).

Ma, Xuezhe, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy (2019). *FlowSeq: Non-Autoregressive Conditional Sequence Generation with Generative Flow*. arXiv: 1909.02480 [cs.CL] (cit. on p. 177).

Madec, G. (2008). *NEMO ocean engine*. Note du Pôle de modélisation, Institut Pierre-Simon Laplace (IPSL), France, No 27, ISSN No 1288-1619 (cit. on p. 50).

Madec, Gurvan, Romain Bourdallé-Badie, Jérôme Chanut, Emanuela Clementi, Andrew Coward, Christian Ethé, Doroteaciro Iovino, Dan Lea, Claire Lévy, Tomas Lovato, Nicolas Martin, Sébastien Masson, Silvia Mocavero, Clément Rousset, Dave Storkey, Martin Vancoppenolle, Simon Müeller, George Nurser, Mike Bell, and Guillaume Samson (Oct. 2019). *NEMO ocean engine*. Version v4.0. Add SI3 and TOP reference manuals (cit. on p. 76).

Mao, Xudong, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley (Oct. 2017). "Least Squares Generative Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2813–2821 (cit. on p. 149).

Mardani, Morteza, Enhao Gong, Joseph Y. Cheng, Shreyas Vasanawala, Greg Zaharchuk, Marcus T. Alley, Neil Thakur, Song Han, William J. Dally, John M. Pauly, and Lei Xing (2017). "Deep Generative Adversarial Networks for Compressed Sensing Automates MRI". In: *CoRR* abs/1706.00051. arXiv: 1706.00051. URL: http://arxiv.org/abs/1706.00051 (cit. on pp. 131, 144).

Marin, Javier, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba (2018). "Recipe1M: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images". In: *arXiv preprint arXiv:1810.06553* (cit. on p. 131).

Matheson, James E and Robert L Winkler (1976). "Scoring rules for continuous probability distributions". In: *Management science* 22.10, pp. 1087–1096 (cit. on p. 173).

Mathieu, Michaël, Camille Couprie, and Yann LeCun (2015). "Deep multi-scale video prediction beyond mean square error". In: *CoRR* abs/1511.05440. URL: http://arxiv.org/abs/1511.05440 (cit. on pp. 52, 54).

Mcculloch, Warren and Walter Pitts (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". In: *Bulletin of Mathematical Biophysics* 5, pp. 127–147 (cit. on p. 20).

Mehta, Viraj, Ian Char, Willie Neiswanger, Youngseog Chung, and Jeff Schneider (2020). "Neural dynamical systems". In: *ICLR 2020 Deep Differential Equations Workshop* (cit. on pp. 16, 63, 69, 71, 231).

Mescheder, Lars, Andreas Geiger, and Sebastian Nowozin (July 2018). "Which Training Methods for GANs do actually Converge?" In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 3481–3490 (cit. on pp. 148, 313).

Mescheder, Lars, Sebastian Nowozin, and Andreas Geiger (2017). "The Numerics of GANs". In: *Advances in Neural Information Processing Systems*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Vol. 30. Curran Associates, Inc., pp. 1825–1835 (cit. on p. 147).

Metz, Luke, Ben Poole, David Pfau, and Jascha Sohl-Dickstein (2017). "Unrolled Generative Adversarial Networks". In: *International Conference on Learning Representations* (cit. on pp. 158, 303).

Miyato, Takeru, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida (2018a). "Spectral Normalization for Generative Adversarial Networks". In: *CoRR* abs/1802.05957. arXiv: 1802.05957. URL: http://arxiv.org/abs/1802.05957 (cit. on p. 90).

Miyato, Takeru, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida (2018b). "Spectral Normalization for Generative Adversarial Networks". In: *CoRR* abs/1802.05957. arXiv: 1802.05957. URL: http://arxiv.org/abs/1802.05957 (cit. on p. 273).

Moriakov, Nikita, Jonas Adler, and Jonas Teuwen (2020). *Kernel of CycleGAN as a Principle homogeneous space*. arXiv: 2001.09061 [cs.LG] (cit. on pp. 94, 96).

Mota, João FC, Nikos Deligiannis, and Miguel RD Rodrigues (2017). "Compressed sensing with prior information: Strategies, geometry, and bounds". In: *IEEE Transactions on Information Theory* 63.7, pp. 4472–4496 (cit. on p. 130).

Mroueh, Youssef and Truyen Nguyen (Apr. 2021). "On the Convergence of Gradient Descent in GANs: MMD GAN As a Gradient Flow". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 1720–1728 (cit. on p. 148).

Mroueh, Youssef, Tom Sercu, and Anant Raj (Apr. 2019). "Sobolev Descent". In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 2976–2985 (cit. on pp. 156, 158).

Muandet, Krikamol, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf (2017). "Kernel Mean Embedding of Distributions: A Review and Beyond". In: *Foundations and Trends® in Machine Learning* 10.1–2, pp. 1–141 (cit. on pp. 155, 159).

Müller, Alfred (1997). "Integral Probability Metrics and Their Generating Classes of Functions". In: *Advances in Applied Probability* 29.2, pp. 429–443 (cit. on p. 149).

Nagabandi, Anusha, Gregory Kahn, Ronald S Fearing, and Sergey Levine (2018). "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7559–7566 (cit. on pp. 16, 62).

Nagarajan, Vaishnavh and J. Zico Kolter (2017). "Gradient descent GAN optimization is locally stable". In: *Advances in Neural Information Processing Systems*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. Vol. 30. Curran Associates, Inc., pp. 5585–5595 (cit. on p. 147).

Nagarajan, Vaishnavh and J. Zico Kolter (2019). "Uniform convergence may be unable to explain generalization in deep learning". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/05e97c207235d63ceb1db43c60db7bbb-Paper.pdf (cit. on p. 84).

Nakkiran, P. et al. (2020). "Deep Double Descent: Where Bigger Models and More Data Hurt". In: *ICLR* (cit. on p. 112).

Neic, Aurel, Fernando O. Campos, Anton J. Prassl, Steven A. Niederer, Martin J. Bishop, Edward J. Vigmond, and Gernot Plank (2017). "Efficient computation of electrograms and ECGs in human whole heart simulations using a reaction-eikonal model". In: *Journal of Computational Physics* 346, pp. 191–211 (cit. on p. 76).

Norcliffe, Alexander, Cristian Bodnar, Ben Day, Jacob Moss, and Pietro Liò (2021). "Neural ODE Processes". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=27acGyyI1BY (cit. on pp. 17, 91).

Novak, R., Y. Bahri, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein (2018). "Sensitivity and Generalization in Neural Networks: an Empirical Study". In: *ICLR* (cit. on p. 112).

Novak, Roman, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A. Alemi, Jascha Sohl-Dickstein, and Samuel S. Schoenholz (2020). "Neural Tangents: Fast and Easy Infinite Neural Networks in Python". In: *International Conference on Learning Representations*. URL: https://github.com/google/neural-tangents (cit. on pp. 148, 158, 302, 303, 315).

Nowozin, Sebastian, Botond Cseke, and Ryota Tomioka (2016). "f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization". In:

Advances in Neural Information Processing Systems. Ed. by Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett. Vol. 29. Curran Associates, Inc., pp. 271–279 (cit. on pp. 147, 149).

Oliva, Junier, Avinava Dubey, Manzil Zaheer, Barnabas Poczos, Ruslan Salakhutdinov, Eric Xing, and Jeff Schneider (Oct. 2018). "Transformation Autoregressive Networks". In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 3898–3907 (cit. on pp. 165, 177).

Oreshkin, Boris N, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio (2019). "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting". In: *arXiv preprint arXiv:1905.10437* (cit. on pp. 61, 176, 233, 234).

Pajot, Arthur, Emmanuel de Bezenac, and Patrick Gallinari (Sept. 2018). "Unsupervised Adversarial Image Reconstruction". In: URL: ICLR%202019%20:%20https://openreview.net/forum?id=BJg4Z3RqF7 (cit. on pp. 7, 130).

Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). "Automatic differentiation in PyTorch". In: (cit. on pp. 14, 29, 32).

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, pp. 8024–8035 (cit. on p. 227).

Patraucean, Viorica, Ankur Handa, and Roberto Cipolla (2015). "Spatio-temporal video autoencoder with differentiable memory". In: *CoRR* abs/1511.06309. URL: http://arxiv.org/abs/1511.06309 (cit. on p. 48).

Patton, Andrew J (2012). "A review of copula models for economic time series". In: *Journal of Multivariate Analysis* 110, pp. 4–18 (cit. on p. 163).

Pearson, John E. (1993). "Complex Patterns in a Simple System". In: *Science* 261.5118, pp. 189–192 (cit. on pp. 84, 85).

Pernot, Pascal and Fabien Cailliez (2017). "A critical review of statistical calibration/prediction models handling data inconsistency and model inadequacy". In: *AIChE Journal* 63.10, pp. 4642–4665 (cit. on pp. 15, 62).

Peyre, Gabriel and Marco Cuturi (Mar. 2018). "Computational Optimal Transport". In: (cit. on pp. 18, 114).

Pinson, Pierre and Julija Tastu (2013). *Discrimination ability of the Energy score*. English. DTU Compute-Technical Report-2013 15. Technical University of Denmark (cit. on p. 173).

Plessix, R-E (2006). "A review of the adjoint-state method for computing the gradient of a functional with geophysical applications". In: *Geophysical Journal International* 167.2, pp. 495–503 (cit. on p. 27).

Psichogios, Dimitris C and Lyle H Ungar (1992). "A hybrid neural network-first principles approach to process modeling". In: *AIChE Journal* 38.10, pp. 1499–1511 (cit. on pp. 16, 62).

Racah, Evan, Christopher Beckham, Tegan Maharaj, Samira Ebrahimi Kahou, Prabhat, and Chris Pal (2017). "ExtremeWeather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, pp. 3402–3413. URL: https://proceedings.neurips.cc/paper/2017/hash/519c84155964659375821f7ca576f095-Abstract.html (cit. on p. 14).

Rahaman, Nasim et al. (2019). "On the Spectral Bias of Neural Networks". In: *ICML* (cit. on p. 112).

Raissi, Maziar (Apr. 2018). "Forward-Backward Stochastic Neural Networks: Deep Learning of High-dimensional Partial Differential Equations". In: *arXiv:1804.07010 [cs, math, stat]*. URL: http://arxiv.org/abs/1804.07010 (cit. on p. 15).

Raissi, Maziar, Paris Perdikaris, and George E. Karniadakis (2019a). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 378, pp. 686–707 (cit. on pp. 76, 91).

Raissi, Maziar, Paris Perdikaris, and George Em Karniadakis (2017). "Machine learning of linear differential equations using Gaussian processes". In: *Journal of Computational Physics* 348, pp. 683–693 (cit. on p. 15).

Raissi, Maziar, Paris Perdikaris, and George Em Karniadakis (2019b). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 473, pp. 686–707 (cit. on pp. 61, 66).

Ramachandran, Prajit, Barret Zoph, and Quoc V. Le (2017). "Searching for Activation Functions". In: *CoRR* abs/1710.05941. arXiv: 1710.05941 (cit. on p. 90).

Ramdas, Aaditya, Sashank Jakkam Reddi, Barnabás Póczos, Aarti Singh, and Larry A. Wasserman (2015). "On the Decreasing Power of Kernel and Distance Based Nonparametric Hypothesis Tests in High Dimensions". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. Ed. by Blai Bonet and Sven Koenig. AAAI Press, pp. 3571–3577. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9727 (cit. on p. 173).

Rangapuram, Syama Sundar, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski (2018). "Deep state space models for time series forecasting". In: *Advances in Neural Information Processing Systems*, pp. 7785–7794 (cit. on pp. 163, 168, 170, 173–177, 323, 324).

Rasul, Kashif, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf (2020). "Multi-variate Probabilistic Time Series Forecasting via Conditioned Normalizing Flows". In: *arXiv preprint arXiv:2002.06103*. arXiv: 2002.06103 [cs.LG] (cit. on p. 177).

Reichstein, Markus, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and Prabhat (2019). "Deep learning and process understanding for data-driven Earth system science". In: *Nature* 566, pp. 195–204 (cit. on pp. 1, 15, 61, 76).

Rezende, Danilo Jimenez and Shakir Mohamed (2015). "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, pp. 1530–1538 (cit. on pp. 163, 324).

Rico-Martinez, R, JS Anderson, and IG Kevrekidis (1994). "Continuous-time nonlinear signal processing: a neural network based approach for gray box identification". In: *Proceedings of IEEE Workshop on Neural Networks for Signal Processing*. IEEE, pp. 596–605 (cit. on pp. 16, 62).

Rolnick, David, 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. (2019). "Tackling climate change with machine learning". In: *NeurIPS 2019 workshop on Climate Change with Machine Learning* (cit. on p. 60).

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597. URL: http://arxiv.org/abs/1505.04597 (cit. on p. 38).

Rosenblatt, F. (1958). "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". In: *Psychological Review*, pp. 65–386 (cit. on p. 20).

Rudy, Samuel H., Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz (Apr. 2017). "Data-driven discovery of partial differential equations". In: *Science Advances* 3.4, e1602614 (cit. on p. 15).

Ruthotto, Lars and Eldad Haber (2020). "Deep Neural Networks Motivated by Partial Differential Equations". In: *J Math Imaging Vis* (cit. on pp. 21, 127).

Saha, Priyabrata, Saurabh Dash, and Saibal Mukhopadhyay (2020). "PhICNet: Physics-Incorporated Convolutional Recurrent Neural Networks for Modeling Dynamical Systems". In: *arXiv preprint arXiv:2004.06243* (cit. on pp. 16, 63).

Salinas, David, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus (2019a). "High-dimensional multivariate forecasting with low-rank Gaussian Copula Processes". In: *Advances in Neural Information Processing*

Systems 32, pp. 6824–6834 (cit. on pp. 162–164, 169, 172–174, 176, 324, 326, 328, 330).

Salinas, David, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski (2019b). "DeepAR: Probabilistic forecasting with autoregressive recurrent networks". In: *International Journal of Forecasting* (cit. on pp. 163, 164, 168, 169, 173, 176).

Sandler, M. et al. (2019). "Non-Discriminative Data or Weak Model? On the Relative Importance of Data and Model Resolution". In: *ICCVW* (cit. on p. 116).

Santambrogio, Filippo (2015). *Optimal transport for Applied Mathematicians: Calculus of Variations, PDEs and Modeling* (cit. on pp. 17, 19, 98–100, 112, 113, 115, 124, 258, 259, 261, 263).

Saxe, Andrew M., James L. Mcclelland, and Surya Ganguli (2014). "Exact solutions to the nonlinear dynamics of learning in deep linear neural network". In: *ICLR* (cit. on p. 125).

Scheuerer, Michael (Oct. 2009). "A Comparison of Models and Methods for Spatial Interpolation in Statistics and Numerical Analysis". PhD thesis. Georg-August-Universität Göttingen. URL: https://ediss.uni-goettingen.de/handle/11858/00-1735-0000-0006-B3D5-1 (cit. on p. 289).

Sen, Rajat, Hsiang-Fu Yu, and Inderjit S Dhillon (2019). "Think Globally, Act Locally: A Deep Neural Network Approach to High-Dimensional Time Series Forecasting". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., pp. 4838–4847 (cit. on p. 176).

Seo, Sungyong, Chuizheng Meng, and Yan Liu (2020). "Physics-aware Difference Graph Networks for Sparsely-Observed Dynamics". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 61).

Shalev-Shwartz, Shai and Shai Ben-David (2014). "Covering Numbers". In: *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, pp. 337–340 (cit. on p. 238).

Shi, Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo (2015a). "Convolutional LSTM network: A machine learning approach for precipitation nowcasting". In: *Advances in neural information processing systems (NeurIPS)*, pp. 802–810 (cit. on p. 61).

Shi, Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo (2015b). "Convolutional LSTM network: A machine learning approach for precipitation nowcasting". In: *Advances in Neural Information Processing Systems 28*, pp. 802–810 (cit. on pp. 14, 52, 54).

Shumway, R. H. and D. S. Stoffer (1982). "An Approach To Time Series Smoothing And Forecasting Using The EM Algorithm". In: *Journal of Time Series Analysis* 3.4, pp. 253–264 (cit. on pp. 169, 171).

Silva, Ashton de, Rob J Hyndman, and Ralph Snyder (2010). "The vector innovations structural time series framework: a simple approach to multivariate forecasting". In: *Statistical Modelling* 10.4, pp. 353–374 (cit. on p. 162).

Sirignano, Justin and Konstantinos Spiliopoulos (2018). "DGM: A deep learning algorithm for solving partial differential equations". In: *Journal of computational physics* 375, pp. 1339–1364 (cit. on pp. 61, 69).

Sirkes, Ziv and Eli Tziperman (1997). "Finite difference of adjoint or adjoint of finite difference?" In: *Monthly weather review* 125.12, pp. 3373–3378 (cit. on pp. 14, 29).

Smyl, Slawek (2020). "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting". In: *International Journal of Forecasting* 36.1, pp. 75–85 (cit. on p. 176).

Sohl-Dickstein, Jascha, Roman Novak, Samuel S. Schoenholz, and Jaehoon Lee (2020). "On the infinite width limit of neural networks with a standard parameterization". In: *arXiv preprint arXiv:2001.07301* (cit. on p. 148).

Sønderby, Casper Kaae, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár (Oct. 2016). "Amortised MAP Inference for Image Super-resolution". In: *arXiv:1610.04490 [cs, stat]*. URL: http://arxiv.org/abs/1610.04490 (cit. on p. 144).

Sonoda, Sho and Noboru Murata (2019). "Transport Analysis of Infinitely Deep Neural Network". In: *JMLR* (cit. on pp. 22, 127).

Soudry, Daniel, Elad Hoffer, Mor Shpigel Nacson, and Nathan Srebro (2018). "The Implicit Bias of Gradient Descent on Separable Data". In: (cit. on pp. 110, 180).

Spieckermann, Sigurd, Siegmund Düll, Steffen Udluft, Alexander Hentschel, and Thomas Runkler (2015). "Exploiting similarity in system identification tasks with recurrent neural networks". In: *Neurocomputing* 169. Learning for Visual Semantic Understanding in Big Data ESANN 2014 Industrial Data Processing and Analysis, pp. 343–349 (cit. on pp. 17, 86, 87, 91).

Sriperumbudur, Bharath K., Kenji Fukumizu, and Gert R. G. Lanckriet (2011). "Universality, Characteristic Kernels and RKHS Embedding of Measures". In: *Journal of Machine Learning Research* 12.70, pp. 2389–2410 (cit. on p. 306).

Sriperumbudur, Bharath K., Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert R. G. Lanckriet (2010). "Hilbert Space Embeddings and Metrics on Probability Measures". In: *Journal of Machine Learning Research* 11.50, pp. 1517–1561 (cit. on p. 152).

Steinwart, Ingo (Nov. 2001). "On the Influence of the Kernel on the Consistency of Support Vector Machines". In: *Journal of Machine Learning Research* 2, pp. 67–93 (cit. on p. 306).

Stuart, A. M. (2010). "Inverse problems: A Bayesian perspective". In: *Acta Numerica* 19, pp. 451–559 (cit. on p. 130).

Sun, Ruoyu, Tiantian Fang, and Alexander Schwing (2020). "Towards a Better Global Loss Landscape of GANs". In: *Advances in Neural Information Processing Systems*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria

Florina Balcan, and Hsuan-Tien Lin. Vol. 33. Curran Associates, Inc., pp. 10186–10198 (cit. on p. 147).

Székely, Gábor J (2003). "E-Statistics: The energy of statistical samples". In: *Bowling Green State University, Department of Mathematics and Statistics Technical Report* 3.05, pp. 1–18 (cit. on p. 330).

Tancik, Matthew, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng (2020). "Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains". In: *Advances in Neural Information Processing Systems*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria Florina Balcan, and Hsuan-Tien Lin. Vol. 33. Curran Associates, Inc., pp. 7537–7547 (cit. on p. 148).

Teney, Damien, Ehsan Abbasnejad, and Anton van den Hengel (2020). "Unshuffling Data for Improved Generalization". In: arXiv: 2002.11894. URL: http://arxiv.org/abs/2002.11894 (cit. on pp. 16, 90).

Thompson, Michael L and Mark A Kramer (1994). "Modeling chemical processes using prior knowledge and neural networks". In: *AIChE Journal* 40.8, pp. 1328–1340 (cit. on pp. 16, 62).

Tolosana, Ruben, Ruben Vera-Rodriguez, Julian Fierrez, Aythami Morales, and Javier Ortega-Garcia (2020). "Deepfakes and beyond: A Survey of face manipulation and fake detection". In: *Information Fusion* 64, pp. 131–148 (cit. on p. 161).

Toth, Peter, Danilo Jimenez Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins (2020). "Hamiltonian generative networks". In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 69, 228, 234).

Toubeau, Jean-François, Jérémie Bottieau, François Vallée, and Zacharie De Grève (2018). "Deep learning-based multivariate probabilistic forecasting for short-term scheduling in power markets". In: *IEEE Transactions on Power Systems* 34.2, pp. 1203–1215 (cit. on p. 60).

Trémolet, Yannick (2006). "Accounting for an imperfect model in 4D-Var". In: *Quarterly Journal of the Royal Meteorological Society* 132.621, pp. 2483–2504 (cit. on p. 52).

Tripathi, Subarna, Zachary C. Lipton, and Truong Q. Nguyen (2018). "Correction by Projection: Denoising Images with Generative Adversarial Networks". In: *CoRR* abs/1803.04477. arXiv: 1803.04477. URL: http://arxiv.org/abs/1803.04477 (cit. on p. 144).

Tripathy, Soumya, Juho Kannala, and Esa Rahtu (2018). "Learning image-to-image translation using paired and unpaired training samples". In: (cit. on pp. 21, 109).

Turkmen, Ali Caner, Yuyang Wang, and Tim Januschowski (2019). "Intermittent Demand Forecasting with Deep Renewal Processes". In: *arXiv preprint arXiv:1911.10416*. arXiv: 1911.10416 [cs.LG] (cit. on p. 176).

Ulyanov, Dmitry, Andrea Vedaldi, and Victor S. Lempitsky (2017). "Deep Image Prior". In: *CoRR* abs/1711.10925. arXiv: 1711.10925. URL: http://arxiv.org/abs/1711.10925 (cit. on pp. 140, 144, 276).

Ummenhofer, Benjamin, Lukas Prantl, Nils Thuerey, and Vladlen Koltun (2020). "Lagrangian Fluid Simulation with Continuous Convolutions". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 61).

Vallis, Geoffrey K. (2017). *Atmospheric and Oceanic Fluid Dynamics: Fundamentals and Large-Scale Circulation*. 2nd ed. Cambridge University Press (cit. on p. 52).

Van der Weide, Roy (2002). "GO-GARCH: a multivariate generalized orthogonal GARCH model". In: *Journal of Applied Econometrics* 17.5, pp. 549–564 (cit. on p. 173).

Van Veen, David, Ajil Jalal, Eric Price, Sriram Vishwanath, and Alexandros G. Dimakis (June 2018). "Compressed Sensing with Deep Image Prior and Learned Regularization". In: *arXiv:1806.06438 [cs, math, stat]*. URL: http://arxiv.org/abs/1806.06438 (cit. on p. 144).

Vandal, Thomas, Evan Kodra, Sangram Ganguly, Andrew Michaelis, Ramakrishna Nemani, and Auroop R Ganguly (July 2018). "Generating High Resolution Climate Change Projections through Single Image Super-Resolution: An Abridged Version". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, pp. 5389–5393. URL: https://doi.org/10.24963/ijcai.2018/759 (cit. on p. 15).

Villani, C (Jan. 2008). "Optimal transport – Old and new". In: vol. 338, pp. xxii+973 (cit. on p. 261).

Wang, Qi, Feng Li, Yi Tang, and Yan Xu (2019). "Integrating model-driven and data-driven methods for power system frequency stability assessment and control". In: *IEEE Transactions on Power Systems* 34.6, pp. 4557–4568 (cit. on pp. 16, 63, 69, 71).

Wang, Ting-Chun, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro (2018). "Video-to-Video Synthesis". In: *CoRR* abs/1808.06601. arXiv: 1808.06601. URL: http://arxiv.org/abs/1808.06601 (cit. on p. 93).

Wang, Yunbo, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S. Yu (Apr. 2018a). "PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning". In: *arXiv:1804.06300 [cs, stat]*. URL: http://arxiv.org/abs/1804.06300 (cit. on pp. 33, 35).

Wang, Yunbo, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S. Yu (2018b). "PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 61, 69).

Wang, Yunbo, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu (2017). "PredRNN: Recurrent Neural Networks for Predictive Learning using

Spatiotemporal LSTMs". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/e5f6ad6ce374177eef023bf5d0c018b6-Paper.pdf (cit. on pp. 87, 90).

Wang, Yuyang, Alex Smola, Danielle Maddix, Jan Gasthaus, Dean Foster, and Tim Januschowski (2019). "Deep factors for forecasting". In: *International Conference on Machine Learning*, pp. 6607–6617 (cit. on p. 176).

Wang, Zhengwei, Qi She, and Tomás E. Ward (Feb. 2021). "Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy". In: *ACM Comput. Surv.* 54.2 (cit. on p. 146).

Weinan, E (2017). "A Proposal on Machine Learning via Dynamical Systems". In: (cit. on pp. 103, 112).

Willard, Jared D., Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar (2020). "Integrating physics-based modeling with machine learning: A survey". In: vol. 1, pp. 1–34 (cit. on p. 76).

Xie, Saining et al. (2017). "Aggregated Residual Transformations for Deep Neural Networks". In: *CVPR* (cit. on pp. 115, 116, 126, 268).

Yan, Hanshu, Jiawei Du, Vincent Tan, and Jiashi Feng (2020). "On Robustness of Neural Ordinary Differential Equations". In: *ICLR* (cit. on pp. 21, 127).

Yang, Chao, Taehwan Kim, Ruizhe Wang, Hao Peng, and C.-C. Jay Kuo (2018). "ESTHER: Extremely Simple Image Translation Through Self-Regularization". In: (cit. on p. 94).

Yang, Greg (2020). "Tensor programs II: Neural tangent kernel for any architecture". In: *arXiv preprint arXiv:2006.14548* (cit. on pp. 148, 302).

Yang, Greg and Edward J Hu (2020). "Feature Learning in Infinite-Width Neural Networks". In: *arXiv preprint arXiv:2011.14522* (cit. on pp. 148, 161, 302).

Yang, Greg and Hadi Salman (2019). "A fine-grained spectral perspective on neural networks". In: *arXiv preprint arXiv:1907.10599* (cit. on pp. 148, 305, 313).

Yin, Yuan, Ibrahim Ayed, Emmanuel de Bézenac, Nicolas Baskiotis, and Patrick Gallinari (2021a). *LEADS: Learning Dynamical Systems that Generalize Across Environments*. arXiv: 2106.04546 [cs.LG] (cit. on pp. 5, 75).

Yin, Yuan, Vincent LE GUEN, Jérémie DONA, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas THOME, and patrick gallinari (2021b). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=kmG8vRXTFv (cit. on pp. 5, 60).

Yin, Yuan, Vincent Le Guen, Jérémie Dona, Emmanuel de Bezenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari (2021c). "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting". In: *International Con-*

ference on Learning Representations. URL: https://openreview.net/forum?id=kmG8vRXTFv (cit. on pp. 76, 89, 91).

Yıldız, Çağatay, Markus Heinonen, and Harri Lähdesmäki (2019). *ODE^{2}VAE: Deep generative second order ODEs with Bayesian neural networks*. arXiv: 1905.10994 [stat.ML] (cit. on pp. 17, 91).

Yoshida, Yuichi and Takeru Miyato (2017). "Spectral Norm Regularization for Improving the Generalizability of Deep Learning". In: *arXiv* (cit. on pp. 22, 127).

Yu, Fisher, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao (2015). "LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop". In: *CoRR* abs/1506.03365. arXiv: 1506.03365. URL: http://arxiv.org/abs/1506.03365 (cit. on p. 131).

Yuan, Yuan, Siyuan Liu, Jiawei Zhang, Yongbing Zhang, Chao Dong, and Liang Lin (2018). "Unsupervised Image Super-Resolution Using Cycle-in-Cycle Generative Adversarial Networks". In: (cit. on p. 93).

Zagoruyko, Sergey and Nikos Komodakis (2016). "Wide Residual Networks". In: *BMVC* (cit. on pp. 115, 116).

Zarchan, Paul and Howard Musoff (2015). *Fundamentals of Kalman Filtering: A Practical Approach, Fourth Edition*. Ed. by Howard Musoff and Paul Zarchan. Reston, VA: American Institute of Aeronautics and Astronautics, Inc. (cit. on pp. 166, 176).

Zhang, C. et al. (2017). "Understanding deep learning requires rethinking generalization". In: *ICLR* (cit. on pp. 112, 116).

Zhang, Chiyuan, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals (2017). "Understanding deep learning requires rethinking generalization". In: (cit. on p. 96).

Zhang, Han, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena (2018). "Self-Attention Generative Adversarial Networks". In: (cit. on pp. 138, 273).

Zhang, Han, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris N. Metaxas (2016). "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks". In: (cit. on p. 93).

Zhang, Jingfeng et al. (2019). "Towards Robust ResNet: A Small Step but a Giant Leap". In: *IJCAI* (cit. on pp. 21, 127).

Zhang, Yaoyu, Zhi-Qin John Xu, Tao Luo, and Zheng Ma (July 2020). "A type of generalization error induced by initialization in deep neural networks". In: *Proceedings of The First Mathematical and Scientific Machine Learning Conference*. Ed. by Jianfeng Lu and Rachel Ward. Vol. 107. Proceedings of Machine Learning Research. Princeton University, Princeton, NJ, USA: PMLR, pp. 144–164 (cit. on pp. 155, 159, 305, 310, 316).

Zhang, Yu and Qiang Yang (2017). "A Survey on Multi-Task Learning". In: *CoRR* abs/1707.08114. arXiv: 1707.08114. URL: http://arxiv.org/abs/1707.08114 (cit. on pp. 17, 90).

Zhou, Zhiming, Jiadong Liang, Yuxuan Song, Lantao Yu, Hongwei Wang, Weinan Zhang, Yong Yu, and Zhihua Zhang (June 2019). "Lipschitz Generative Adversarial Nets". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 7584–7593 (cit. on pp. 147, 148).

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros (Mar. 2017). "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks". In: *arXiv:1703.10593 [cs]*. arXiv: 1703.10593. URL: http://arxiv.org/abs/1703.10593 (visited on 04/27/2018) (cit. on p. 144).

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros (2017). "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks". In: *CoRR* abs/1703.10593. arXiv: 1703.10593. URL: http://arxiv.org/abs/1703.10593 (cit. on pp. 6, 93, 96).

Ziegler, Zachary and Alexander Rush (Sept. 2019). "Latent Normalizing Flows for Discrete Sequences". In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 7673–7682 (cit. on p. 177).

APPENDIX CHAPTER 3

Contents

A.1 Appendix 3.1: Learning Dynamical Systems from Partial Observations **215**
A.1.1 Dataset of 3.1.3.1 215
A.1.2 Proof of Theorem Theorem S2 217
A.1.3 Proof of Theorem S1 220
A.2 Appendix 3.3: Incorporating Imperfect Knowledge **223**
A.2.1 A reminder on Chebyshev sets 223
A.2.2 Proof of Proposition S2 223
A.2.3 Parameter estimation in incomplete physical models 224
A.2.4 A discussion of supervision over derivatives 226
A.2.5 Implementation details 227
A.2.6 Ablation study 231
A.2.7 Additional experiments 233
A.3 Appendix 3.4: Leveraging the Data from Different Environments **237**
A.3.1 Proof of Proposition S4 237
A.3.2 Further details on the generalization with *LEADS* 238
A.3.3 Optimizing Ω in practice 248
A.3.4 Additional experimental details 249

A.1 Appendix 3.1: Learning Dynamical Systems from Partial Observations

A.1.1 Dataset of 3.1.3.1

The Shallow Water Equations

The shallow-water model can be written as:

$$\begin{aligned}
\frac{\partial u}{\partial t} &= +(f+\zeta).v - \partial_x(\frac{u^2+v^2}{2} + g^*.h) + \\
&\quad \frac{\tau_x}{\rho_0(H+h)} - \gamma.u + \nu\Delta u \\
\frac{\partial v}{\partial t} &= -(f+\zeta).u - \partial_y(\frac{u^2+v^2}{2} + g^*.h) + \\
&\quad \frac{\tau_y}{\rho_0(H+h)} - \gamma.v + \nu\Delta v \\
\frac{\partial h}{\partial t} &= -\partial_x(u(H+h)) - \partial_y(v(H+h))
\end{aligned} \tag{S1}$$

where:

- u, v, h are state variables, standing for velocity and mixed layer depth anomaly)
- ζ is the vorticity.
- $g^* = 0.02$ is the reduced gravity
- $H = 500m$ is the mean mixed-layer depth.
- ρ_0 is the density of the water set to $1000mg/m^3$
- γ is the dissipation coefficient set to $2 \cdot 10^{-7}s^{-1}$
- ν is the diffusion coefficient set to $0.72m^2/s$
- τ_x is the zonal wind forcing defined in Eq. A.1.1

The zonal wind forcing is defined as:

$$\tau_x(y) = \tau_0 \sin(2\pi(y - y_c)/L_y$$

where:

– τ_0 is the maximum intensity of the wind stress(in the standard case $0.15m.s^{-2}$).

- y is the latitude coordinate
- y_c is the center y coordinate of the domain
- L_y is the length of the domain ($L_y = 1600km$ in our case).

Here, the state is composed of the velocity vector and the mixed layer depth:

$$X = \begin{pmatrix} u \\ v \\ h \end{pmatrix} \text{ and } \mathcal{H}(X) = h$$

For our simulations, the spatial differential operators have been discretized using finite differences on a Arakawa C-grid.

The Navier-Stokes Equations

$$\begin{aligned} \frac{\partial u}{\partial t} + (u \cdot \nabla)u &= -\frac{\nabla p}{\rho} + g + \nu \nabla^2 u \\ \frac{\partial \rho}{\partial t} + (u \cdot \nabla)\rho &= 0 \\ \nabla \cdot u &= 0 \end{aligned} \tag{S2}$$

where $\nabla\cdot$ is the divergence operator, u corresponds to the flow velocity vector, p to the pressure, and ρ to the density.

$$a = \nabla b + c$$

and

$$\nabla \cdot c = 0$$

Moreover, this pair is unique up to an additive constant for b. Thus, we can define a linear operator $\mathbb{P}$ by :

$$\mathbb{P}(a) = c$$

This operator is a continuous linear projector which is the identity for divergence-free vector fields and vanishes for those deriving from a potential.

By taking a solution of eq. S2 and applying $\mathbb{P}$ on the first equation, we have, as u is divergence free from the third equation and as g derives from a potential :

$$\frac{\partial u}{\partial t} = -\mathbb{P}[(u \cdot \nabla)u] + \nu \mathbb{P}(\nabla^2 u)$$

where permuting derivation and $\mathbb{P}$ is justified by the continuity of the operator [1].

1. One can use a finite difference approximation to show it for example.

Thus, if u is solution to eq. S2, it is also a solution of :

$$\frac{\partial u}{\partial t} = -\mathbb{P}[(u \cdot \nabla)u] + \nu\mathbb{P}(\nabla^2 u)$$
$$\frac{\partial \rho}{\partial t} = -(u \cdot \nabla)\rho$$

which is of the form of eq. S1.

Conversely, the solution of the above system is such that :

$$u_t = \int \frac{\partial u}{\partial t} = \int -\mathbb{P}[(u \cdot \nabla)u] + \nu\mathbb{P}(\nabla^2 u)$$

which gives, by exchanging $\mathbb{P}$ and the integral [2] :

$$u_t = \mathbb{P}\left[\int -(u \cdot \nabla)u + \nu\nabla^2 u\right]$$

so that u is automatically of null divergence by definition of $\mathbb{P}$. The two systems are thus equivalent.

In conclusion, we have:

$$X = \begin{pmatrix} u \\ \rho \end{pmatrix}, \text{ and } \mathcal{H}(X) = \rho$$

Moreover, u is generally a two or three-dimensional spatial field while ρ is a scalar field.

A.1.2 Proof of Theorem Theorem S2

In the following, bold $\mathbf{x}$ and $\mathbf{y}$ will denote vectors of $\mathbb{R}^2$, while x and y will correspond to the first and second components of $\mathbf{x}$, respectively. Analogously, u and v will correspond to the components of w. The 2D Fourier Transformation $\mathcal{F}$ of $f : \mathbb{R}^2 \to \mathbb{R}$ is defined as

$$\begin{aligned} \mathcal{F}(f) &= \int_{\mathbb{R}^2} f(\mathbf{x}) e^{-i<\xi,\mathbf{x}>} d\mathbf{x} \\ &= \int_{\mathbb{R}} \int_{\mathbb{R}} f(x,y) e^{-ix\xi_1 - iy\xi_2} dxdy \end{aligned} \tag{S3}$$

2. To prove this, we can take a sum approximation to the integral and use again the linearity then the continuity of $\mathbb{P}$.

We apply the Fourier Transform $\mathcal{F}$ to both sides of Equation S14. As consequence of the linearity of the Fourier transform, we can calculate decompose the Fourier transform of the left hand side in the sum of the transforms of each term. We have three terms: $\frac{\partial I}{\partial t}$, $(w.\nabla)I$ and $-D\nabla^2 I$.

$$\begin{aligned}\mathcal{F}(\frac{\partial I}{\partial t}) &= \int_{\mathbb{R}^2} \frac{\partial I}{\partial t} e^{-i<\mathbf{x},\xi>} d\mathbf{x} \\ &= \int_{\mathbb{R}^2} \frac{\partial}{\partial t}(I e^{-i<\mathbf{x},\xi>}) d\mathbf{x} \\ &= \frac{\partial}{\partial t} \int_{\mathbb{R}^2} I e^{-i<\mathbf{x},\xi>} d\mathbf{x} \\ &= \frac{\partial \mathcal{F}(I)}{\partial t}\end{aligned} \tag{S4}$$

$$\begin{aligned}\mathcal{F}((w.\nabla)I) &= \int_{\mathbb{R}^2} (w.\nabla) I e^{-i<\mathbf{x},\xi>} d\mathbf{x} \\ &= \int_{\mathbb{R}} \int_{\mathbb{R}} (u\frac{\partial I}{\partial x} + v\frac{\partial I}{\partial y}) e^{-ix\xi_1 - iy\xi_2} dxdy \\ &= u \int_{\mathbb{R}} e^{-iy\xi_2} \int_{\mathbb{R}} \frac{\partial I}{\partial x} e^{-ix\xi_1} dxdy + v \int_{\mathbb{R}} e^{-ix\xi_1} \int_{\mathbb{R}} \frac{\partial I}{\partial y} e^{-iy\xi_2} dydx \\ &= i\xi_1 u \int_{\mathbb{R}} e^{-iy\xi_2} \int_{\mathbb{R}} I e^{-ix\xi_1} dxdy + i\xi_2 v \int_{\mathbb{R}} e^{-ix\xi_1} \int_{\mathbb{R}} \frac{\partial I}{\partial y} e^{-iy\xi_2} dydx \\ &= i\xi_1 u \int_{\mathbb{R}} \int_{\mathbb{R}} I e^{-ix\xi_1 - iy\xi_2} dxdy + i\xi_2 v \int_{\mathbb{R}} \int_{\mathbb{R}} I e^{-ix\xi_1 - iy\xi_2} dxdy \\ &= (i\xi_1 u + i\xi_2 v) \int_{\mathbb{R}} \int_{\mathbb{R}} I e^{-ix\xi_1 - iy\xi_2} dxdy \\ &= i < \xi, w > \mathcal{F}(I)\end{aligned} \tag{S5}$$

$$
\begin{aligned}
\mathcal{F}(-D\nabla^2 I) &= -\int_{\mathbb{R}^2} D\nabla^2 I e^{-i<\mathbf{x},\xi>} d\mathbf{x} \\
&= -\int_{\mathbb{R}}\int_{\mathbb{R}} D(\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}) e^{-ix\xi_1 - iy\xi_2} dxdy \\
&= -D\int_{\mathbb{R}} e^{-iy\xi_2} \int_{\mathbb{R}} \frac{\partial^2 I}{\partial x^2} e^{-ix\xi_1} dxdy - D\int_{\mathbb{R}} e^{-ix\xi_1} \int_{\mathbb{R}} \frac{\partial^2 I}{\partial y^2} e^{-iy\xi_2} dydx \\
&= -(i\xi_1)^2 D\int_{\mathbb{R}} e^{-iy\xi_2} \int_{\mathbb{R}} I e^{-ix\xi_1} dxdy - (i\xi_2)^2 D\int_{\mathbb{R}} e^{-ix\xi_1} \int_{\mathbb{R}} I e^{-iy\xi_2} dydx \\
&= D\xi_1^2 \int_{\mathbb{R}} e^{-iy\xi_2} \int_{\mathbb{R}} I e^{-ix\xi_1} dxdy + D\xi_2^2 \int_{\mathbb{R}} e^{-ix\xi_1} \int_{\mathbb{R}} I e^{-iy\xi_2} dydx \\
&= D\xi_1^2 \int_{\mathbb{R}}\int_{\mathbb{R}} I e^{-ix\xi_1 - iy\xi_2} dxdy + D\xi_2^2 \int_{\mathbb{R}}\int_{\mathbb{R}} I e^{-ix\xi_1 - iy\xi_2} dxdy \\
&= D\left\|\xi\right\|^2 \mathcal{F}(I)
\end{aligned}
\tag{S6}
$$

Regrouping all three previously calculated terms, we obtain

$$
\frac{\partial \mathcal{F}(I)}{\partial t} + (i < \xi, w > + D\left\|\xi\right\|^2)\mathcal{F}(I) = 0 \tag{S7}
$$

This is a first order ordinary differential equation of the form $f'(t) + af(t) = 0$, which admits a known solution $f(t) = f(0)e^{-at}$. Thus, the solution of Equation S7 is

$$
\begin{aligned}
\mathcal{F}(I) &= \mathcal{F}(I)_0\, e^{-(i<\xi,w> + D\|\xi\|^2)t} \\
&= \mathcal{F}(I)_0\, e^{-i<\xi,w>t} e^{-Dt\|\xi\|^2}
\end{aligned}
\tag{S8}
$$

where $\mathcal{F}(I)_0$ denotes the initial condition of the advection diffusion equation in the frequency domain. In order to obtain a solution of Equation S14 in the spatial domain, we calculate the inverse Fourier Transform $\mathcal{F}^{-1}$ of Equation S8. The multiplication of two functions in the frequency domain is equivalent to their convolution in the spatial domain, *i.e.* $\mathcal{F}(f * g) = \mathcal{F}(f)\mathcal{F}(g)$. Hence, the inverse of both terms $\mathcal{F}(I)_0\, e^{-i<\xi,w>t}$ and $e^{-Dt\|\xi\|^2}$ can be calculated separately:

Multiplication by a complex exponential in the frequency domain is equivalent to a shift in the spatial domain : $e^{-i<\xi,w>}\mathcal{F}(f(\mathbf{x})) = \mathcal{F}(f(\mathbf{x} - w))$, for $v \in \mathbb{R}^2$. Thus, for the first term,

$$
\mathcal{F}^{-1}(\mathcal{F}(I)_0\, e^{-(i<\xi,w>)t}) = I_0(\mathbf{x} - w) \tag{S9}
$$

For the second term, we use the fact that the Fourier Transform of a Gaussian function also is a Gaussian function, *i.e.* $\mathcal{F}(\frac{1}{2\pi\sigma^2}e^{-\frac{1}{2\sigma^2}\|\mathbf{x}\|^2}) = e^{-\frac{1}{2}\sigma^2\|\xi\|^2}$. Identifying σ^2 with $2Dt$, we have:

$$\mathcal{F}^{-1}(e^{-Dt\|\xi\|^2}) = \frac{1}{4\pi Dt}e^{-\frac{1}{4Dt}\|\mathbf{x}\|^2} \tag{S10}$$

As has been stated above, the solution is a convolution of both previously calculated terms:

$$\begin{aligned} I(\mathbf{x},t) &= \int_{\mathbb{R}^2} \frac{1}{4\pi Dt}e^{-\frac{1}{4Dt}\|\mathbf{y}\|^2} I_0(\mathbf{x}-w-\mathbf{y})dy \\ &= \int_{\mathbb{R}^2} \frac{1}{4\pi Dt}e^{-\frac{1}{4Dt}\|\mathbf{x}-w-\mathbf{y}\|^2} I_0(\mathbf{y})d\mathbf{y} \end{aligned} \tag{S11}$$

which concludes the proof. □

A.1.3 Proof of Theorem S1

In order to use gradient descent, we must first calculate the gradient of the cost functional under the constraints, *i.e.* the differential of $\theta \to \mathbb{E}_Y \mathcal{J}(Y, \mathcal{H}(X^\theta))$. However, this implies calculating $\dfrac{\partial X^\theta}{\partial \theta}$, which is often very computationally demanding, as it implies solving $\dim(\theta)$ forward equations. The adjoint state method avoids those costly computations by considering the Lagrangian formulation of the constrained optimization problem introduced in eq. S6, the Lagrangian being defined as:

$$\begin{aligned} \mathcal{L}(X,\lambda,\mu,\theta) = \mathcal{J}(X) + \int_0^T \left\langle \lambda_t, \frac{dX_t}{dt} - F_\theta(X_t) \right\rangle \mathrm{dt} \\ + \langle \mu, X_0 - g_\theta \rangle \end{aligned} \tag{S12}$$

here, the scalar product $\langle \cdot, \cdot \rangle$ is the scalar product associated to the L^2 space over Ω.

As, for any θ, X^θ satisfies the constraints by definition, we can now write:

$$\forall \theta, \lambda, \mu, \ \mathcal{L}(X^\theta, \lambda, \mu, \theta) = \mathcal{J}(Y, \mathcal{H}(X^\theta))$$

which gives :

$$\forall \lambda, \mu, \ \frac{\partial}{\partial \theta}\mathcal{L}(X^\theta, \lambda, \mu, \theta) = \frac{\partial}{\partial \theta}\mathcal{J}(X^\theta)$$

$$F(u_0 + \delta u) = F(u_0) + \partial_u F(u_0)\delta u + o(\delta u)$$

By hypothesis, we consider this operator to be always continuous in our case.

Straightforward calculus gives us:

$$\frac{\partial \mathcal{J}(X_t^\theta)}{\partial \theta} = \int_0^T 2 \left\langle \partial_X \mathcal{H}(X_t^\theta) \cdot \partial_\theta X_t^\theta, \mathcal{H}(X_t^\theta) - Y_t \right\rangle \mathrm{dt}$$

Let us fix θ and a variation $\delta\theta$. Then, we have, by definition:

$$X^{\theta+\delta\theta} = X_t^\theta + \partial_\theta X_t^\theta \cdot \delta\theta + o(\delta\theta)$$

and, for any X and any δX:

$$F_\theta(X + \delta X) = F_\theta(X) + \partial_X F_\theta(X) \cdot \delta X + o(\delta X)$$

and:

$$F_{\theta+\delta\theta}(X) = F_\theta(X) + \partial_\theta F_\theta(X) \cdot \delta\theta + o(\delta\theta)$$

so that:

$$F_{\theta+\delta\theta}(X_t^{\theta+\delta\theta}) = F_\theta(X_t^{\theta+\delta\theta}) + \partial_\theta F_\theta(X_t^{\theta+\delta\theta}) \cdot \delta\theta + o(\delta\theta)$$

Then, because F is twice continuously differentiable:

$$\begin{aligned}\partial_\theta F_\theta(X_t^{\theta+\delta\theta}) &= \partial_\theta F_\theta \left(X_t^\theta + \partial_\theta X_t^\theta \cdot \delta\theta + o(\delta\theta)\right) \\ &= \partial_\theta F_\theta(X_t^\theta) + \partial_X \partial_\theta F_\theta(X_t^\theta) \cdot \partial_\theta X_t^\theta \cdot \delta\theta \\ &\quad + o(\delta\theta)\end{aligned}$$

and:

$$\begin{aligned}F_\theta(X_t^{\theta+\delta\theta}) &= F_\theta \left(X_t^\theta + \partial_\theta X_t^\theta \cdot \delta\theta + o(\delta\theta)\right) \\ &= F_\theta(X_t^\theta) + \partial_X F_\theta(X_t^\theta) \cdot \partial_\theta X_t^\theta \cdot \delta\theta + o(\delta\theta)\end{aligned}$$

Moreover, as all differential operators below are continuous by hypothesis, we have that:

$$\|(\partial_X \partial_\theta F_\theta(X_t^\theta) \cdot \partial_\theta X_t^\theta \cdot \delta\theta) \cdot \delta\theta\| \leq \|\partial_X \partial_\theta F_\theta(X_t^\theta)\| \, \|\partial_\theta X_t^\theta\| \, \|\delta\theta\|^2$$

so that:

$$\begin{aligned}&F_{\theta+\delta\theta}(X_t^{\theta+\delta\theta}) \\ &\quad = F_\theta(X_t^\theta) + \left(\partial_X F_\theta(X_t^\theta) \cdot \partial_\theta X_t^\theta + \partial_\theta F_\theta(X_t^\theta)\right) \cdot \delta\theta + o(\delta\theta)\end{aligned}$$

We now have all elements to conclude calculating the derivative of $\mathcal{L}$, with some more easy calculus:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \theta} = \int_0^T \big(2\left\langle \partial_X \mathcal{H}(X_t^\theta)\cdot \partial_\theta X_t^\theta, \mathcal{H}(X_t^\theta) - Y_t\right\rangle + \\ \left\langle \lambda_t, \partial_\theta \partial_t X_t^\theta - \partial_X F_\theta(X_t^\theta)\cdot \partial_\theta X_t^\theta - \partial_\theta F_\theta(X_t^\theta)\right\rangle\big)\,\mathrm{dt} \\ + \left\langle \mu, \partial_\theta X_0^\theta - \partial_\theta g_\theta\right\rangle\end{aligned}$$

By the Schwarz theorem, as X is twice continuously differentiable, we have that $\partial_\theta \partial_t X_t^\theta = \partial_t \partial_\theta X_t^\theta$. Integrating by parts, we get:

$$\begin{aligned}\int_0^T \left\langle \lambda_t, \partial_\theta \partial_t X_t^\theta\right\rangle \mathrm{dt} = \left\langle \lambda_T, \partial_\theta X_T^\theta\right\rangle - \left\langle \lambda_0, \partial_\theta X_0^\theta\right\rangle \\ - \int_0^T \left\langle \partial_t \lambda_t, \partial_\theta X_t^\theta\right\rangle \mathrm{dt}\end{aligned}$$

Putting all this together and arranging it, we get:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \theta} = \int_0^T \big\langle \partial_\theta X_t^\theta, 2\partial_X \mathcal{H}(X_t^\theta)^\star \left(\mathcal{H}(X_t^\theta) - Y_t\right) \\ -\partial_t \lambda_t - \partial_X F_\theta(X_t^\theta)^\star \lambda_t\big\rangle\,\mathrm{dt} \\ - \int_0^T \left\langle \lambda_t, \partial_\theta F_\theta(X_t^\theta)\right\rangle \mathrm{dt} + \left\langle \lambda_T, \partial_\theta X_T^\theta\right\rangle + \left\langle \mu - \lambda_0, \partial_\theta X_0^\theta\right\rangle \\ - \left\langle \mu, \partial_\theta g_\theta\right\rangle\end{aligned}$$

We can now define:

$$A_t = -(\partial_X F_\theta(X_t^\theta))^\star$$

and

$$B_t = 2(\partial_X \mathcal{H}(X_t^\theta))^\star(\mathcal{H}(X_t^\theta) - Y_t)$$

and, recalling that λ can be freely chosen, impose that λ is solution of:

$$\partial_t \lambda_t = A_t \lambda_t + B_t$$

with final condition $\lambda_T = 0$. We also choose $\mu = \lambda_0$ so that, finally, we have:

$$\frac{\partial \mathcal{L}}{\partial \theta} = -\int_0^T \left\langle \lambda_t, \partial_\theta F_\theta(X_t^\theta)\right\rangle \mathrm{dt} - \left\langle \lambda_0, \partial_\theta g_\theta\right\rangle$$

which concludes the proof. □

A.2 Appendix 3.3: Incorporating Imperfect Knowledge

A.2.1 A reminder on Chebyshev sets

We begin by giving a definition of Chebyshev sets, taken from Fletcher et al. 2014:

Definition S1. A *Chebyshev set* of a normed space $(E, \|\cdot\|)$ is a subset $\mathcal{C} \subset E$ such that every $x \in E$ admits a unique nearest point in $\mathcal{C}$.

In Euclidean spaces, Chebyshev sets are simply the closed convex subsets. The question of knowing whether it is the case that all Chebyshev sets are closed convex sets in infinite dimensional Hilbert spaces is still an open question. In general, there exists examples of non-convex Chebyshev sets, a famous one being presented in Johnson 1987 for a non-complete inner-product space.

Given the importance of this topic in approximation theory, finding necessary conditions for a set to be Chebyshev and studying the properties of those sets have been the subject of many efforts. Some of those properties are summarized below:

- The metric projection on a boundedly compact Chebyshev set is continuous.
- If the norm is strict, every closed convex space, in particular any finite dimensional subspace is Chebyshev.
- In a Hilbert space, every closed convex set is Chebyshev.

A.2.2 Proof of Proposition S2

We prove the following result:

Proposition S16. *If $\mathcal{F}_p$ is Chebyshev, the optimization problem:*

$$\min_{F_p \in \mathcal{F}_p, F_a \in \mathcal{F}} \|F_a\| \quad \text{subject to} \quad \forall X \in \mathcal{D}, \forall t, \frac{\mathrm{d}X_t}{\mathrm{d}t} = (F_p + F_a)(X_t) \tag{S13}$$

admits a unique solution which element in $\mathcal{F}_p$ is the metric projection of $\mathcal{F}$ onto $\mathcal{F}_p$.

Proof. The idea is to reconstruct the full functional from the trajectories of $\mathcal{D}$. By definition, $\mathcal{A}$ is the set of points reached by trajectories in $\mathcal{D}$ so that:

$$\mathcal{A} = \{x \in \mathcal{R}^d \mid \exists X_\cdot \in \mathcal{D}, \exists t,\ X_t = x\}$$

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. S13, we then have that $(F_p + F_a)(a) = \frac{\mathrm{d}X_t}{\mathrm{d}t}_{|t_0} = F^{\mathcal{D}}(a)$ 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.

As $\mathcal{F}_p$ is a Chebyshev set, the optimization problem:

$$\underset{F_p \in \mathcal{F}_p}{\text{minimize}} \quad \left\|F^{\mathcal{D}} - F_p\right\|$$

has a unique minimum which is the projection of $F^{\mathcal{D}}$ on $\mathcal{F}_p$ and 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. S19. Moreover, if there is (F_p, F_a) satisfying the constraint of eq. S19, 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. Finally, by uniqueness of the projection, if $F_p \neq F_p^\star$ then $\|F_a\| > \|F_a^\star\|$. Thus the minimal pair is unique. □

A.2.3 Parameter estimation in incomplete physical models

Classically, when a set $\mathcal{F}_p \subset \mathcal{F}$ summarising the most important properties of a system is available, this gives a *simplified model* of the true dynamics and the adopted problem is then to fit the trajectories using this model as well as possible, solving:

$$\begin{aligned} &\underset{F_p \in \mathcal{F}_p}{\text{minimize}} && \mathbb{E}_{X \sim \mathcal{D}} L(\widetilde{X}^{X_0}, X) \\ &\text{subject to} && \forall g \in \mathcal{I},\ \widetilde{X}_0^g = g \text{ and } \forall t,\ \frac{\mathrm{d}\widetilde{X}_t^g}{\mathrm{d}t} = F_p(\widetilde{X}_t^g) \end{aligned} \tag{S14}$$

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 makes us lose in terms of accuracy. [3]
- 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 the paper in Section 4.2.6: 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:

$$\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2} + \omega_0^2\theta + \lambda\frac{\mathrm{d}\theta}{\mathrm{d}t} = 0$$

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:

$$F : X \mapsto (\frac{\mathrm{d}\theta}{\mathrm{d}t}, 2(\theta - \frac{\mathrm{d}\theta}{\mathrm{d}t}))$$

It is then easy to see that the estimate of a with the physical model alone can be obtained by minimizing:

$$\int_0^T |e^{-t}\cos t - \cos\sqrt{a}\, t|^2$$

3. This is true in theory, although not necessarily in practice when F overfits a small dataset.

This expression depends on T and thus, depending on the chosen time interval and the way the integral is discretized will almost always give biased estimates. In other words, the estimated value of a won't give us the desired solution $t \mapsto \cos t$.

On the other hand, for a given a, in the APHYNITY framework, the residual must be equal to:

$$F_r^a : X \mapsto (0, (2-a)\theta - 2\frac{\mathrm{d}\theta}{\mathrm{d}t})$$

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.

A.2.4 A discussion of supervision over derivatives

In order to find the appropriate decomposition (F_p, F_a), we use a trajectory-based error by solving:

$$\begin{aligned} \underset{F_p \in \mathcal{F}_p, F_a \in \mathcal{F}}{\text{minimize}} \quad & \|F_a\| \\ \text{subject to} \quad & \forall g \in \mathcal{I},\ \widetilde{X}_0^g = g \text{ and } \forall t,\ \frac{\mathrm{d}\widetilde{X}_t^g}{\mathrm{d}t} = (F_p + F_a)(\widetilde{X}_t^g), \\ & \forall X \in \mathcal{D},\ L(X, \widetilde{X}^{X_0}) = 0 \end{aligned} \tag{S15}$$

In the continuous setting where the data is available at all times t, this problem is in fact equivalent to the following one:

$$\underset{F_p \in \mathcal{F}_p}{\text{minimize}} \quad \mathbb{E}_{X \sim \mathcal{D}} \int \left\| \frac{\mathrm{d}X_t}{\mathrm{d}t} - F_p(X_t) \right\| \tag{S16}$$

where the supervision is done directly over derivatives, obtained through finite-difference schemes. This echoes the proof in Section A.2.2 of the Supplementary where F can be reconstructed from the continuous data.

However, in practice, data is only available at discrete times with a certain time resolution. While eq. S16 is indeed equivalent to eq. S15 in the continuous setting, in the practical discrete one, the way error propagates isn't anymore: For eq. S15 it is controlled over integrated trajectories while for eq. S16 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. S15, 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. S16 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.

A.2.5 Implementation details

We describe here the three use cases studied in the paper for validating APHYNITY. All experiments are implemented with PyTorch Paszke et al. 2019 and the differentiable ODE solvers with the adjoint method implemented in `torchdiffeq`.[4]

4. https://github.com/rtqichen/torchdiffeq

A.2.5.1 Non-linear pendulum

We consider the non-linear damped pendulum problem, governed by the ODE

$$\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2} + \omega_0^2 \sin\theta + \lambda \frac{\mathrm{d}\theta}{\mathrm{d}t} = 0$$

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 $\frac{\mathrm{d}X_t}{\mathrm{d}t} = F(X_t)$ with $F : X \mapsto (\frac{\mathrm{d}\theta}{\mathrm{d}t}, -\omega_0^2 \sin\theta - \lambda \frac{\mathrm{d}\theta}{\mathrm{d}t})$.

Dataset For each train / validation / test split, we simulate a dataset with 25 trajectories of 40 timesteps (time interval $[0, 20]$, timestep $\delta t = 0.5$) with fixed ODE coefficients ($T_0 = 12, \lambda = 0.2$) and varying initial conditions. The simulation integrator is Dormand-Prince Runge-Kutta method of order (4)5 (DOPRI5) J. R. Dormand et al. 1980. We also add a small amount of white gaussian noise ($\sigma = 0.01$) to the state. Note that our pendulum dataset is much more challenging than the ideal frictionless pendulum considered in Greydanus et al. 2019.

Neural network architectures We detail in Table S1 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 Greydanus et al. 2019 is implemented by a MLP that takes the state X_t and outputs a scalar estimation of the Hamiltonian $\mathcal{H}$ of the system: the derivative is then computed by an in-graph gradient of $\mathcal{H}$ with respect to the input: $F(X_t) = \left(\frac{\partial \mathcal{H}}{\partial(\mathrm{d}\theta/\,\mathrm{d}t)}, -\frac{\partial \mathcal{H}}{\mathrm{d}\theta}\right)$.

Table S1. – Neural network architectures for the damped pendulum experiments. n/a corresponds to non-applicable cases.

Method	Physical model	Data-driven model
Neural ODE T. Q. Chen et al. 2018c	n/a	MLP(in=2, units=200, layers=3, out=2)
Hamiltonian Greydanus et al. 2019; Toth et al. 2020	MLP(in=2, units=200, layers=3, out=1)	n/a
APHYNITY Hamiltonian	MLP(in=2, units=200, layers=3, out=1)	MLP(in=2, units=200, layers=3, out=2)
Param ODE (ω_0)	1 trainable parameter ω_0	n/a
APHYNITY Param ODE (ω_0)	1 trainable parameter ω_0	MLP(in=2, units=200, layers=3, out=2)
Param ODE (ω_0, λ)	2 trainable parameters ω_0, λ	n/a
APHYNITY Param ODE (ω_0, λ)	2 trainable parameters ω_0, λ	MLP(in=2, units=200, layers=3, out=2)

Optimization hyperparameters The hyperparameters of the APHYNITY optimization algorithm ($Niter, \lambda_0, \tau_1, \tau_2$) were cross-validated on the validation set

and are shown in Table S2. All models were trained with a maximum number of 5000 steps with early stopping.

Table S2. – 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

A.2.5.2 Reaction-diffusion equations

The system is driven by a 2D FitzHugh-Nagumo type PDE Klaasen et al. 1984

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

where a and b are respectively the diffusion coefficients of u and v, Δ is the Laplace operator. The local reaction terms are $R_u(u, v; k) = u - u^3 - k - v, R_v(u, v) = u - v$.

The state $X = (u, v)$ is defined over a compact rectangular domain $\Omega = [-1, 1]^2$ with periodic boundary conditions. Ω is spatially discretized with a 32×32 2D uniform square mesh grid. The periodic boundary condition is implemented with 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 Section A.2.7.2 of the Supplementary.

Neural network architectures Our F_a here is a 3-layer convolution network (ConvNet) in the APHYNITY. The two input channels are (u, v) and two output ones are $(\frac{\partial u}{\partial t}, \frac{\partial v}{\partial t})$. The purely data-driven Neural ODE uses such ConvNet as its F. The detailed architecture is provided in Table S3.

Table S3. – ConvNet architecture in reaction-diffusion and wave equation experiments, used as data-driven derivative operator in APHYNITY and Neural ODE T. Q. Chen et al. 2018c.

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

The estimated physical parameters θ_p in F_p are simply a trainable vector $(a, b) \in \mathbb{R}^2_+$ or $(a, b, k) \in \mathbb{R}^3_+$.

Optimization hyperparameters We choose to apply the same hyperparameters for all the reaction-diffusion experiments: $Niter = 1, \lambda_0 = 1, \tau_1 = 1 \times 10^{-3}, \tau_2 = 1 \times 10^3$.

A.2.5.3 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:

$$w(x, y, t = 0) = C \times \exp \frac{(x - x_0)^2 + (y - y_0)^2}{\sigma^2} \tag{S17}$$

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 S3. 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 S3.

Optimization hyperparameters We use the same hyperparameters for the experiments: $Niter = 3, \lambda_0 = 1, \tau_1 = 1 \times 10^{-4}, \tau_2 = 1 \times 10^2$.

A.2.6 Ablation study

We conducted ablation studies to show the effectiveness of APHYNITY's adaptive optimization and trajectory-based supervision. In Tables S4, S5, and S6, we consider the following ablation cases:

- *deriv.*: in which F_a is trained with supervision over approximated derivatives on ground truth trajectory, as performed in Greydanus et al. 2019; Cranmer et al. 2020. More precisely, APHYNITY's $\mathcal{L}_{\text{traj}}$ is here replaced with $\mathcal{L}_{\text{deriv}} = \|\frac{\mathrm{d}X_t}{\mathrm{d}t} - F(X_t)\|$ as in Eq. eq. S16, where $\frac{\mathrm{d}X_t}{\mathrm{d}t}$ is approximated by finite differences on X_t.

- *no min.*: in which we train APHYNITY without the minimization of $\|F_a\|$. It corresponds to a naive cooperation scheme as in Le Guen et al. 2020b; Mehta et al. 2020.

- *vanilla optim.*: in which we train APHYNITY by minimizing $\|F_a\|$ without the adaptive optimization of λ shown in Algorithm 2. This case is equivalent to $\lambda = 1, \tau_2 = 0$.

Firstly, we can notice that minimizing the action of $\|F_a\|$ is beneficial in all cases compared to the naive cooperation scheme (no min.) used in Le Guen et al. 2020a; Mehta et al. 2020: for example log MSE= -0.35 vs.-3.97 for the Hamiltonian in the

Table S4. – Ablation study for the damped pendulum.

Method	log MSE	%Error T_0	%Error λ	$\|F_a\|^2$
Hamiltonian deriv.	-0.83±0.3	n/a	n/a	642±121
Hamiltonian no min.	-0.35±0.1	n/a	n/a	837±117
Hamiltonian vanilla optim.	-0.49±0.58	n/a	n/a	165±30
APHYNITY Hamiltonian	**-3.97±1.2**	n/a	n/a	623±68
Param ODE (ω_0) deriv.	-1.02±0.04	5.8±0.4	n/a	136±13
Param ODE (ω_0) no min.	-7.02±1.7	4.5±0.1	n/a	148±49
Param ODE (ω_0) vanilla optim.	-4.30±1.3	4.4±0.2	n/a	90.4±27
APHYNITY Param ODE (ω_0)	**-7.86±0.6**	**4.0±1.4**	n/a	132±11
Param ODE (ω_0, λ) deriv.	-2.61±0.2	5.2±0.1	4.8±0.1	3.2±1.7
Param ODE (ω_0, λ) no min.	-7.60±0.6	3.9±0.3	5.4±1.9	35.5±6.2
Param ODE (ω_0, λ) vanilla optim.	-7.69±1.3	1.0±1.6	2.3±2.5	4.8±7.7
APHYNITY Param ODE (ω_0, λ)	**-8.31±0.3**	**0.12±0.07**	**0.65±0.6**	8.5±2.0
True ODE deriv.	-2.14±0.3	n/a	n/a	4.1±0.6
True ODE no min.	**-8.40±0.2**	n/a	n/a	3.4±0.8
True ODE vanilla optim.	**-8.34±0.4**	n/a	n/a	1.4±0.3
APHYNITY True ODE	**-8.44±0.2**	n/a	n/a	2.3±0.4

Table S5. – Ablation study for the reaction-diffusion equation.

Method	log MSE	%Err a	%Err b	%Err k	$\|F_a\|^2$
Param. PDE (a, b) deriv.	-4.42±0.25	8.20	16.9	n/a	(6.8±0.6)e1
Param. PDE (a, b) no min.	-4.56±0.52	4.08	12.7	n/a	(7.5±1.4)e1
Param. PDE (a, b) vanilla optim.	-4.55±0.11	4.20	10.8	n/a	(7.6±1.0)e1
APHYNITY Param. PDE (a, b)	**-5.10±0.21**	**0.887**	**3.77**	n/a	(6.7±0.4)e1
Param. PDE (a, b, k) deriv.	-4.90±0.06	4.37	8.78	22.0	(1.9±0.3)e-1
Param. PDE (a, b, k) no min.	-8.04±0.03	**0.002**	**0.003**	76.3	(1.5±0.2)e-2
Param. PDE (a, b, k) vanilla optim.	-9.10±0.02	0.018	0.010	0.608	(5.5±2.9)e-7
APHYNITY Param. PDE (a, b, k)	**-9.35±0.02**	0.013	0.014	**0.261**	(1.5±0.4)e-6
True PDE deriv.	-6.03±0.01	n/a	n/a	n/a	(3.1±0.8)e-3
True PDE no min.	-8.12±0.05	n/a	n/a	n/a	(6.1±2.3)e-4
True PDE vanilla optim.	-9.01±0.01	n/a	n/a	n/a	(1.5±0.8)e-6
APHYNITY True PDE	**-9.17±0.02**	n/a	n/a	n/a	(1.4±0.8)e-7

pendulum case. Then when $\|F_a\|$ is minimized, we highlight the importance to use a principled adaptive optimization algorithm (APHYNITY algorithm described in paper) compared to a vanilla optimization: for example in the reaction-diffusion case, log MSE= -4.55 vs.-5.10 for Param PDE(a, b). Finally, when the supervision

Table S6. – Ablation study for the wave equation.

Method	log MSE	%Error c	%Error k	$\|F_a\|^2$
Param PDE (c) deriv.	-1.16±0.48	12.1	n/a	0.00024
Param PDE (c) vanilla optim.	-2.57±0.21	3.1	n/a	43.6
APHYNITY Param PDE (c)	**-4.64±0.21**	**0.3**	n/a	71.0
Param PDE (c, k) deriv.	-4.19±0.36	4.75	9.68	0.00012
Param PDE (c, k) vanilla optim.	-4.93±0.51	0.75	1.9	0.054
APHYNITY Param PDE (c, k)	**-5.92±0.43**	**0.72**	**1.02**	0.44
True PDE vanilla optim.	-4.97±0.49	n/a	n/a	0.23
APHYNITY True PDE	**-5.24±0.45**	n/a	n/a	0.14

occurs on the derivative, both forecasting and parameter identification results are systematically lower than with APHYNITY's trajectory based approach: for example, log MSE=-1.16 vs.-4.64 for Param PDE(c) in the wave equation.

A.2.7 Additional experiments

A.2.7.1 Damped pendulum with varying parameters

To extend the experiments conducted in the paper (section 4.2.6) with fixed parameters ($T_0 = 6, \lambda = 0.2$) and varying initial conditions, we evaluate APHYNITY on a much more challenging dataset where we vary both the parameters (T_0, λ) and the initial conditions between trajectories.

We simulate 500/50/50 trajectories for the train/valid/test sets integrated with DOPRI5. For each trajectory, the period T_0 (resp. the damping coefficient λ) 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), $(\omega_0, \lambda) \in \mathbb{R}_+^2$ for Param ODE (ω_0, λ)).

In this varying ODE context, we also compare to the state-of-the-art univariate time series forecasting method N-Beats Oreshkin et al. 2019.

Results shown in Table S7 are consistent with those presented in the paper. Pure data-driven models (Neural ODE T. Q. Chen et al. 2018c and N-Beats Oreshkin et al. 2019) fail to properly extrapolate the pendulum dynamics. Incomplete physical models (Hamiltonian and ParamODE (ω_0)) are even worse since they do not account for friction. Augmenting them with APHYNITY significantly and consistently improves forecasting results and parameter identification.

Table S7. – 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 λ	$\|F_a\|^2$
data-driven	Neural ODE T. Q. Chen et al. 2018c	-4.35±0.9	n/a	n/a	n/a
	N-Beats Oreshkin et al. 2019	-4.57±0.5	n/a	n/a	n/a
Incomplete physics	Hamiltonian Greydanus et al. 2019; Toth et al. 2020	-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, λ)	**-5.71±0.4**	4.08±0.8	152±129	n/a
	APHYNITY Param ODE (ω_0, λ)	**-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

A.2.7.2 Reaction-diffusion systems with varying diffusion parameters

We also conducted a similar 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 here use 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 S3 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 S8, 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 S8. – 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 T. Q. Chen et al. 2018c	-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

A.2.7.3 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 condition 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 S3 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 A.2.5.3.

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 S9. – 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 (c)	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

A.3 Appendix 3.4: Leveraging the Data from Different Environments

A.3.1 Proof of Proposition S4

Proposition S4 (Existence and Uniqueness)**.** *Assume Ω is convex, then the existence of a minimal decomposition $f^\star, \{g_e^\star\}_{e\in E} \in \mathcal{F}$ of Eq. S24 is guaranteed. Furthermore, if Ω is strictly convex, this decomposition is unique.*

Proof. The optimization problem is:

$$\min_{f, g_e \in \mathcal{F}} \quad \sum_{e\in E} \Omega(g_e) \quad \text{subject to} \quad \forall x^{e,i} \in \hat{\mathcal{T}}, \forall t, \frac{\mathrm{d}x_t^{e,i}}{\mathrm{d}t} = (f + g_e)(x_t^{e,i}) \tag{3}$$

The idea is to first reconstruct the full functional from the trajectories of $\hat{\mathcal{T}}$. By definition, $\mathcal{A}^e$ is the set of points reached by trajectories in $\hat{\mathcal{T}}$ from env. e so that:

$$\mathcal{A}^e = \{x \in \mathcal{R}^d \mid \exists x_\cdot^e \in \hat{\mathcal{T}}, \exists t,\ x_t^e = x\}$$

Then let us define a function f_e^{data} in the following way, $\forall e \in E$, take $a \in \mathcal{A}^e$, we can find $x_\cdot^e \in \hat{\mathcal{T}}$ and t_0 such that $x_{t_0}^e = a$. Differentiating $x_\cdot^e$ at t_0, which is possible by definition of $\hat{\mathcal{T}}$, we take:

$$f_e^{\text{data}}(a) = \left.\frac{\mathrm{d}x_t^e}{\mathrm{d}t}\right|_{t=t_0}$$

For any (f, g_e) satisfying the constraint in Eq. 3, we then have $(f + g_e)(a) = \left.\frac{\mathrm{d}x_t}{\mathrm{d}t}\right|_{t_0} = f_e^{\text{data}}(a)$ for all $a \in \mathcal{A}^e$. Conversely, any pair such that $(f, g_e) \in \mathcal{F} \times \mathcal{F}$ and $f + g_e = f_e^{\text{data}}$, verifies the constraint.

Thus we have the equivalence between Eq. 3 and the following objective:

$$\min_{f\in\mathcal{F}} \quad \sum_e \Omega(f_e^{\text{data}} - f) \tag{S18}$$

The result directly follows from the fact that the objective is a sum of (strictly) convex functions in f and is thus (strictly) convex in f. □

A.3.2 Further details on the generalization with *LEADS*

In this section, we will give more details on the link between our framework and its generalization performance. After introducing the necessary definitions in Sec. A.3.2.1, we show the proofs of the results for the general case in Sec. 3.4.3. Then in Sec. A.3.2.3 we provide the instantiation for linear approximators. Finally, we show how we derived our heuristic instantiation for neural networks in Eq. S28 in Sec. 3.4.3.3 from the existing capacity bound for neural networks.

A.3.2.1 Preliminaries

Table S10 gives the definition of the different capacity instances considered in the paper for each hypothesis space, and the associated distances. We say that a space $\mathcal{H}$ is ε-covered by a set H, with respect to a metric or pseudo-metric $d(\cdot, \cdot)$, if for all $h \in \mathcal{H}$ there exists $h' \in H$ with $d(h, h') \leq \varepsilon$. We define by $\mathcal{N}(\varepsilon, \mathcal{H}, d)$ the cardinality of the smallest H that ε-covers $\mathcal{H}$, also called covering number Shalev-Shwartz et al. 2014. The capacity of each hypothesis space is then defined by the maximum covering number over all distributions. Note that the loss function is involved in every metric in Table S10. For simplicity, we therefore omit the notation of loss function for the hypothesis spaces.

As in Baxter 2000, covering numbers are based on pseudo-metrics. We can verify that all distances in Table S10 are pseudo-metrics:

Proof. This is trivially verified. For example, for the distance $d_{\mathcal{P}}(f+g, f+g')$ given in Table S10, which is the distance between $f+g, f+g' \in f+\hat{\mathcal{G}}$, it is easy to check that the following properties do hold:

- $d_{\mathcal{P}}(f+g, f+g') = 0$ (subtraction of same functions evaluated on same x and y)
- $d_{\mathcal{P}}(f+g, f+g') = d_{\mathcal{P}}(f+g', f+g)$ (evenness of absolute value)
- $d_{\mathcal{P}}(f+g, f+g') \leq d_{\mathcal{P}}(f+g, f+g'') + d_{\mathcal{P}}(f+g'', f+g')$ (triangular inequality of absolute value)

Other distances in Table S10 can be proven to be pseudo-metrics in the same way. □

Table S10. – Capacity definitions of different sets by covering number with associated metric or pseudo-metric.

Capacity	Metric or pseudo-metric	Mentioned in
$\mathcal{C}(\varepsilon, \mathbb{H}^m) := \sup_{\beta\mathcal{P}} \mathcal{N}(\varepsilon, \mathbb{H}^m, d_{\beta\mathcal{P}})$	$d_{\beta\mathcal{P}}((f+g_1,\dots,f+g_m),(f'+g'_1,\dots,f'+g'_m)) = \int_{(\mathcal{A}\times T\mathcal{A})^m} \frac{1}{m} \lvert \sum_{e\in E} \lVert (f+g_e)(x^e)-y^e\rVert^2 - \sum_{e\in E} \lVert (f'+g'_e)(x^e)-y^e\rVert^2 \rvert \, \mathrm{d}\beta\mathcal{P}(\mathrm{x},\mathrm{y})$	Theorem S2; Prop. S17
$\mathcal{C}_{\hat{\mathcal{G}}}(\varepsilon, \hat{\mathcal{F}}) := \sup_{\mathcal{P}} \mathcal{N}(\varepsilon, \hat{\mathcal{F}}, d_{[\mathcal{P},\hat{\mathcal{G}}]})$	$d_{[\mathcal{P},\hat{\mathcal{G}}]}(f,f') = \int_{\mathcal{A}\times T\mathcal{A}} \sup_{g\in\hat{\mathcal{G}}} \lvert \lVert (f+g)(x)-y\rVert^2 - \lVert (f'+g)(x)-y\rVert^2 \rvert \, \mathrm{d}\mathcal{P}(x,y)$	Prop. S5, S17, S19; Cor. S1
$\mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}}) := \sup_{\mathcal{P}} \mathcal{N}(\varepsilon, \hat{\mathcal{G}}, d_{[\mathcal{P},\hat{\mathcal{F}}]})$	$d_{[\mathcal{P},\hat{\mathcal{F}}]}(g,g') = \int_{\mathcal{A}\times T\mathcal{A}} \sup_{f\in\hat{\mathcal{F}}} \lvert \lVert (f+g)(x)-y\rVert^2 - \lVert (f+g')(x)-y\rVert^2 \rvert \, \mathrm{d}\mathcal{P}(x,y)$	Prop. S5, S17, S18
$\mathcal{C}(\varepsilon, f+\hat{\mathcal{G}}) := \sup_{\mathcal{P}} \mathcal{N}(\varepsilon, f+\hat{\mathcal{G}}, d_{\mathcal{P}})$	$d_{\mathcal{P}}(f+g,f+g') = \int_{\mathcal{A}\times T\mathcal{A}} \lvert \lVert (f+g)(x)-y\rVert^2 - \lVert (f+g')(x)-y\rVert^2 \rvert \, \mathrm{d}\mathcal{P}(x,y)$	Prop. S6
$\mathcal{C}(\varepsilon, \hat{\mathcal{G}}, L^1) := \sup_{\mathcal{P}} \mathcal{N}(\varepsilon, \hat{\mathcal{G}}, d_{L^1(\mathcal{P})})$	$d_{L^1(\mathcal{P})}(g,g') = \int_{\mathcal{R}^d} \lVert (g-g')(x)\rVert_1 \, \mathrm{d}\mathcal{P}(x)$	Prop. S18; Theorem S4
$\mathcal{C}(\varepsilon, \hat{\mathcal{G}}, L^2) := \sup_{\mathcal{P}} \mathcal{N}(\varepsilon, \hat{\mathcal{G}}, d_{L^2(\mathcal{P})})$	$d_{L^2(\mathcal{P})}(g,g') = \sqrt{\int_{\mathcal{R}^d} \lVert (g-g')(x)\rVert_2^2 \, \mathrm{d}\mathcal{P}(x)}$	Prop. S7; Lemma S1

A.3.2.2 General Case

Proof of Proposition S5

Proposition S5. *Given m envs., let $\varepsilon_1, \varepsilon_2, \delta > 0, \varepsilon = \varepsilon_1 + \varepsilon_2$. Assume the number of examples n per env. satisfies*

$$n \geq \max\left\{\frac{64}{\varepsilon^2}\left(\frac{1}{m}\log\frac{4\mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_1}{16},\hat{\mathcal{F}})}{\delta} + \log\mathcal{C}_{\hat{\mathcal{F}}}\left(\frac{\varepsilon_2}{16},\hat{\mathcal{G}}\right)\right), \frac{16}{\varepsilon^2}\right\} \tag{S26}$$

Then with probability at least $1-\delta$ (over the choice of training sets $\{\hat{\mathcal{P}}_e\}$), any learner $(f+g_1,\dots,f+g_m)$ will satisfy $\frac{1}{m}\sum_{e\in E} \mathrm{er}_{\mathcal{P}_e}(f+g_e) \leq \frac{1}{m}\sum_{e\in E} \hat{\mathrm{er}}_{\hat{\mathcal{P}}_e}(f+g_e) + \varepsilon$.

Proof. We introduce some extra definitions that are necessary for proving the proposition. Let $\mathcal{H} = f + \hat{\mathcal{G}}$ defined for each $f \in \hat{\mathcal{F}}$, and let us define the product space $\mathcal{H}^m = \{(f+g_1,\dots,f+g_m) : f+g_e \in \mathcal{H}\}$. Functions in this hypothesis space all have the same f, but not necessarily the same g_e. Let $\mathbb{H}$ be the collection of all hypothesis spaces $\mathcal{H} = f + \hat{\mathcal{G}}, \forall f \in \hat{\mathcal{F}}$. The hypothesis space associated to multiple environments is then defined as $\mathbb{H}^m := \bigcup_{\mathcal{H}\in\mathbb{H}} \mathcal{H}^m$.

Our proof makes use of two intermediary results addressed in Theorem S2 and Prop. S17.

Theorem S2 (Baxter 2000, Theorem 4, adapted to our setting)**.** *Assuming $\mathbb{H}$ is a permissible hypothesis space family. For all $\varepsilon > 0$, if the number of examples n of each env. satisfies:*

$$n \geq \max\left\{\frac{64}{m\varepsilon^2}\log\frac{4\mathcal{C}(\frac{\varepsilon}{16},\mathbb{H}^m)}{\delta}, \frac{16}{\varepsilon^2}\right\}$$

Then with probability at least $1-\delta$ (over the choice of $\{\hat{\mathcal{P}}_e\}$), any $(f+g_1,\ldots,f+g_m)$ will satisfy

$$\frac{1}{m}\sum_{e\in E}\mathrm{er}_{\mathcal{P}_e}(f+g_e) \leq \frac{1}{m}\sum_{e\in E}\hat{\mathrm{er}}_{\hat{\mathcal{P}}_e}(f+g_e)+\varepsilon$$

Note that permissibility (as defined in Baxter 2000) is a weak measure-theoretic condition satisfied by many real world hypothesis space families Baxter 2000. We will now express the capacity of $\mathbb{H}^m$ in terms of the capacities of its two constituent component-spaces $\hat{\mathcal{F}}$ and $\hat{\mathcal{G}}$, thus leading to the main result.

Proposition S17. *For all $\varepsilon, \varepsilon_1, \varepsilon_2 > 0$ such that $\varepsilon = \varepsilon_1 + \varepsilon_2$,*

$$\log \mathcal{C}(\varepsilon, \mathbb{H}^m) \leq \log \mathcal{C}_{\hat{\mathcal{G}}}(\varepsilon_1, \hat{\mathcal{F}}) + m \log \mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon_2, \hat{\mathcal{G}}) \tag{S19}$$

Proof of Proposition S17. To prove the proposition it is sufficient to show the property of covering sets for any joint distribution defined on all environments $\beta\mathcal{P}$ on the space $(\mathcal{A}\times T\mathcal{A})^m$. Let us then fix such a distribution $\beta\mathcal{P}$. and let $\bar{\mathcal{P}} = \frac{1}{m}\sum_{e\in E}\mathcal{P}_e$ be the average distribution.

Suppose that F is an ε_1-cover of $(\hat{\mathcal{F}}, d_{[\bar{\mathcal{P}},\hat{\mathcal{G}}]})$ and $\{G_e\}_{e\in E}$ are ε_2-covers of $(\hat{\mathcal{G}}, d_{[\mathcal{P}_e,\hat{\mathcal{F}}]})$. Let $H = \{(x_1,\ldots,x_m) \mapsto ((f+g_1)(x_1),\ldots,(f+g_m)(x_m)),\ f\in F,\ g_e \in G_e\}$, be a set built from the covering sets aforementioned. Note that by definition $|H| = |F|\cdot\prod_{e\in E}|G_e| \leq \mathcal{C}_{\hat{\mathcal{G}}}(\varepsilon_1,\hat{\mathcal{F}})\mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon_2,\hat{\mathcal{G}})^m$ as we take some distribution instances.

For each learner $(f+g_1,\ldots,f+g_m)\in\mathbb{H}^m$ in the hypothesis space, we take any $f'\in F$ such that $d_{[\bar{\mathcal{P}},\hat{\mathcal{G}}]}(f,f')\leq\varepsilon_1$ and $g'_e\in G_e$ for all e such that $d_{[\mathcal{P}_e,\hat{\mathcal{F}}]}(g_e,g'_e)\leq\varepsilon_2$, and we build $(f'+g'_1,\ldots,f'+g'_m)$. The distance is then:

$$\begin{aligned}
&d_{\beta\mathcal{P}}((f+g_1,\ldots,f+g_m),(f'+g'_1,\ldots,f'+g'_m))\\
\leq\ &d_{\beta\mathcal{P}}((f+g_1,\ldots,f+g_m),(f'+g_1,\ldots,f'+g_m))\\
&+d_{\beta\mathcal{P}}((f'+g_1,\ldots,f'+g_m),(f'+g'_1,\ldots,f'+g'_m))\\
&\qquad\qquad\text{(triangular inequality of pseudo-metric)}\\
\leq\ &\frac{1}{m}\left[\sum_{e\in E}d_{\mathcal{P}_e}(f+g_e,f'+g_e)+\sum_{e\in E}d_{\mathcal{P}_e}(f'+g_e,f'+g'_e)\right]\\
&\qquad\qquad\text{(triangular inequality of absolute value)}
\end{aligned}$$

$$\leq \frac{1}{m}\sum_{e\in E} d_{[\mathcal{P}_e,\hat{\mathcal{G}}]}(f,f') + \frac{1}{m}\sum_{e\in E} d_{[\mathcal{P}_e,\hat{\mathcal{F}}]}(g_e,g'_e) \qquad \text{(by definition of } d_{[\mathcal{P}_e,\hat{\mathcal{G}}]} \text{ and } d_{[\mathcal{P}_e,\hat{\mathcal{F}}]})$$

$$= d_{[\bar{\mathcal{P}},\hat{\mathcal{G}}]}(f,f') + \frac{1}{m}\sum_{e\in E} d_{[\mathcal{P}_e,\hat{\mathcal{F}}]}(g_e,g'_e) \leq \varepsilon_1+\varepsilon_2$$

(mean of the distance on different $\mathcal{P}_e$ is the distance on $\bar{\mathcal{P}}$)

To conclude, for any distribution $\beta\mathcal{P}$, when F is an ε_1-cover of $\hat{\mathcal{F}}$ and $\{G_e\}$ are ε_2-covers of $\hat{\mathcal{G}}$, the set H built upon them is an $(\varepsilon_1+\varepsilon_2)$-cover of $\mathbb{H}^m$. Then if we take the maximum over all distributions we conclude that $\mathcal{C}(\varepsilon_1+\varepsilon_2,\mathbb{H}^m) \leq \mathcal{C}_{\hat{\mathcal{G}}}(\varepsilon_1,\hat{\mathcal{F}})\mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon_2,\hat{\mathcal{G}})^m$ and we have Eq. S19. ■

We can now use the bound developed in Prop. S17 and use it together with Theorem S2, therefore concluding the proof of Prop. S5. □

Proof of Proposition S6

Proposition S6. *For all ε,δ with $0<\varepsilon,\delta<1$ if the number of samples n' satisfies*

$$n' \geq \max\left\{\frac{64}{\varepsilon^2}\log\frac{4\mathcal{C}(\frac{\varepsilon}{16},f+\hat{\mathcal{G}})}{\delta},\frac{16}{\varepsilon^2}\right\}, \tag{S27}$$

then with probability at least $1-\delta$ (over the choice of novel training set $\hat{\mathcal{P}}_{e'}$), any learner $f+g_{e'}\in f+\hat{\mathcal{G}}$ will satisfy $\mathrm{er}_{\mathcal{P}_{e'}}(f+g_{e'}) \leq \hat{\mathrm{er}}_{\hat{\mathcal{P}}_{e'}}(f+g_{e'})+\varepsilon$.

Proof. The proof is derived from the following theorem which can be easily adapted to our context:

Theorem S3 (Baxter 2000, Theorem 3). *Let $\mathcal{H}$ a permissible hypothesis space. For all $0<\varepsilon,\delta<1$, if the number of examples n of each env. satisfies:*

$$n \geq \max\left\{\frac{64}{m\varepsilon^2}\log\frac{4\mathcal{C}(\frac{\varepsilon}{16},\mathcal{H})}{\delta},\frac{16}{\varepsilon^2}\right\}$$

Then with probability at least $1-\delta$ (over the choice of dataset $\hat{\mathcal{P}}$ sampled from $\mathcal{P}$), any $h\in\mathcal{H}$ will satisfy

$$\mathrm{er}_{\mathcal{P}}(h) \leq \hat{\mathrm{er}}_{\hat{\mathcal{P}}}(h)+\varepsilon$$

Given that $\hat{\mathcal{P}}_{e'}$ is sampled from the same environment distribution Q, then by fixing the pre-trained f, we fix the space of hypothesis to $f+\hat{\mathcal{G}}$, and we apply the Theorem S3 to obtain the proposition. □

A.3.2.3 Linear case

We provide here the proofs of theoretical bounds given in Sec. 3.4.3.2. See the description in Sup. A.3.4 for the detailed information on the example linear ODE dataset and the training with varying number of environments.

Proof of Proposition 4

Proposition S7. *If for all linear maps* $\mathrm{L}_{\beta G_e} \in \hat{\mathcal{G}}$, $\|\beta G\|_F^2 \leq r$, *if the input space is bounded s.t.* $\|x\|_2 \leq b$, *and the MSE loss function is bounded by* c, *then*

$$\log \mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}}) \leq \lceil {}^{rcd(2b)^2}/_{\varepsilon^2} \rceil \log 2d^2 =: \omega(r, \varepsilon)$$

Proof. Let us take G an $\frac{\varepsilon}{2\sqrt{c}}$-cover of $\hat{\mathcal{G}}$ with L^2-distance: $d_{L^2(\mathcal{P})}$ (see definition in Table S10). Therefore, for each $\mathrm{L}_{\beta G} \in \hat{\mathcal{G}}$ take $g' \in G$ such that $d_{L^2}(\mathrm{L}_{\beta G}, \mathrm{L}_{\beta G'}) \leq \frac{\varepsilon}{2\sqrt{c}}$, then

$$\begin{aligned}
&d_{[\mathcal{P}, \hat{\mathcal{F}}]}(\mathrm{L}_{\beta G}, \mathrm{L}_{\beta G'}) \\
=& \int_{\mathcal{A} \times \mathcal{A}'} \sup_{\mathrm{L}_{\beta F} \in \hat{\mathcal{F}}} \left| \|(\beta F + \beta G)x - y\|_2^2 - \|(\beta F + \beta G')x - y\|_2^2 \right| \mathrm{d}\mathcal{P}(x, y) \\
\leq& \int_{\mathcal{A} \times T\mathcal{A}} \sup_{\mathrm{L}_{\beta F} \in \hat{\mathcal{F}}} \|(\beta G - \beta G')x\|_2 (\|(\beta F + \beta G)x - y\|_2 + \|(\beta F + \beta G')(x) - y\|_2) \, \mathrm{d}\mathcal{P}(x, y) \\
\leq& \sqrt{\int_{\mathcal{A}} \|(\beta G - \beta G')x\|_2 \, \mathrm{d}\mathcal{P}(x)} \sqrt{\int_{\mathcal{A} \times T\mathcal{A}} \sup_{\mathrm{L}_{\beta F} \in \hat{\mathcal{F}}} (\|(\beta F + \beta G)x - y\|_2 + \|(\beta F + \beta G')x - y\|_2)^2 \, \mathrm{d}\mathcal{P}(x, y)} \\
\leq& 2\sqrt{c} \sqrt{\int_{\mathcal{R}^d} \|(\beta G - \beta G')x\|_2 \, \mathrm{d}\mathcal{P}(x)} \leq \varepsilon
\end{aligned}$$

We have the $\mathcal{C}_{\mathcal{F}}(\varepsilon, \hat{\mathcal{G}}) \leq \mathcal{C}(\frac{\varepsilon}{2\sqrt{c}}, \hat{\mathcal{G}}, L^2)$. According to the following lemma:

Lemma S1 (Bartlett et al. 2017, Lemma 3.2, Adapted)**.** *Given positive reals* (a, b, ε) *and positive integer* d. *Let vector* $x \in \mathcal{R}^d$ *be given with* $\|x\|_p \leq b$, $\hat{\mathcal{G}} = \{\mathrm{L}_{\beta G} : \beta G \in \mathcal{R}^{d \times d}, \|\beta G\|_F^2 \leq r\}$ *where* $\|\cdot\|_F$ *is the Frobenius norm. Then*

$$\log \mathcal{C}(\varepsilon, \hat{\mathcal{G}}, L^2) \leq \lceil \frac{rdb^2}{\varepsilon^2} \rceil \log 2d^2$$

And we obtain that

$$\log \mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}}) \leq \left\lceil \frac{rcd(2b)^2}{\varepsilon^2} \right\rceil \log 2d^2 =: \omega(r, \varepsilon)$$

where $\omega(r, \varepsilon)$ is a strictly increasing function w.r.t. r. □

Proof of Proposition 5

Proposition S8. *If for linear maps* $\mathrm{L}_{\beta F} \in \hat{\mathcal{F}}, \|\beta F\|_F^2 \leq r', \mathrm{L}_{\beta G} \in \hat{\mathcal{G}}, \|\beta G\|_F^2 \leq r, \|x\|_2 \leq b$, *and if the MSE loss function is bounded by* c, *given* m *envs. and* n *samples per env., with the probability* $1-\delta$, *the generalization error upper bound is* $\varepsilon = \max\left\{\sqrt{(p+\sqrt{p^2+4q})/2}, \sqrt{16/n}\right\}$ *where* $p = \frac{64}{mn}\log\frac{4}{\delta}$ *and* $q = \frac{64}{n}\left\lceil\left(\frac{r'}{mz^2} + \frac{r}{(1-z)^2}\right)cd(32b)^2\right\rceil \log 2d^2$ *for any* $0 < z < 1$.

Proof. This can be derived from Prop. S5 with the help of Prop. S7 for linear maps. If we take the lower bounds of two capacities $\log \mathcal{C}_{\hat{\mathcal{F}}}(\frac{\varepsilon_1}{16}, \hat{\mathcal{G}})$ and $\log \mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_2}{16}, \hat{\mathcal{F}})$ for the linear maps hypothesis spaces $\hat{\mathcal{F}}, \hat{\mathcal{G}}$, then the number of required samples per env. n now can be expressed as follows:

$$n = \max\left\{\frac{64}{\varepsilon^2}\left(\frac{1}{m}\log\frac{4}{\delta} + \frac{1}{m}\left\lceil\frac{r'cd(32b)^2}{\varepsilon_1^2}\right\rceil\log 2d^2 + \left\lceil\frac{rcd(32b)^2}{\varepsilon_2^2}\right\rceil\log 2d^2\right), \frac{16}{\varepsilon^2}\right\}$$

To simplify the resolution of the equation above, we take $\varepsilon_1 = z\varepsilon$ for any $0 < z < 1$, then $\varepsilon_2 = \varepsilon - \varepsilon_1 = (1-z)\varepsilon$. Then by resolving the equation, the generalization margin is then upper bounded by ε with:

$$\varepsilon = \max\left\{\sqrt{\frac{p+\sqrt{p^2+4q}}{2}}, \sqrt{\frac{16}{n}}\right\}$$

where $p = \frac{64}{mn}\log\frac{4}{\delta}$ and $q = \frac{64}{n}\left\lceil\left(\frac{r}{mz^2} + \frac{r'}{(1-z)^2}\right)cd(32b)^2\right\rceil\log 2d^2$. □

A.3.2.4 Nonlinear case: instantiation for neural networks

We show in this section how we design a concrete model for nonlinear dynamics following the general guidelines given in Sec. 3.4.3.1. This is mainly composed of the following two parts: (a) choosing an appropriate approximation space and (b) choosing a penalization function Ω for this space. It is important to note that, even if the bounds given in the following sections may be loose in general, it could provide useful intuitions on the design of the algorithms which can be validated by experiments in our case.

Choosing approximation space

We choose the space of feed-forward neural networks with a fixed architecture. Given the universal approximation properties of neural networks Kidger et al.

2020, and the existence of efficient optimization algorithms Chizat et al. 2018, this is a reasonable choice, but other families of approximating functions could be used as well.

We then consider the function space of neural networks with D-layers with inputs and outputs in $\mathcal{R}^d$: $\hat{\mathcal{F}}_{\text{NN}} = \{\nu : x \mapsto \sigma_D(W_D \cdots \sigma_1(W_1 x))) : x, \nu(x) \in \mathcal{R}^d\}$, D is the depth of the network, σ_j is a Lipschitz activation function at layer j, and W_j weight matrix from layer $j-1$ to j. The number of adjustable parameters is fixed to W for the architecture. This definition covers fully connected NNs and convolutional NNs. Note that the Fourier Neural Operator Z. Li et al. 2021 used in the experiments for NS can be also covered by the definition above, as it performs alternatively the convolution in the Fourier space.

Choosing penalization Ω

Now we choose an Ω for the space above. Let us first introduce a practical way to bound the capacity of $\hat{\mathcal{G}} \in \hat{\mathcal{F}}_{\text{NN}}$. Proposition S18 tells us that for a fixed NN architecture (implying constant parameter number W and depth D), we can control the capacity through the maximum output norm R and Lipschitz norm L defined in the proposition.

Proposition S18. *If for all neural network $g \in \hat{\mathcal{G}}$, $\|g\|_\infty = \operatorname{ess\,sup} |g| \leq R$ and $\|g\|_{\text{Lip}} \leq L$, with $\|\cdot\|_{\text{Lip}}$ the Lipschitz semi-norm, then:*

$$\log \mathcal{C}_{\hat{\mathcal{F}}}(\varepsilon, \hat{\mathcal{G}}) \leq \omega(R, L, \varepsilon) \tag{S20}$$

where $\omega(R, L, \varepsilon) = c_1 \log \frac{RL}{\varepsilon} + c_2$ for $c_1 = 2W$ and $c_2 = 2W \log 8e\sqrt{c}D$, with c the bound of MSE loss. $\omega(R, L, \varepsilon)$ is a strictly increasing function w.r.t. R and L.

Proof. To link the capacity to some quantity that can be optimized for neural networks, we need to apply the following theorem:

Theorem S4 (Haussler 1992, Theorem 11, Adapted)**.** *With the neural network function space $\hat{\mathcal{F}}_{NN}$, let W be the total number of adjustable parameters, D the depth of the architecture. Let $\hat{\mathcal{G}} \subseteq \hat{\mathcal{F}}_{NN}$ be all functions into $[-R, R]^d$ representable on the architecture, and all these functions are at most L-Lipschitz. Then for all $0 < \varepsilon < 2R$,*

$$\mathcal{C}(\varepsilon, \hat{\mathcal{G}}, L^1) \leq \left(\frac{2e \cdot 2R \cdot DL}{\varepsilon}\right)^{2W}$$

Here, we need to prove firstly that the $\hat{\mathcal{F}}$-dependent capacity of $\hat{\mathcal{G}}$ is bounded by a scaled independent capacity on L^1 of itself. We suppose that the MSE loss function (used in the definitions in Table S10) is bounded by some constant c. This

is a reasonable assumption given that the input and output of neural networks are bounded in a compact set. Let us take G an $\frac{\varepsilon}{2\sqrt{c}}$-cover of $\hat{\mathcal{G}}$ with L^1-distance: $d_{L^1(\mathcal{P})}$ (see definition in Table S10). Therefore, for each $g \in \hat{\mathcal{G}}$ take $g' \in G$ such that $d_{L^1}(g, g') \leq \frac{\varepsilon}{2\sqrt{c}}$, then

$$
\begin{aligned}
d_{[\mathcal{P},\hat{\mathcal{F}}]}(g, g') &= \int_{\mathcal{A}\times\mathcal{A}'} \sup_{f\in\hat{\mathcal{F}}} |\|(f+g)(x) - y\|_2^2 - \|(f+g')(x) - y\|_2^2| \, \mathrm{d}\mathcal{P}(x, y) \\
&\leq \int_{\mathcal{A}\times T\mathcal{A}} \sup_{f\in\hat{\mathcal{F}}} \|(g-g')(x)\|_2 (\|(f+g)(x) - y\|_2 + \|(f+g')(x) - y\|_2) \, \mathrm{d}\mathcal{P}(x, y) \\
&\leq 2\sqrt{c} \int_{\mathcal{R}^d} \|(g-g')(x)\|_1 \, \mathrm{d}\mathcal{P}(x) \leq \varepsilon
\end{aligned}
$$

Then we have the first inequality $\mathcal{C}_{\mathcal{F}}(\varepsilon, \hat{\mathcal{G}}) \leq \mathcal{C}(\frac{\varepsilon}{2c}, \hat{\mathcal{G}}, L^1)$. As we suppose that $\|g\|_\infty \leq R$ for all $g \in \hat{\mathcal{G}}$, then for all $g \in \hat{\mathcal{G}}$, we have $g(x) \in [-R, R]^d$. We now apply the Theorem S4 on $\hat{\mathcal{G}}$, we then have the following inequality

$$
\log \mathcal{C}\left(\frac{\varepsilon}{2\sqrt{c}}, \hat{\mathcal{G}}, L^1\right) \leq 2W \log \frac{8e\sqrt{c}\, DRL}{\varepsilon} \tag{S21}
$$

where e is the base of the natural logarithm, W is the number of parameters of the architecture, D is the depth of the architecture. Then if we consider W, c, D as constants, the bound becomes:

$$
\log \mathcal{C}\left(\frac{\varepsilon}{2\sqrt{c}}, \hat{\mathcal{G}}, L^1\right) \leq c_1 \log \frac{RL}{\varepsilon} + c_2 = \omega(R, L, \varepsilon) \tag{S22}
$$

for $c_1 = 2W$ and $c_2 = 2W \log 8e\sqrt{c}\, D$. □

This leads us to choose for Ω a strictly increasing function that bounds $\omega(R, L, \varepsilon)$. Given the inequality (Eq. S20), this choice for Ω will allow us to bound practically the capacity of $\hat{\mathcal{G}}$.

Minimizing Ω will then reduce the effective capacity of the parametric set used to learn g_e. Concretely, we choose for Ω:

$$
\Omega(g_e) = \|g_e\|_\infty^2 + \alpha \|g_e\|_{\mathrm{Lip}}^2 \tag{7}
$$

where $\alpha > 0$ is a hyper-parameter. This function is strictly convex and attains its unique minimum at the null function.

With this choice, let us instantiate Prop. S5 for our familly of NNs. Let $r = \sup_{g\in\hat{\mathcal{G}}} \Omega(g)$, and $\omega(r, \varepsilon) = c_1 \log \frac{r}{\varepsilon\sqrt{\alpha}} + c_2$ (strictly increasing w.r.t. the r) for given parameters $c_1, c_2 > 0$. We have:

Proposition S19. *If $r = \sup_{g \in \hat{\mathcal{G}}} \Omega(g)$ is finite, the number of samples n in Eq. S26, required to satisfy the error bound in Proposition S5 with the same $\delta, \varepsilon, \varepsilon_1$ and ε_2 becomes:*

$$n \geq \max \left\{ \frac{64}{\varepsilon^2} \left(\frac{1}{m} \log \frac{4\mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_1}{16}, \hat{\mathcal{F}})}{\delta} + \omega \left(r, \frac{\varepsilon_2}{16} \right) \right), \frac{16}{\varepsilon^2} \right\} \tag{S23}$$

Proof. If $\Omega(g_e) \leq r$, we have $2 \log R \leq \log r$ and $2 \log L + \log \alpha \leq \log r$, then

$$\log RL \leq \log \frac{r}{\sqrt{\alpha}}$$

We can therefore bound $\omega(R, L, \varepsilon)$ by

$$\omega(R, L, \varepsilon) = c_1 \log \frac{RL}{\varepsilon} + c_2 \leq c_1 \log \frac{r}{\varepsilon \sqrt{\alpha}} + c_2 = \omega(r, \varepsilon)$$

The result follows from Proposition S18. □

This means that the number of required samples will decrease with the size the largest possible $\Omega(g) = r$. The optimization process will reduce $\Omega(g_e)$ until a minimum is reached. The maximum size of the effective hypothesis space is then bounded and decreases throughout training. In particular, the following result follows:

Corollary S1. Optimizing Eq. S12 for a given λ, we have that the number of samples n in Eq. S26 required to satisfy the error bound in Proposition S5 with the same $\delta, \varepsilon, \varepsilon_1$ and ε_2 is:

$$n \geq \max \left\{ \frac{64}{\varepsilon^2} \left(\frac{1}{m} \log \frac{4\mathcal{C}_{\hat{\mathcal{G}}}(\frac{\varepsilon_1}{16}, \hat{\mathcal{F}})}{\delta} + \omega \left(\lambda\kappa, \frac{\varepsilon_2}{16} \right) \right), \frac{16}{\varepsilon^2} \right\} \tag{S24}$$

where $\kappa = \sum_{e \in E} \sum_{i=1}^{l} \int_0^T \left\| \frac{\mathrm{d}x_s^{e,i}}{\mathrm{d}t} \right\|^2 \mathrm{d}s$.

Proof. Denote $\mathcal{L}_\lambda(f, \{g_e\})$ the loss function defining Eq. S12. Consider a minimizer $(f^\star, \{g_e^\star\})$ of $\mathcal{L}_\lambda$. Then:

$$\mathcal{L}_\lambda(f^\star, \{g_e^\star\}) \leq \mathcal{L}_\lambda(0, \{0\}) = \kappa$$

which gives:

$$\forall e, \ \Omega(g_e^\star) \leq \sum_e \Omega(g_e^\star) \leq \lambda\kappa$$

Table S11. – Details for the results of evaluation error in test in Fig. S22.

Samples/env.	Method	$m = 1$	$m = 2$	$m = 4$	$m = 8$
$n = 2 \cdot K$	*LEADS no min.*	8.13±5.56 e-2	6.81±4.44 e-2	4.92±4.26 e-2	4.50±3.10 e-2
	LEADS (Ours)		**5.11±3.20 e-2**	**3.93±2.88 e-2**	**2.10±0.96 e-2**
$n = 4 \cdot K$	*LEADS no min.*	4.08±2.57 e-2	3.96±2.56 e-2	3.10±2.08 e-2	2.23±1.44 e-2
	LEADS (Ours)		**2.74±1.96 e-2**	**1.61±1.24 e-2**	**1.02±0.74 e-2**

Defining $\hat{\mathcal{G}} = \{g \in \hat{\mathcal{F}} \mid \Omega(g) \leq \lambda\kappa\}$, we then have that Eq. S12 is equivalent to:

$$\min_{f \in \hat{\mathcal{F}}, \{g_e\}_{e \in E} \in \hat{\mathcal{G}}} \sum_{e \in E} \left(\frac{\Omega(g_e)}{\lambda} + \sum_{i=1}^{l} \int_0^T \left\| \frac{\mathrm{d}x_s^{e,i}}{\mathrm{d}t} - (f + g_e)(x_s^{e,i}) \right\|^2 \mathrm{d}s \right) \tag{S25}$$

and the result follows from Proposition S19. □

We can then decrease the sample complexity in the chosen NN family by: (a) increasing the number of training environments engaged in the framework, and (b) decreasing $\Omega(g_e)$ for all g_e, with $\Omega(g_e)$ instantiated as in Sec. 3.4.3.1. Ω provides a bound based on the largest output norm and the Lipschitz constant for a family of NNs. The experiments (Sec. 3.4.4) confirm that this is indeed an effective way to control the capacity of the approximating function family. Note that in our experiments, the number of samples needed in practice is much smaller than suggested by the theoretical bound.

Table S12. – Detailed results of evaluation error in test for Fig. S25. The case of $m = 1$ is ignored in the table, as three methods are reduced to model *One-Per-Env.* when applied to the groups containing 1 env. each. The vertical and horizontal arrows indicate that the table cells share the same value.

Samples/env.	Method	$m = 1$	$m = 2$	$m = 4$	$m = 8$
$n = 1 \cdot K$	*One-For-All*	↑	0.22±0.06	0.33±0.06	0.47±0.04
	One-Per-Env.	7.87±7.54 e-3	⟶	⟶	⟶
	LEADS (Ours)	↓	**3.65±2.99 e-3**	**2.39±1.83 e-3**	**1.37±1.14 e-3**
$n = 2 \cdot K$	*One-For-All*	↑	0.22±0.04	0.36±0.07	0.60±0.11
	One-Per-Env.	1.38±1.61 e-3	⟶	⟶	⟶
	LEADS (Ours)	↓	**8.65±9.61 e-4**	**8.40±9.76 e-4**	**6.02±6.12 e-4**
$n = 4 \cdot K$	*One-For-All*	↑	0.19±0.02	0.31±0.04	0.50±0.04
	One-Per-Env.	1.36±1.25 e-4	⟶	⟶	⟶
	LEADS (Ours)	↓	**1.10±0.92 e-4**	**1.03±0.98 e-4**	**9.66±9.79 e-5**
$n = 8 \cdot K$	*One-For-All*	↑	0.16±0.03	0.35±0.06	0.52±0.06
	One-Per-Env.	5.98±5.13 e-5	⟶	⟶	⟶
	LEADS (Ours)	↓	**5.47±4.63 e-5**	**4.52±3.98 e-5**	**3.94±3.49 e-5**

Table S13. – Test MSE of experiments on LV ($m = 4, n = 1 \cdot K$) with different empirical norms.

Empirical Norm	$p = 1$	$p = 2$	$p = 3$	$p = 10$	$p = \infty$
Test MSE	2.30e-3	2.36e-3	2.34e-3	3.41e-3	6.12e-3

A.3.3 Optimizing Ω in practice

In Sec. 3.4.3.3, we developed an instantiation of the *LEADS* framework for neural networks. We proposed to control the capacity of the g_es components through a penalization function Ω defined as $\Omega(g_e) = \|g_e\|_\infty^2 + \alpha\|g_e\|_{\text{Lip}}^2$. This definition ensures the properties required to control the sample complexity.

However, in practice, both terms in $\Omega(g_e)$ are difficult to compute as they do not yield an analytical form for neural networks. For a fixed activation function, the Lipschitz-norm of a trained model only depends on the model parameters and, for our class of neural networks, can be bounded by the spectral norms of the weight matrices, as described in Sec. 3.4.4.4. This allows for a practical implementation.

The infinity norm on its side depends on the domain definition of the function and practical implementations require an empirical estimate. Since there is no trivial estimator for the infinity norm of a function, we performed tests with different proxies such as the *empirical* L^p and L^∞ norms, respectively defined as $\|g_e\|_{L^p(\hat{\mathcal{P}}_e)} = \left(\frac{1}{n}\sum_{x\in\hat{\mathcal{P}}_e} |g_e(x)|^p\right)^{1/p}$ for $1 \leq p < \infty$ and $\|g_e\|_{L^\infty(\hat{\mathcal{P}}_e)} = \max_{x\in\hat{\mathcal{P}}_e} |g_e(x)|$. Here $|\cdot|$ is an ℓ^2 vector norm. Note that on a finite set of points, these norms reduce to vector norms $\|(|g_e(x_1)|, \ldots, |g_e(x_n)|)^\top\|_p$. They are then all equivalent on the space defined by the training set. Table S13 shows the results of experiments performed on LV equation with different $1 \leq p \leq \infty$. Overall we found that L^p for small values of p worked better and chose in our experiments set $p = 2$.

Moreover, using both minimized quantities $\|g_e\|^2_{L^2(\hat{\mathcal{P}}_e)}$ and the spectral norm of the product of weight matrices, denoted $L(g_e)$ and $\Pi(g_e)$ respectively, we can give a bound on $\Omega(g_e)$. First, for any x in the compact support of $\mathcal{P}_e$, we have that, fixing some $x_0 \in \hat{\mathcal{P}}_e$:

$$|g_e(x)| \leq |g_e(x) - g_e(x_0)| + |g_e(x_0)|$$

For the first term:

$$|g_e(x) - g_e(x_0)| \leq \|g_e\|_{\text{Lip}}|x - x_0| \leq \Pi(g_e)|x - x_0|$$

and the support of $\mathcal{P}_e$ being compact by hypothesis, denoting by δ its diameter:

$$|g_e(x) - g_e(x_0)| \leq \delta\Pi(g_e)$$

Moreover, for the second term:

$$|g_e(x_0)| = \sqrt{|g_e(x_0)|^2} \leq \sqrt{L(g_e)}$$

and summing both contributions gives us the bound:

$$\|g_e\|_\infty \leq \delta\Pi(g_e) + \sqrt{L(g_e)}$$

so that:

$$\Omega(g_e) \leq (\delta + \alpha)\Pi(g_e) + \sqrt{L(g_e)}$$

Note that this estimation is a crude one and improvements can be made by considering the closest x_0 from x and taking δ to be the maximal distance between points not from the support of $\mathcal{P}_e$ and $\hat{\mathcal{P}}_e$.

Finally, we noticed that minimizing $\|\frac{g_e}{id}\|^2_{L^2(\hat{\mathcal{P}}_e)}$ in domains bounded away from zero gave better results as normalizing by the norm of the output allowed to adaptively rescale the computed norm. Formally, minimizing this quantity does not fundamentally change the optimization as we have that:

$$\forall x, \frac{1}{M^2}|g_e(x)|^2 \leq \left|\frac{g_e(x)}{x}\right|^2 \leq \frac{1}{m^2}|g_e(x)|^2$$

meaning that:

$$\frac{1}{M^2}L(g_e) \leq \left\|\frac{g_e}{id}\right\|^2_{L^2(\hat{\mathcal{P}}_e)} \leq \frac{1}{m^2}L(g_e)$$

where m, M are the lower and upper bound of $|x|$ on the support of $\mathcal{P}_e$ with $m > 0$ by hypothesis (the quantity we minimize is still higher than $L(g_e)$ even if this is not the case).

A.3.4 Additional experimental details

A.3.4.1 Details on the environment dynamics

Lotka-Volterra (LV). The model dynamics follow the ODE:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \alpha u - \beta uv, {}^{\mathrm{d}v}/_{\mathrm{d}t} = \delta uv - \gamma v$$

with u, v the number of prey and predator, $\alpha, \beta, \gamma, \delta > 0$ defining how the two species interact. The initial conditions u_0^i, v_0^i are sampled from a uniform distribution $P_0 = \text{Unif}([1,2]^2)$. We characterize the dynamics by $\theta = (\alpha/\beta, \gamma/\delta) \in \Theta = \{0.5, 1, 1.44, 1.5, 1.86, 2\}^2$. An env. e is then defined by parameters θ_e sampled from a uniform distribution over the parameter set Θ.

Gray-Scott (GS). The governing PDE is:

$$\frac{\partial u}{\partial t} = D_u \Delta u - uv^2 + F(1-u), {}^{\partial v}/_{\partial t} = D_v \Delta v + uv^2 - (F+k)v$$

where the u, v represent the concentrations of two chemical components in the spatial domain S with periodic boundary conditions. D_u, D_v denote the diffusion coefficients respectively for u, v, and are held constant to $D_u = 0.2097, D_v = 0.105$, and F, k are the reaction parameters depending on the environment. As for the initial conditions $(u_0, v_0) \sim P_0$, we place 3 2-by-2 squares at uniformly sampled positions in S to trigger the reactions. The values of (u_0, v_0) are fixed to $(0, 1)$ outside the squares and to $(1-\epsilon, \epsilon)$ with a small $\epsilon > 0$ inside. An env. e is defined by its parameters $\theta_e = (F_e, k_e) \in \Theta = \{(0.037, 0.060), (0.030, 0.062), (0.039, 0.058)\}$. We consider a set of θ_e parameters uniformly sampled from the environment distribution Q on Θ.

Navier-Stokes (NS). We consider the Navier-Stokes PDE for incompressible flows:

$$\frac{\partial w}{\partial t} = -v \cdot \nabla w + \nu \Delta w + \xi \qquad \nabla \cdot v = 0$$

where v is the velocity field, $w = \nabla \times v$ is the vorticity, both v, w lie in a spatial domain S with periodic boundary conditions, ν is the viscosity and ξ is the constant forcing term in the domain S. We fix $\nu = 10^{-3}$ across the envs. We sample the initial conditions $w_0^e \sim P_0$ as in Z. Li et al. 2021. An env. e is defined by its forcing term $\xi_e \in \Theta_\xi = \{\xi_1, \xi_2, \xi_3, \xi_4\}$ with

$$\begin{aligned}
\xi_1(x,y) &= 0.1(\sin(2\pi(x+y)) + \cos(2\pi(x+y))) \\
\xi_2(x,y) &= 0.1(\sin(2\pi(x+y)) + \cos(2\pi(x+2y))) \\
\xi_3(x,y) &= 0.1(\sin(2\pi(x+y)) + \cos(2\pi(2x+y))) \\
\xi_4(x,y) &= 0.1(\sin(2\pi(2x+y)) + \cos(2\pi(2x+y)))
\end{aligned}$$

where $(x, y) \in S$ is the position in the domain S. We uniformly sampled a set of forcing terms from Q on Θ_ξ.

Linear ODE. We take an example of linear ODE expressed by the following formula:

$$\frac{\mathrm{d}u_t}{\mathrm{d}t} = \mathrm{L}_{\beta Q \beta \Lambda \beta Q^\top}(u_t) = \beta Q \beta \Lambda \beta Q^\top u_t$$

where $u_t \in \mathcal{R}^8$ is the system state, $\beta Q \in M_{8,8}(\mathcal{R})$ is an orthogonal matrix such that $\beta Q \beta Q^\top = 1$, and $\beta\Lambda \in M_{8,8}(\mathcal{R})$ is a diagonal matrix containing eigenvalues. We sample $\beta\Lambda_e$ from a uniform distribution on $\Theta_{\beta\Lambda} = \{\beta\Lambda_1, \ldots, \beta\Lambda_8\}$, defined for each $\beta\Lambda_i$ by:

$$[\beta\Lambda_i]_{jj} = \begin{cases} 0, & \text{if } i = j \text{ for } i, j \in \{1, \ldots, 8\}, \\ -0.5, & \text{otherwise.} \end{cases}$$

which means that the i-th eigenvalue is set to 0, while others are set to a common value -0.5.

A.3.4.2 Details on the experiments with a varying number of environments

We conducted large-scale experiments respectively for linear ODEs (Sec. 3.4.3.2, Fig. S22) and LV (Sec. 3.4.4, Fig. S25) to compare the tendency of *LEADS* w.r.t. the theoretical bound and the baselines by varying the number of environments available for the instantiated model.

To guarantee the comparability of the test-time results, we need to use the same test set when varying the number of environments. We therefore propose to firstly generate a global set of environments, separate it into subgroups for training, then we test these separately trained models on the global test set.

We performed the experiments as follows:

- In the training phase, we consider $M = 8$ environments in total in the environment set E_{total}. We denote here the cardinality of an environment set E by $\mathrm{card}(E)$, the environments are then arranged into $b = 1, 2, 4$ or 8 disjoint groups of the same size, *i.e.* $\{E_1, \ldots, E_b\}$ such that $\bigcup_{i=1}^{b} E_i = E_{\text{total}}$, $\mathrm{card}(E_1) = \cdots = \mathrm{card}(E_b) = \lfloor M/b \rfloor =: m$, where m is the number of environments per group, and $E_i \cap E_j = \emptyset$ whenever $i \neq j$. For example, for $m = 1$, all the original environments are gathered into one global environment, when for $m = 8$ we keep all the original environments. The methods are then instantiated respectively for each E_i. For example, for *LEADS* with b environment groups, we instantiate *LEADS*$_1, \ldots,$ *LEADS*$_b$ respectively on $E_1, \ldots, E_b$. Other frameworks are applied in the same way.

  Note that when $m = 1$, having $b = 8$ environment groups of one single environment, *One-For-All*, *One-Per-Env.* and *LEADS* are reduced to *One-Per-Env.* applied on all M environments. We can see in Fig. S25 that each group of plots starts from the same point.

- In the test phase, the performance of the model trained with the group E_i is tested with the test samples of the corresponding group. Then we take the mean error over all b groups to obtain the results on all M environments. Note that the result at each point in figures S22 and S25 is calculated on the same total test set, which guarantees the comparability between results.

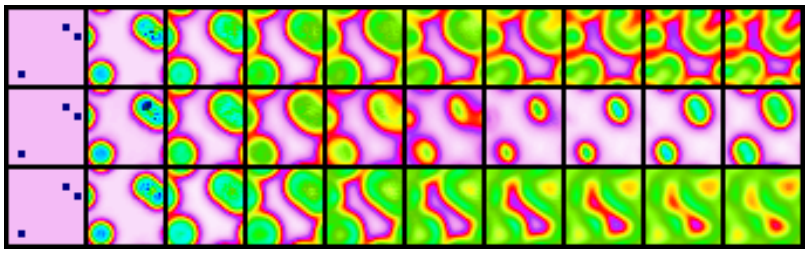
We show also in tables S12 and S11 the detailed results used for the plots in Fig. S25.

A.3.4.3 Implementation

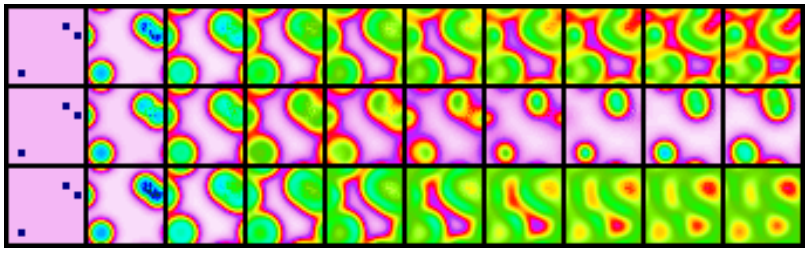
The implementation of *LEADS* is available with the datasets for LV, GS and NS in the supplemental material alongside the present document.

A.3.4.4 Full-length trajectories

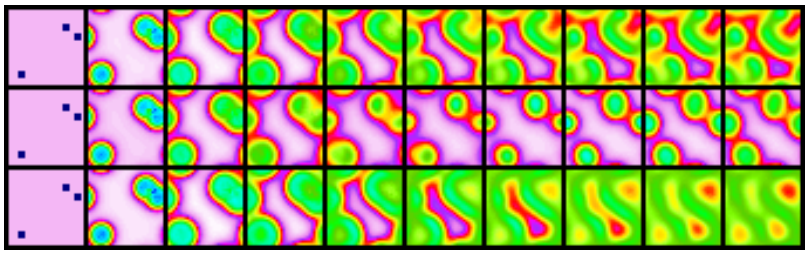
We provide in figures S1-S4 the full-length sample trajectories for GS and NS of Fig. S23.



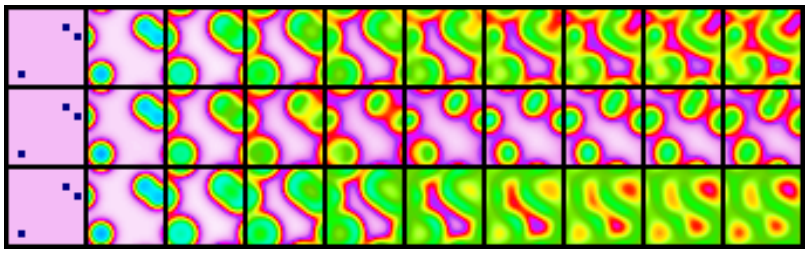
(a) *One-Per-Env.*



(b) FT-NODE

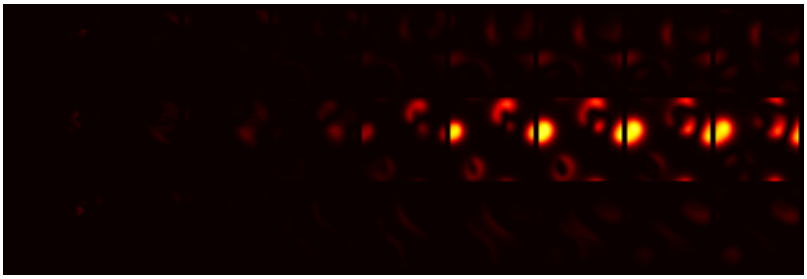


(c) *LEADS*

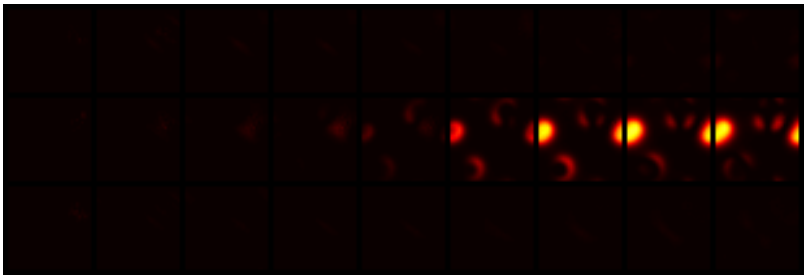


(d) Ground truth

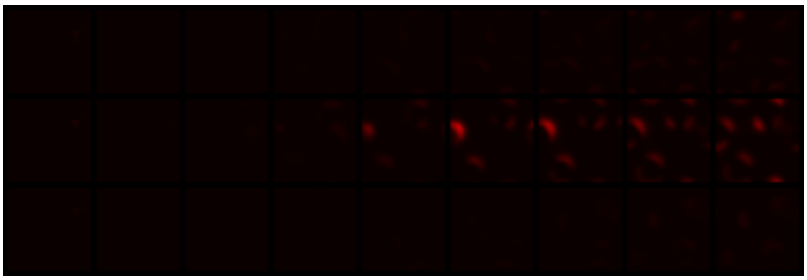
Figure S1. – Full-length prediction comparison of Fig. S23 for GS. In each figure, from top to bottom, the trajectory snapshots are output respectively from 3 training environments. The temporal resolution of each sequence is $\Delta t = 40$.



(a) Difference between *One-Per-Env.* and Ground truth

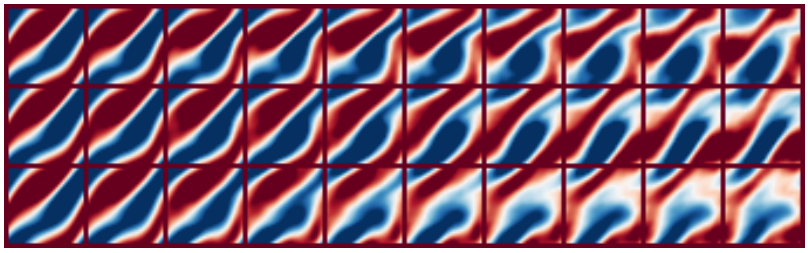


(b) Difference between FT-NODE and Ground truth

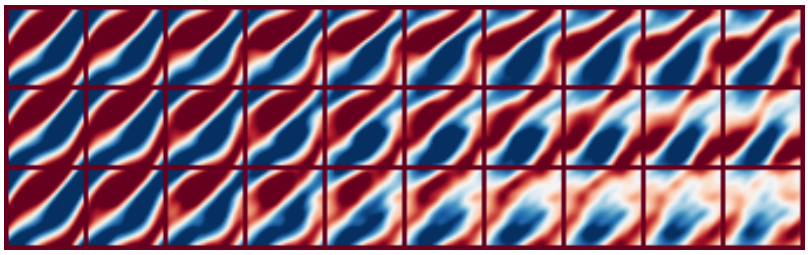


(c) Difference between *LEADS* and Ground truth

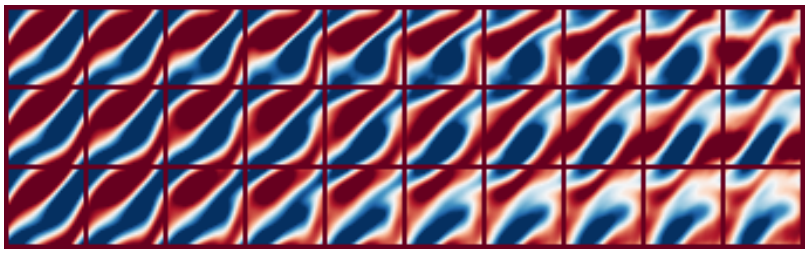
Figure S2. – Full-length error maps of Fig. S23 for GS. In each figure, from top to bottom, the trajectory snapshots correspond to 3 training environments, one per row. The temporal resolution of each sequence is $\Delta t = 40$.



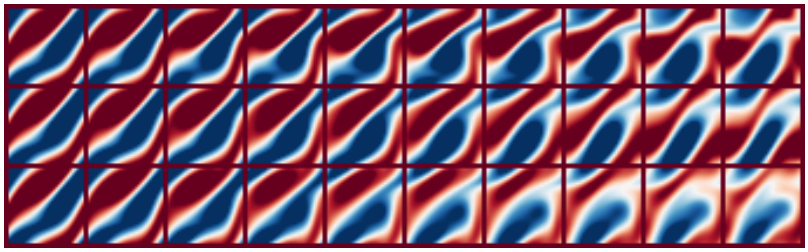
(a) *One-Per-Env.*



(b) FT-NODE

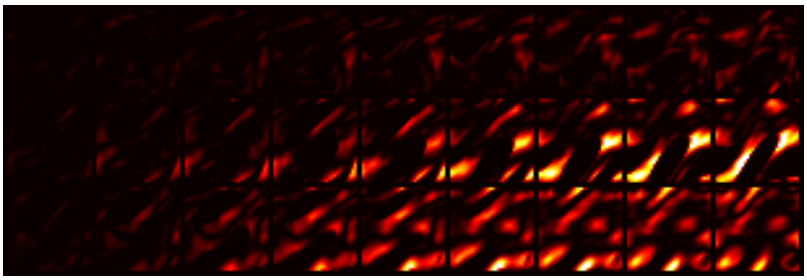


(c) *LEADS*

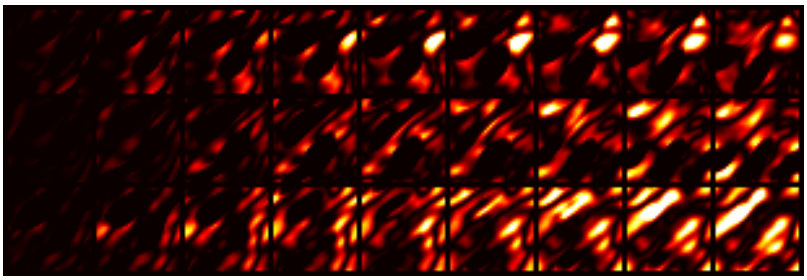


(d) Ground truth

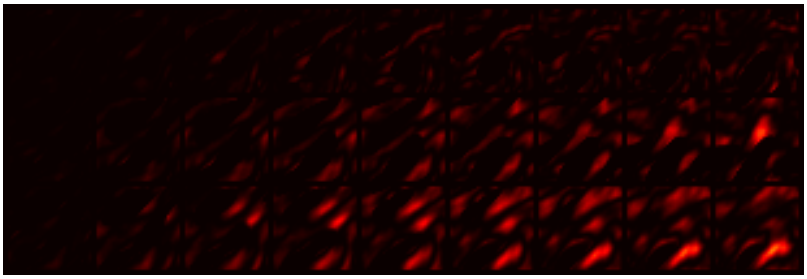
Figure S3. – Full-length prediction comparison of Fig. S23 for NS. In each figure, from top to bottom, the trajectory snapshots correspond to 3 training environments. The temporal resolution of each sequence is $\Delta t = 1$.



(a) Difference between *One-Per-Env.* and Ground truth



(b) Difference between FT-NODE and Ground truth



(c) Difference between *LEADS* and Ground truth

Figure S4. – Full-length error maps of Fig. S23 for NS. In each figure, from top to bottom, the trajectory snapshots correspond to from 3 training environments. The temporal resolution of each sequence is $\Delta t = 1$.

APPENDIX CHAPTER 4

B.1 Appendix 4.2: Analysis of Classification Networks

B.1.1 From the Monge problem to Dynamical Optimal Transport

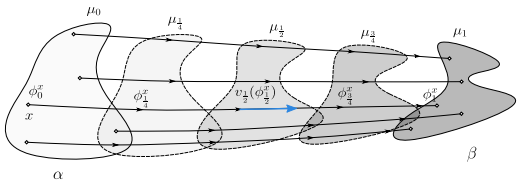


Figure S1. – The Figure illustrates successive steps of the dynamic transportation of α to β together with the notations used in the text. Each step could for example correspond to a transformation performed by an elementary module of a ResNet.

Instead of directly pushing α to β in $\mathcal{R}^d$, it is possible to view α and β as points in a space of measures, and consider trajectories from α to β in this abstract space. Thus, a way to transport the probability mass from α to β is a curve between two points in this space. The curve corresponding to the optimal mapping is then the *shortest* one, in other words it is the *geodesic curve* between the two points.

More formally, let us introduce the *Wasserstein metric space* $\mathbb{W}_p(\mathcal{R}^d)$, *i.e.* the space of absolutely continuous measures of $\mathcal{R}^d$ with finite p-th moment endowed with the Wasserstein distance:

$$W_p(\mu,\nu) = \min_{T_\sharp\mu=\nu} \mathcal{C}(T)^{\frac{1}{p}}$$

when costs of the form $c(x,y) = \|x-y\|_p^p$ are considered, for some integer $p \geq 2$. As $\mathbb{W}_p(\mathcal{R}^d)$ is a space of measures, α and β are seen as points of this space of measures, and thus, any continuous path linking both distributions defines a gradual transformation from α to β and a mapping transporting α to β.

The following result (from Theorem 5.27 of Santambrogio 2015) motivates the dynamical formulation of OT:

Proposition S20. *$\mathbb{W}_p$ is a geodesic space, meaning that, for any measures $\mu,\nu \in \mathbb{W}_p$, there exists a geodesic curve $(\mu_t)_{t\in[0,1]}$ between μ and ν.*

Thus, according to this result, finding the optimal mapping between two distributions amounts to finding a curve of minimal length in a certain abstract measure space. However, it still does not provide much in the way of a practically useful algorithm. The following theorem makes a formal link with fluid dynamics and basically states that moving probability masses from one distribution to another is the same as moving fluid densities from one configuration to another under a certain velocity field Santambrogio 2015:

Theorem S1. *Given α and β absolutely continuous w.r.t. the Lebesgue measure and $(\mu_t)_{t\in[0,1]}$ the geodesic curve with $\mu_0 = \alpha$ and $\mu_1 = \beta$, we can associate a vector field $v_t \in L^p(\mu_t)$ that solves the continuity equation*[1]*:*

$$\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$$

with:

$$W_p^p(\alpha, \beta) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{dt}$$

In other words, the geodesic curve $(\mu_t)_{t\in[0,1]}$ between both distributions, together with the minimal energy velocity vector field v solve the continuity equation. Moreover, its energy along this path is precisely equal to the Wasserstein distance $W_p^p(\alpha, \beta)$. If this vector field of minimal energy v could be obtained, probability mass could be displaced according to the flow defined by the continuity equation, and the geodesic curve could be retrieved. Thus, we can reformulate the problem as a problem of optimal control, where v is the control variate:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \mathcal{C}^{\text{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} dt \\ \text{subject to} \quad & \partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0, \mu_0 = \alpha, \mu_1 = \beta \end{aligned} \tag{S1}$$

It is worth noting that this approach not only gives a mapping between the two distributions but it also gives the entire geodesic curve so that smooth interpolations in $\mathbb{W}_p(\mathcal{R}^d)$ can be recovered.

B.1.2 Regularity of Optimal Transport Mappings

In this section, we recall some classical and more recent results of regularity for Optimal Transport mappings. This is an intricate subject and the problem had been open for some time after OT theory had been established. The most

1. ∂_t is the partial derivative operator *w.r.t.* variable t, and $\nabla\cdot$ the divergence operator *w.r.t.* space.

important results have been established through the study of the Monge-Ampère equation by Caffarelli then De Philippis and Figalli. Extensions for larger families of costs were developed by Ma, Trudinger and Wang X. Ma et al. 2005a but this is out of the scope of this work.

In particular, Theorem 6.27 of Ambrosio et al. 2005 gives a classical almost-everywhere regularity result:

Theorem S2. *If $c(x, y) = \|x - y\|^p$ for $p > 1$, and α and β have compact supports with $d(supp(\alpha), supp(\beta)) > 0$, then the Optimal Transport map T between α and β is $\alpha - a.e.$ differentiable and its Jacobian $\nabla T(x)$ has non-negative eigenvalues $\alpha - a.s.$*

More recently, results summarized below, which correspond to Theorems 4.23, 4.24 and Remark 4.25 of A. Figalli 2017, state that the OT map has one degree of regularity more than the initial transported density:

Theorem S3. *Suppose there are X, Y, bounded open sets, such that the densities of α and β are null in their respective complements and bounded away from zero and infinity over them respectively.*

Then, if Y is convex, there exists $\eta > 0$ such that the OT map T between α and β is $C^{0,\eta}$ over X.

If Y isn't convex, there exists two relatively closed sets A, B in X, Y respectively such that $T \in C^{0,\eta}(X \setminus A, Y \setminus B)$, where A and B are of null Lebesgue measure.

Moreover, if the densities are in $C^{k,\eta}$, then $C^{0,\eta}$ can be replaced by $C^{k+1,\eta}$ in the conclusions above. In particular, if the densities are smooth, then the transport map is a diffeomorphism (between the reduced input and target domains if the target support is not convex).

B.1.3 Finding the right cost

The following proposition shows that, under some technical assumptions, there always exists a ground transport cost which finds the desired mapping:

Proposition S21. *With the same notations as in the paper, suppose that a given UDT task between α and β is solved by a mapping T.*

Then, if we assume that T is a differentiable map with a Jacobian everywhere invertible, there exists a cost function c such that the corresponding Monge problem yields T as its unique optimal transport map.

Proof. Suppose that T verifies the assumption. Let us define the following cost:

$$c(x, y) = \|T(x) - y\|_2^2$$

In this case, c is differentiable w.r.t. its first variable by differentiability of T. For $x_0 \in \mathcal{A}$, we then have that:

$$\forall y,\ \nabla_{x_0} c(x_0, y) = 2\ {}^t(\text{Jac}_{x_0} T)(T(x_0) - y)$$

which is injective in y by invertibility of $\text{Jac}_{x_0} T$. Thus c verifies the Twist condition as defined in Santambrogio 2015 and we can conclude about the existence and uniqueness of the OT map of the corresponding Monge problem using Remark 1.24. of the same reference.

Moreover, for any transport map T', we have that $\mathcal{C}(T') \geq 0$ and we also have that $\mathcal{C}(T) = 0$ which shows that T is indeed the OT map for c by unicity of the optimum. □

□

Note that the proof is not constructive. In practice, such a cost can handcrafted using knowledge about the task, as has been done in the Imbalanced dataset experiment, or it could even be learned if some paired samples are available as a supervision (along with some additional assumptions on the desired mapping for example).

B.1.5 Some Elements of Optimal Transport Theory

We state here the most important results of Optimal Transport theory and its dynamical formulation. Our main reference is Santambrogio 2015. Villani 2008 is another classical reference. The dynamical formulation of OT has been of great importance, both theoretically and practically. It stems mainly from the work of Benamou and Brenier Benamou et al. 2000.

B.1.5.1 Optimal Transport

OT studies the task of "transporting" mass from one configuration to another while minimizing the effort as described by a certain ground cost c. Let α and β be two absolutely continuous distributions. The Monge formulation of OT is:

$$\begin{aligned} \underset{T}{\text{minimize}} \quad & \mathcal{C}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{subject to} \quad & T_\sharp \alpha = \beta \end{aligned} \tag{S2}$$

We then have the following result, proven for example in Theorem 1.17 of Santambrogio 2015, which gives a condition on the cost under which problem eq. S2 has a unique minimum.

Theorem S4. *α, β absolutely continuous measures on $\mathbb{R}^d$. If $c(x, y) = h(x - y)$, with h strictly convex, then there exists a unique T such that $\mathcal{C}(T)$ is minimal.*

B.1.4 Additional Samples

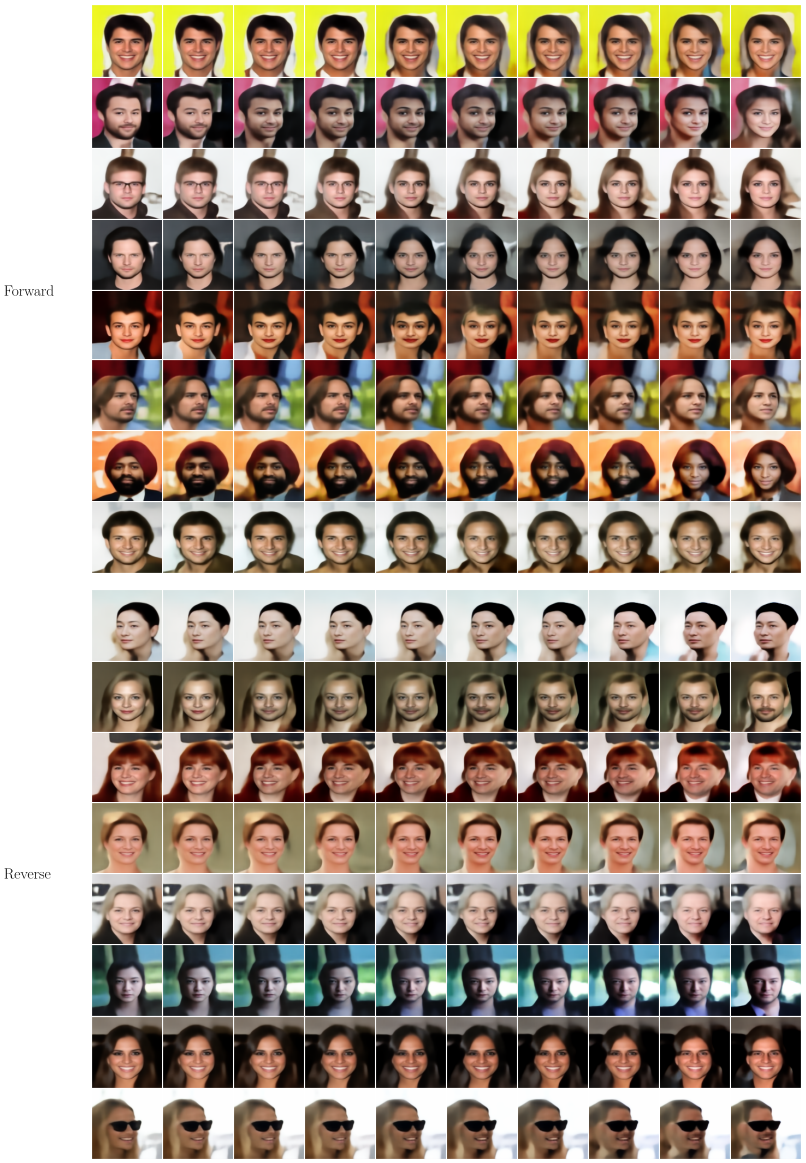


Figure S2. – Male to Female, and Back.

B.1.5.2 Dynamical Formulation

Instead of directly pushing samples of α to β in $\mathbb{R}^d$, we can view α and β as points in a space of measures, and consider trajectories from α to β in this space.A way to transport the probability mass from α to β is a curve between two points in this space. The curve corresponding to the optimal mapping is the *shortest* one, in other words it is the *geodesic curve* between α and β. More formally, we introduce the *Wasserstein metric space* $\mathbb{W}_p(\mathbb{R}^d)$, *i.e.* the space of absolutely continuous measures of $\mathbb{R}^d$ with finite p-th moment endowed with the Wasserstein distance:

$$W_p(\mu,\nu) = \min_{T_\sharp\mu=\nu} \mathcal{C}(T)^{\frac{1}{p}}$$

when costs $c(x,y) = \|x-y\|_q^p$ are considered, for $q, p > 1$. The OT map can then be seen as a trajectory of minimal length between α and β, in other words a *geodesic*. The following result (from Theorem 5.27 of Santambrogio 2015) motivates this approach:

Theorem S5. *$\mathbb{W}_p$ is a geodesic space, meaning that, for any measures $\mu, \nu \in \mathbb{W}_p$, there exists a geodesic curve $(\mu_t)_{t\in[0,1]}$ between μ and ν.*

Thus, according to this result, finding the optimal mapping between two distributions amounts to finding a curve of minimal length in a certain abstract measure space. However, it still does not provide much in the way of a practically useful algorithm. The following theorem makes a formal link with fluid dynamics and basically states that moving probability masses from one distribution to another is the same as moving fluid densities from one configuration to another under a certain velocity field Santambrogio 2015:

Theorem S6. *Given α and β absolutely continuous w.r.t. the Lebesgue measure and $(\mu_t)_{t\in[0,1]}$ the geodesic curve with $\mu_0 = \alpha$ and $\mu_1 = \beta$, we can associate a vector field $v_t \in L^p(\mu_t)$ that solves the continuity equation*[2]*:*

$$\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$$

with:

$$W_p^p(\alpha,\beta) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{dt}$$

In other words, the geodesic curve $(\mu_t)_{t\in[0,1]}$ between the two distributions and the minimal energy velocity vector field v solve the continuity equation. Moreover, the energy along this path is precisely equal to the Wasserstein distance $W_p^p(\alpha,\beta)$.

2. ∂_t is the partial derivative operator *w.r.t.* variable t, and $\nabla\cdot$ the divergence operator *w.r.t.* space.

If this vector field of minimal energy v could be obtained, probability mass could be displaced according to the flow defined by the continuity equation, and the geodesic curve could be retrieved. Thus, we can reformulate the problem as a problem of optimal control, where v is the control variate:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \mathcal{C}^{\text{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{d}t \\ \text{subject to} \quad & \partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0, \mu_0 = \alpha, \mu_1 = \beta \end{aligned} \tag{S3}$$

B.1.6 Regularity of Optimal Transport Maps

In this section, we recall some classical and more recent results of regularity for Optimal Transport mappings. This is an intricate subject and the problem was open for some time after OT theory had been established. The most important results have been established through the study of the Monge-Ampère equation by Caffarelli then De Philippis and Figalli. Extensions for larger families of costs were developed by Ma, Trudinger and Wang X. Ma et al. 2005b but this is out of the scope of this work. In particular, Theorem 6.27 of Ambrosio et al. 2005 gives a classical almost-everywhere regularity result:

Theorem S7. *If $c(x, y) = \|x - y\|^p$ for $p > 1$, and α and β have compact supports with $d(supp(\alpha), supp(\beta)) > 0$, then the optimal transportation map T between α and β is $\alpha - a.e.$ differentiable and its Jacobian $\nabla T(x)$ has non-negative eigenvalues $\alpha - a.s.$*

More recently, results summarized below, which correspond to Theorems 4.23, 4.24 and Remark 4.25 of A. Figalli 2017, state that the optimal transportation map has one degree of regularity more than the initial transported density:

Theorem S8. *Suppose there are X, Y, bounded open sets, such that the densities of α and β are null in their respective complements and bounded away from zero and infinity over them respectively. Then, if Y is convex, there exists $\eta > 0$ such that the OT map T between α and β is $C^{0,\eta}$ over X. If Y isn't convex, there exists two relatively closed sets A, B in X, Y respectively, such that $T \in C^{0,\eta}(X \setminus A, Y \setminus B)$, where A and B are of null Lebesgue measure.*

Moreover, if the densities are in $C^{k,\eta}$, then $C^{0,\eta}$ can be replaced by $C^{k+1,\eta}$ in the conclusions above. In particular, if the densities are smooth, then the transport map is a diffeomorphism (between the reduced input and target domains if the target support is not convex).

B.1.7 Additional Results

B.1.7.1 Visualization of the Transport on MNIST

If we pretrain an autoencoder on MNIST, we can use its encoder as the encoder of the ResNet and freeze it during training. This makes it possible to visualize the transport of the data by decoding, using the pretrained decoder, the output of each residual block. We show this below on MNIST. In Figure S3, we see the decodings of a basic ResNet trained to achieve 99.4% test accuracy.

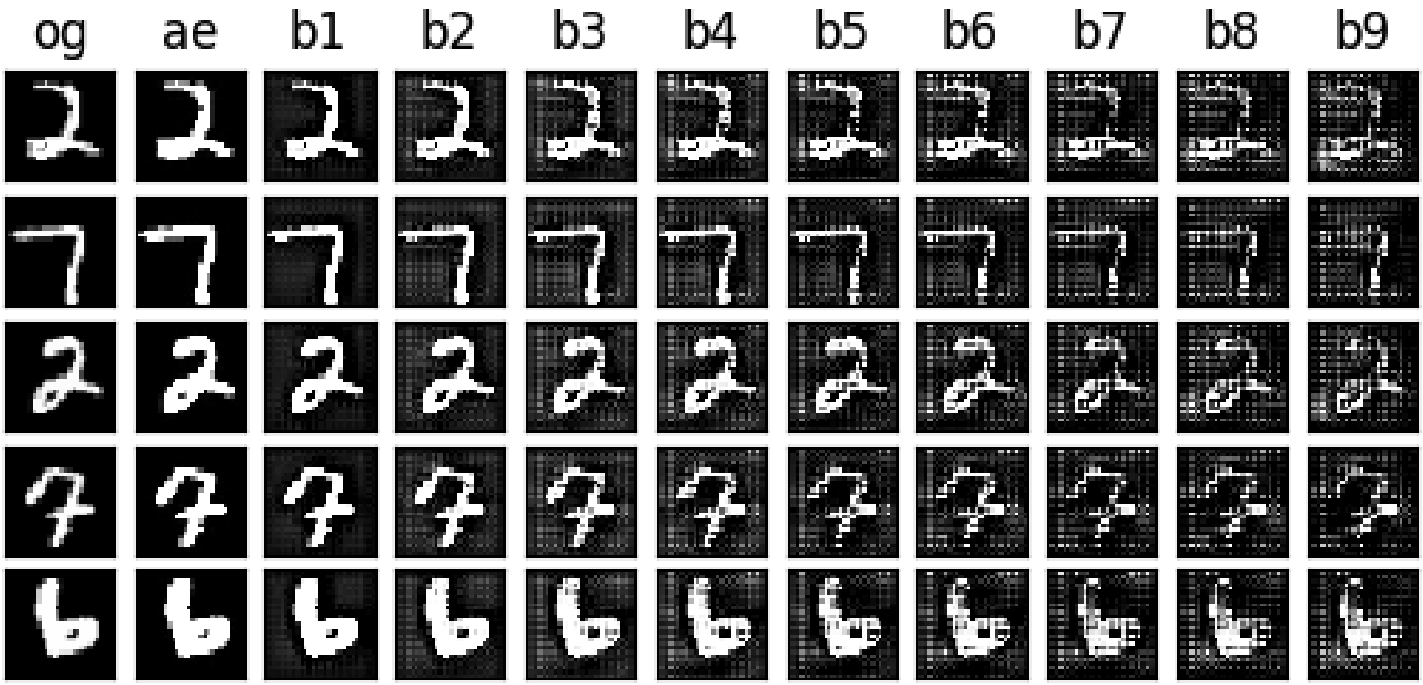


Figure S3. – Decodings of the internal representations (the outputs of the residual blocks) after training a ResNet9 on MNIST (og: original image, ae: encoding, b1: output of block 1...)

We add the transport cost with $\lambda_0 = 5$, $\tau = 1$ and $s = 5$. The performance barely drops (99.3% test accuracy) but we can see in Figure S4 the effect of the regularization as the decodings change much less.

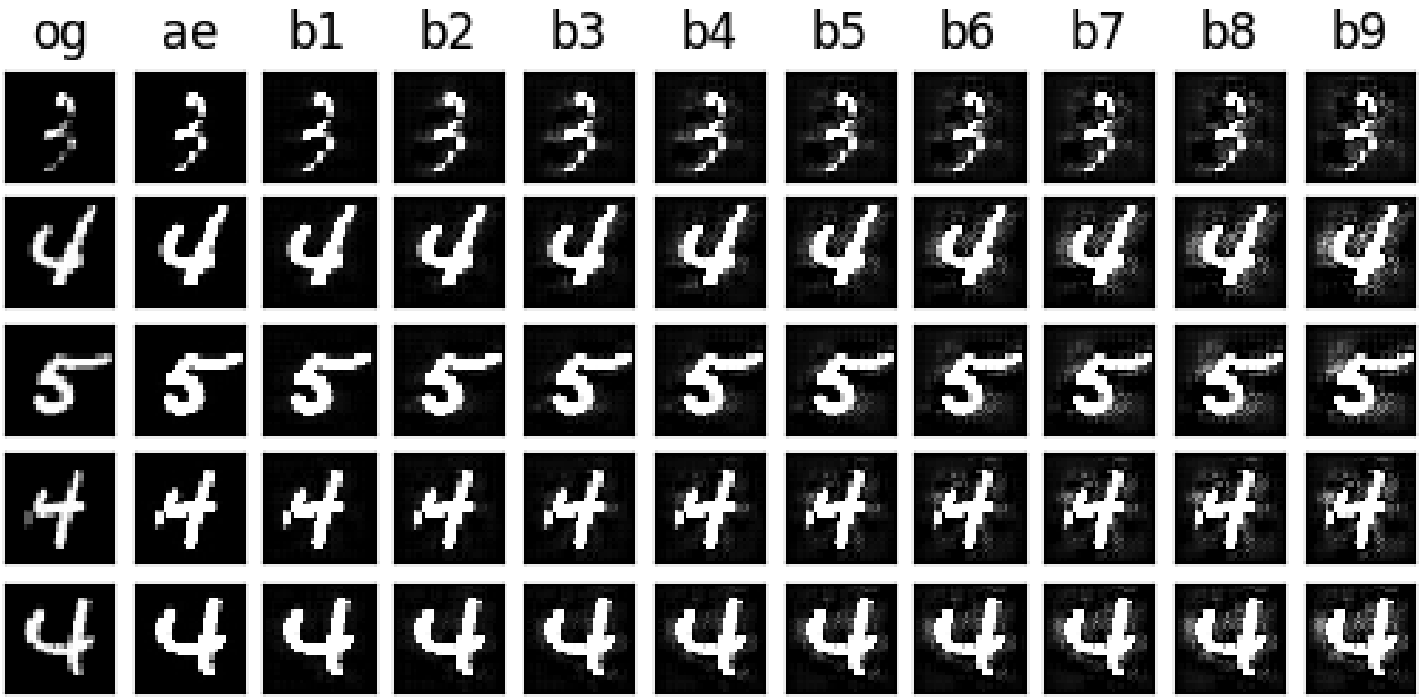


Figure S4. – Decodings of the internal representations (the outputs of the residual blocks) after training a LAP-ResNet9 on MNIST (og: original image, ae: encoding, b1: output of block 1...)

B.1.7.2 Additional Results with Fixed λ

In this section, we show some additional experimental results with a model where, instead of using an adaptive optimization algorithm, we simply take the transport cost as a regularizer, thus giving us a minimization objective:

$$\mathcal{L}(\theta) + \lambda \ \ \mathcal{C}(\theta)$$

This is an easier and more straightforward method, simply considering a relaxed constraint in the optimization problem. Aside from the advantage of simpler implementation, it allows for easier fine-tuning of the regularization hyperparameter which is useful when the datasets and networks are big. The adaptivity is lost but this still leads to better test accuracy than the non-regularized networks. Results on the same tasks as in Section 6 are below.

Training set size	ResNet	LAP-ResNet	Regularized ResNet, $\lambda = 0.2$
50 000	91.49, [91.40, 91.59]	**91.94**, [91.84, 92.04]	91.36, [91.28, 91.44]
30 000	88.61, [88.47, 88.75]	**89.41**, [89.31, 89.50]	88.50, [88.38, 88.61]
20 000	85.73, [85.59, 85.87]	**86.74**, [86.61, 86.87]	85.82, [85.70, 85.93]
10 000	79.25, [79.00, 79.49]	**80.90**, [80.74, 81.06]	80.15, [80.02, 80.28]
5 000	70.32, [70.00, 70.63]	**72.58**, [72.36, 72.79]	72.03, [71.71, 72.34]
4 000	67.80, [67.55, 68.07]	**70.12**, [69.81, 70.42]	69.64, [69.35, 69.94]
1 000	49.22, [48.69, 49.74]	**51.14**, [50.69, 51.59]	50.38, [49.92, 50.82]
500	41.55, [41.14, 41.96]	**42.92**, [42.54, 43.29]	42.30, [41.88, 42.73]
100	26.98, [25.98, 27.97]	25.34, [24.63, 26.10]	**27.53**, [26.59, 28.47]

Table S1. – Average highest test accuracy and 95% confidence interval of ResNet9 over 20 instances on CIFAR10 with training sets of different sizes (in %)

Training set size	ResNet	LAP-ResNet	Regularized ResNet, $\lambda \in \{0.05, 0.2\}$
50 000	72.32, [72.08, 72.56]	72.43, [72.25, 72.61]	**72.62**, [72.41, 72.83]
25 000	64.34, [64.10, 64.57]	64.34, [64.11, 64.58]	**64.76**, [64.52, 65.00]
10 000	49.27, [48.84, 49.69]	**50.57**, [50.34, 50.80]	50.46, [50.19, 50.72]
5 000	34.74, [33.90, 35.58]	37.97, [37.68, 38.27]	**38.44**, [37.99, 38.89]
1 000	15.66, [15.23, 16.08]	**16.42**, [16.10, 16.75]	16.03, [15.55, 16.52]

Table S2. – Average highest test accuracy and 95% confidence interval of ResNet9 over 10 instances on CIFAR100 with training sets of different sizes (in %)

Training set size	ResNeXt	LAP-ResNeXt	Regularized ResNeXt, $\lambda = 0.01$
50 000	72.97, [71.79, 74.14]	**76.11**, [75.32, 76.89]	75.96, [74.92, 77.01]
25 000	62.55, [60.18,64.92]	**64.11**, [62.25, 65.96]	64.10, [62.36, 65.84]
12 500	45.90, [43.16, 48.67]	**48.23**, [46.39, 50.07]	47.77, [45.93, 49.62]

Table S3. – Average highest test accuracy and 95% confidence interval of ResNeXt50 over 10 instances on CIFAR100 with training sets of different sizes (in %)

Finally, we point out that the least action principle acts by speeding up training in the first epochs as seen for the training of ResNeXt50 models on CIFAR100 in Figure S5. Batch training times are similar for the 3 models in Figure S5 on the same hardware (around 0.7 seconds).

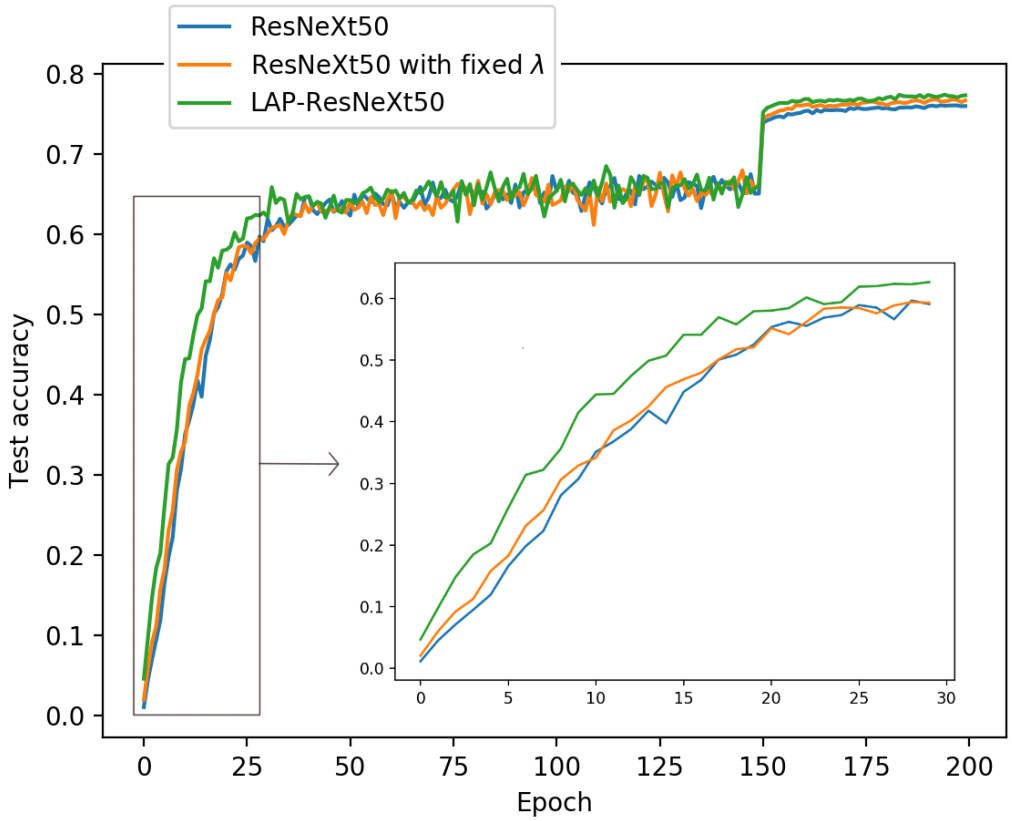


Figure S5. – Test accuracy during training of ResNeXt50 models on CIFAR100

B.1.7.3 Implementation Details

These are further implementation details about the experiments in Sections 6 and B.1.7.2. Orthogonal initialization with gain 0.01 is used for all ResNet models. Kaiming initialization is used for all ResNeXt models. SGD is used for training all models. The momentum is always set to 0.9 and weight decay to 0.0001. For ResNet models, the learning rate is 0.01 and is divided by 5 at epochs 120, 160 and 200 (when the training goes that far). For ResNeXt models, the learning rate is 0.1 and is divided by 10 at epochs 150, 225 and 250. Batch size is 128 for all experiments. Architectures of ResNet K. He et al. 2016 and ResNeXt Xie et al. 2017 blocks are standard and exactly as in the references. The ResNets used are single representation ResNets (i.e. containing one residual stage) with 9 blocks. The ResNeXt architecture used is the ResNeXt-50-32×4d from Xie et al. 2017.

B.1.7.4 Additional Results on 2D Toy Data

Here is a comparison of our method with batch normalization (BN), which is known to impact the loss surface's geometry Bjorck et al. 2018.We find that our method cooperates well with BN to improve test accuracy on the same 2D task as in Section 4 when the model is too small (1 block, Table S4), too big (100 blocks, Table S5), badly initialized ($\mathcal{N}(0, 5)$ initialization, Table S6) and when the dataset is small (50 points, Table S7). LAP-ResNets use $\lambda_0 = 0.1$, $\tau = 0.1$ and $s = 5$.

	No batch normalization	Batch normalization
ResNet	76.6, [73.1, 80.2]	75.4, [72.3, 78.6]
Regularized ResNet, $\lambda = 0.005$	76.5, [73.0, 80.0]	75.6, [72.2, 78.9]
LAP-ResNet	82.1, [79.5, 84.7]	**84.6**, [81.5, 87.6]

Table S4. – Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 1000 points and 1 block (in %)

	No batch normalization	Batch normalization
ResNet	89.1, [87.2, 91.00]	99.4, [99.0, 99.8]
Regularized ResNet, $\lambda = 0.09$	69.7, [65.6, 73.7]	99.5, [98.9, 1.00]
LAP-ResNet	75.7, [72.8, 78.6]	**99.8**, [99.7, 1.00]

Table S5. – Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 1000 points and 100 blocks (in %)

	No batch normalization	Batch normalization
ResNet	90.2, [88.8, 91.5]	98.0, [97.2, 98.8]
Regularized ResNet, $\lambda = 0.04$	89.7, [88.2, 91.3]	**99.7**, [99.5, 99.9]
LAP-ResNet	79.1, [75.3, 83.0]	99.4, [99.0, 99.8]

Table S6. – Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with a $\mathcal{N}(0, 5)$ initialization (in %)

	No batch normalization	Batch normalization
ResNet	88.2, [85.5, 90.1]	92.9, [90.9, 94.9]
Regularized ResNet, $\lambda = 0.04$	93.5, [91.4, 95.6]	94.4, [92.4, 96.3]
LAP-ResNet	95.8, [94.0, 97.6]	**96.0**, [94.6, 97.3]

Table S7. – Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 50 points and 9 blocks (in %)

APPENDIX CHAPTER 5

C.1 Appendix 5.1: Unsupervised Image Reconstruction

C.1.1 Additional steps for handling the likelihood

We develop below the different steps for handling the likelihood summarized in Section 5.1.3.1:

1. Making use of the independence between X and Θ, we rewrite the expectation term $\mathbb{E}_{\mathrm{p}_Y} \log \mathrm{p}_{Y|X}(y|G(y))$ in equation (S4) to $\mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X \mathrm{p}_{Y|X,\Theta}} \log \mathrm{p}_{Y|X,\Theta}(y|G(y),\theta) + c_1$, with c_1 constant w.r.t. G:

   For all x, (in particular for $G(y)$), if X and Θ are independent, the log-likelihood can be decomposed as:

$$\begin{aligned} \log \mathrm{p}_{Y|X}(y|x) &= \log \mathrm{p}_{Y,\Theta|X}(y,\theta|x) - \log \mathrm{p}_{\Theta|X,Y}(\theta|x,y) \\ &\overset{X \perp\!\!\!\perp \Theta}{=} \log \mathrm{p}_{Y|X,\Theta}(y|x,\theta) + \log \mathrm{p}_\Theta(\theta) - \log \mathrm{p}_{\Theta|Y}(\theta|y) \end{aligned} \tag{S1}$$

   Applying the expectation *w.r.t* to the joint $\mathrm{p}_{Y,\Theta}$ on both sides, we obtain

$$\begin{aligned} \mathbb{E}_{\mathrm{p}_Y} \log \mathrm{p}_{Y|X}(y|x) &= \mathbb{E}_{\mathrm{p}_{Y,\Theta}} \big\{ \log \mathrm{p}_{Y|X,\Theta}(y|x,\theta) + \log \mathrm{p}_\Theta(\theta) - \log \mathrm{p}_{Y,\Theta}(\theta|y) \big\} \\ &= \mathbb{E}_{\mathrm{p}_{Y,\Theta}} \big\{ \log \mathrm{p}_{Y|X,\Theta}(y|x,\theta) \big\} + c_1 \end{aligned} \tag{S2}$$

   The terms $\log \mathrm{p}_\Theta(\theta)$ and $\log \mathrm{p}_{\Theta|Y}(\theta|y)$ do not depend on x, hence c_1 is a constant w.r.t. x. Plugging back $G(y)$ in place of x and applying the law of total probabilities w.r.t. X on the right hand side, we obtain:

$$\mathbb{E}_{\mathrm{p}_Y} \log \mathrm{p}_{Y|X}(y|G(y)) = \mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X \mathrm{p}_{Y|X,\Theta}} \log \mathrm{p}_{Y|X,\Theta}(y|G(y),\theta) + c_1 \tag{S3}$$

2. The general measurement process in equation (S1), $Y = F(X;\Theta) + \mathcal{E}$ induces $\log \mathrm{p}_{Y|X,\Theta}(y|G(y),\theta)$ to yield a simple analytic expression:

$$\log p(y|G(y),\theta) = -\frac{1}{2\sigma^2} \left\| y - F(G(y);\theta) \right\|_2^2 + c_2 \tag{S4}$$

with c_2 constant. This result is directly obtained using the fact that $\mathcal{E} \sim \mathcal{N}(0, \sigma^2 I)$.

3. The likelihood term $\mathbb{E}_{\mathrm{p}_Y} \log \mathrm{p}_{Y|X}(y|G(y))$ can then be replaced by $-\mathbb{E}_{\mathrm{p}_\Theta \mathrm{p}_X \mathrm{p}_{Y|X,\Theta}} \frac{1}{2\sigma^2} \|y - F(G(y$ in objective (S4). This is because the constant c_2 does not change the objective.

C.1.2 Architecture Details

Network architecture. Our network architectures are inspired by the Self-Attention GAN architecture in H. Zhang et al. 2018. They use residual networks (Kaiming He et al. 2016b), where each residual block of the generator and discriminator is comprised of 2 repeated sequences of batch normalization (Bjorck et al. 2018), ReLU activation, spectral normalization (Miyato et al. 2018b) and 3×3 convolutional layers. For the discriminator, we use the same as H. Zhang et al. 2018, and for reconstruction network G, we propose an image-to-image variant of their generator. We have not added downsampling layers: we have found that they degraded the overall model's performance. For corruption processes that yield observations that are very correlated with the input, such as Patch Band and Convolve-Noise, we have found that using $G(y) := y + \mathrm{Net}(y)$ for the reconstruction network allows us to initialize G close to identity, accelerates training and augments the overall quality of the samples.

Hyperparameters. Hyperparameters have been selected on the validation set, based on the mean square error between the reconstructions $\hat{x}$ and the image x. As in H. Zhang et al. 2018, we use imbalanced learning rates for the generator and the discriminator (0.0001 and 0.0004, respectively), using the Adam optimizer (Kingma et al. 2014), using $\beta_1 = 0$ and $\beta_2 = 0.9$. The weights are initialized using orthogonal initialization. We set $\lambda = 2$, and exponentially decay the learning rate every 400 iterations, setting the decay factor to 0.995.

C.1.3 Additional Information on the Baselines

C.1.3.1 Unpaired Variant

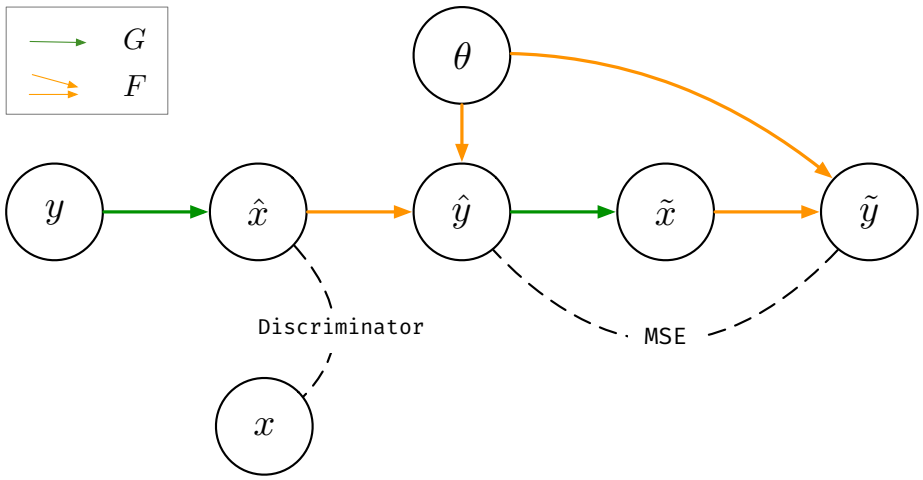


Figure S1. – Unpaired Variant of our model. As opposed to our model, this baseline has access to samples of the signal distribution p_X. This baseline is similar to our model, however, instead of discriminating between a measurement from the data y and a simulated measurement $\hat{y}$, we directly discriminate between samples from the signal distribution and the output of the reconstruction network $\hat{x}$.

C.1.3.2 Paired Variant

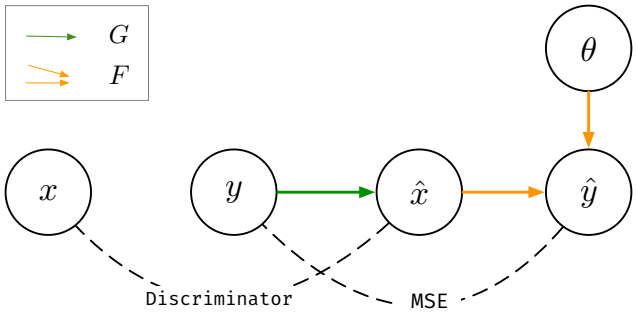


Figure S2. – Paired Variant of our model. As opposed to our model, this baseline not only has access to samples of the signal distribution p_X, but to signal measurement pairs (y, x) from the joint distribution $p_{Y,X}$. Given input measurement y, the reconstruction is obtained by regressing y to the associated signal x. In order to avoid blurry samples, we add add a adversarial term in the objective in order to enforce G to produce realistic samples, as in Isola et al. 2016b. The model is trained using the same architectures as the ones from our model.

C.1.3.3 Deep Image Prior (DIP)

Given an input measurement y, a generator G_ϕ parameterized by random parameters ϕ, and a random latent code z, the reconstruction $G_{\phi^*}(z)$ is obtained by resolving the following optimization problem:

$$\phi^* = \arg\min_\phi \|y - G_\phi(z)\|_2^2 \tag{S5}$$

For measurement processes Patch-Band, Remove-Pixel and Remove-Pixel-Channel (refer to Section 5.1.4.2), the resulting reconstruction $G_{\phi^*}(z)$ was not satisfactory: G was consistently regressing to the corrupted values in the measurement y, which led to unsatisfactory results. Instead of presenting these results, we have chosen to remove the contribution of the error terms where the measurement process induced null values from objective (S5), and present the latter instead. However, this assumes access to the true value θ that corrupted datum y: $y = F(x; \theta)$. Formally, we resolve the following objective:

$$\arg\min_\phi \|F(y - G_\phi(z); \theta)\|_2^2 \tag{S6}$$

Where F acts as a mask, and eliminates the terms associated to the pixels from y that have been put to 0. Note that this method corresponds to the inpainting

formulation in Ulyanov et al. 2017. We used the implementation provided by the authors [1].

C.1.4 Additional Samples

Figure S3. – Baseline comparison for the CelebA dataset. Corruption is Remove-Pixel ($p = 0.95$).

Figure S4. – Baseline comparison for the CelebA dataset. Corruption is Remove-Pixel-Channel ($p = 0.95$).

1. https://dmitryulyanov.github.io/deep_image_prior

Figure S5. – Baseline comparison for the CelebA dataset. Corruption is Convnoise($\sigma_C = 0.2$, $l = 3$).

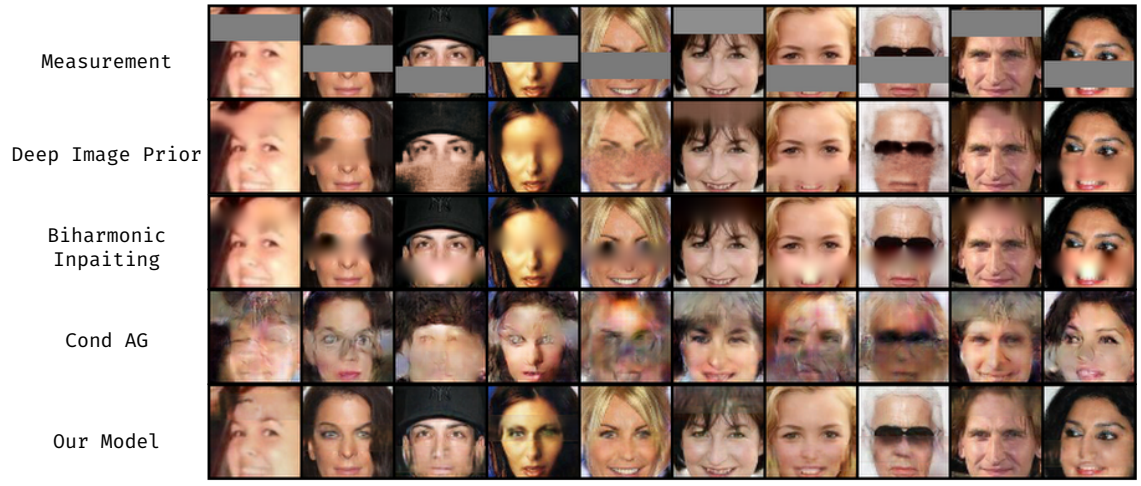


Figure S6. – Baseline comparison for the CelebA dataset. Corruption is PatchBand($h = 20$).

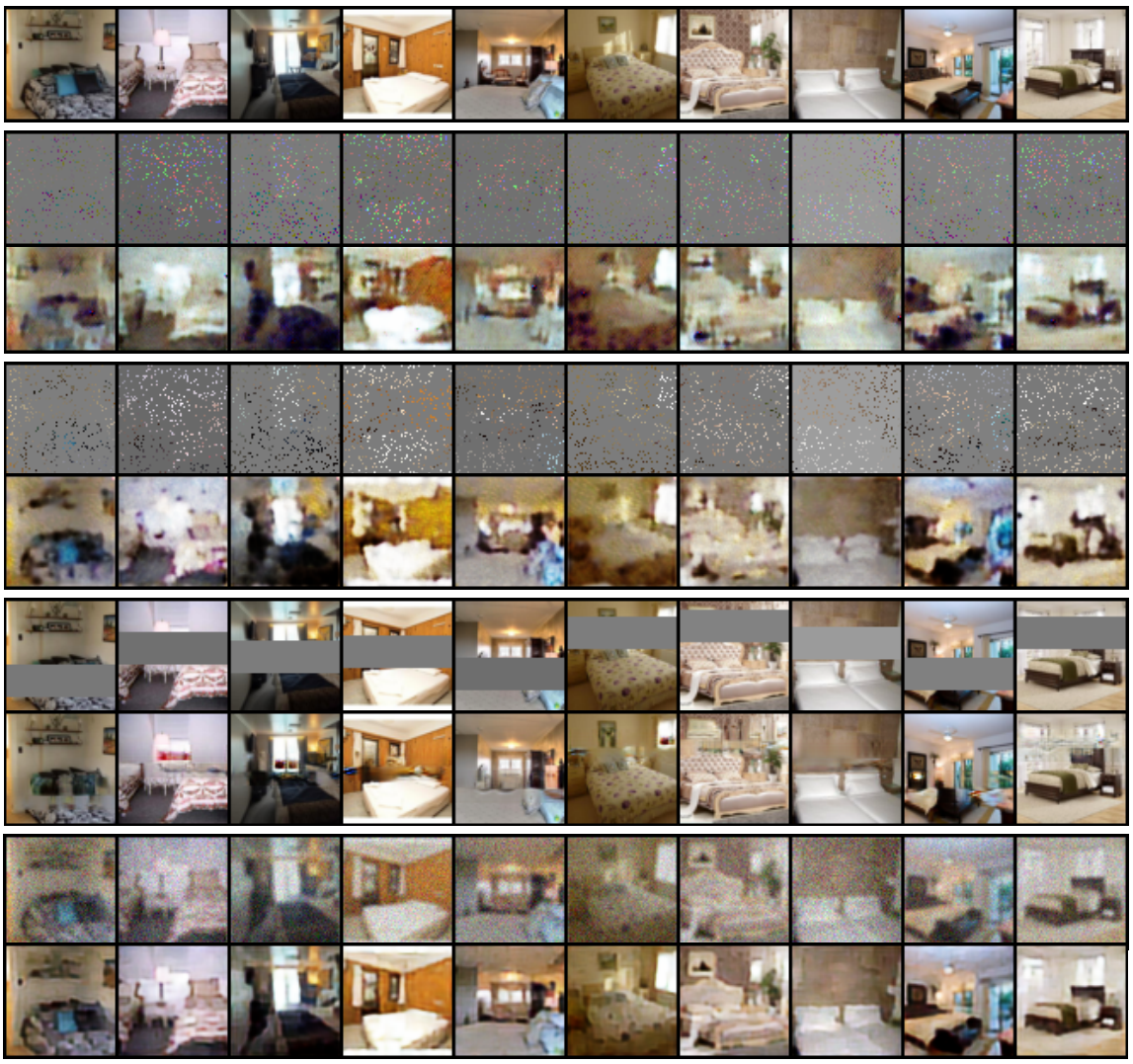


Figure S7. – On the top row, randomly sampled test set images from LSUN. Below, associated couples of corrupted observations and subsequent reconstructions from our model. From top to bottom, corruptions are Remove-Pixel-Channel($p = 0.95$), Remove-Pixel($p = 0.90$), Patch-Band($h = 20$), Convnoise($\sigma_C = 0.15$, $l = 3$).

Figure S8. – On the top row, randomly sampled test set images from Recipe. Below, associated couples of corrupted observations and subsequent reconstructions from our model. From top to bottom, corruptions are Remove-Pixel-Channel($p = 0.95$), Remove-Pixel($p = 0.90$), Patch-Band($h = 20$), Convnoise($\sigma_C = 0.15$, $l = 3$).

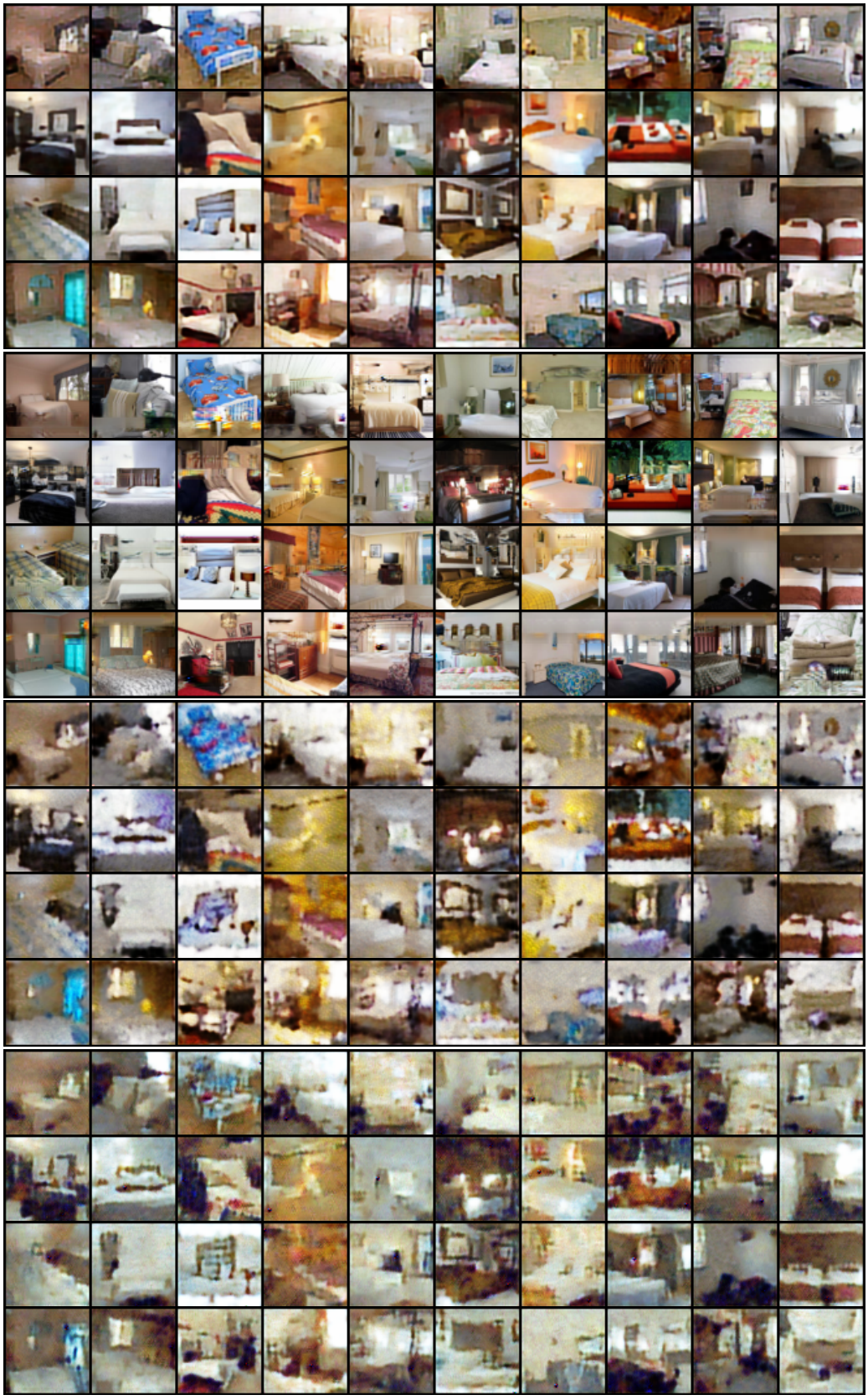


Figure S9. – Additional samples from our model of the LSUN Bedrooms dataset. From top to bottom, corruptions are Convnoise($\sigma_C = 0.15$, $l = 3$), Patch-Band($h = 20$), Remove-Pixel-Channel($p = 0.90$) and Remove-Pixel($p = 0.95$).

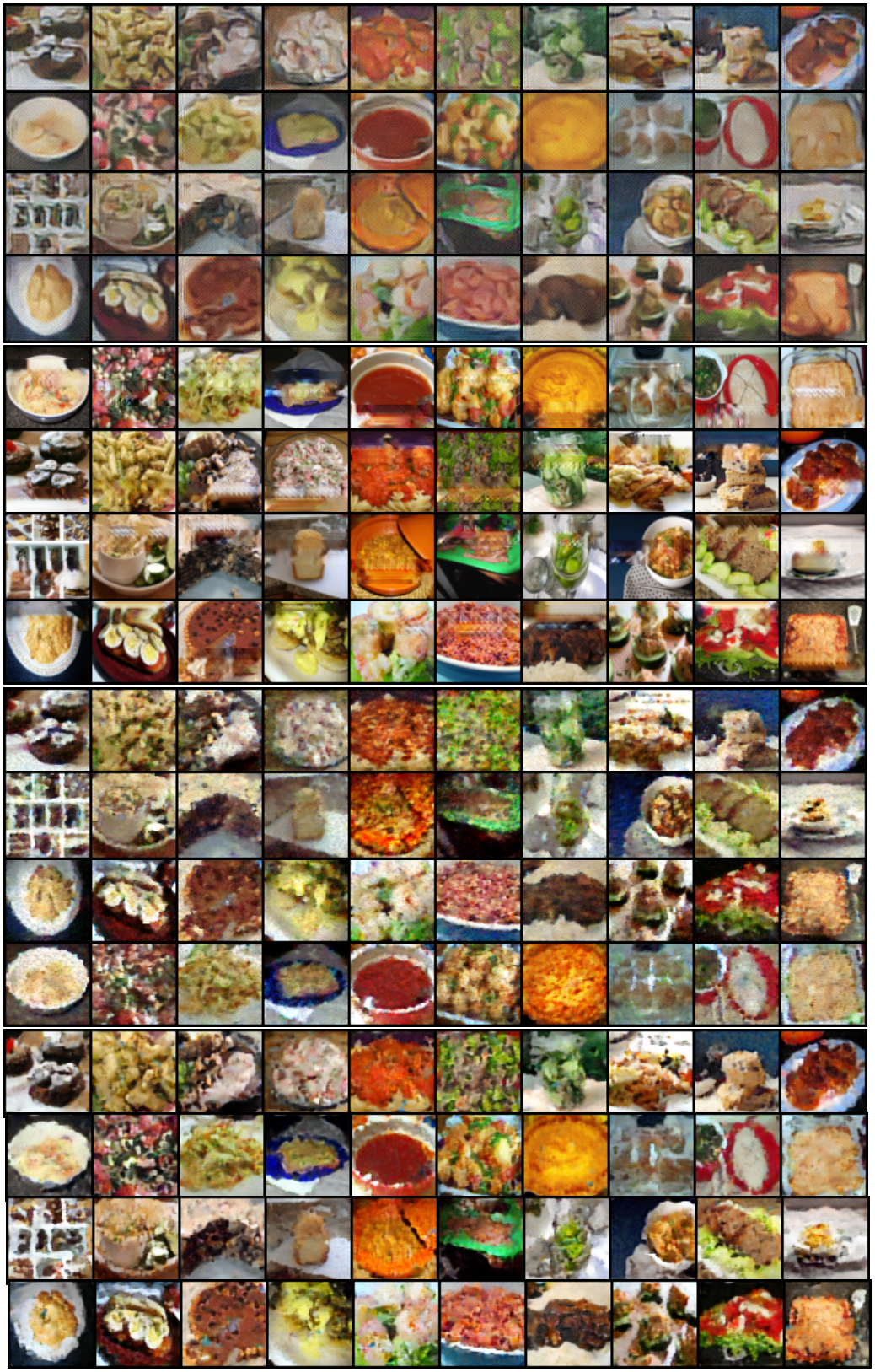


Figure S10. – Additional sample from our model, on the Recipe dataset. From top to bottom, corruptions are Convnoise($\sigma_C = 0.3$, $l = 5$), Patch-Band($h = 20$), Remove-Pixel-Channel($p = 0.90$) and Remove-Pixel($p = 0.90$).

C.2 Appendix 5.2: A Neural Tangent Kernel Perspective of GANs

C.2.1 Proofs of Theoretical Results and Additional Results

We prove in this section all theoretical results mentioned in Section 5.2.3 and 5.2.4. Section C.2.1.2 is devoted to the proof of Theorem S3, Appendix C.2.1.3 focuses on proving the differentiability results skimmed in Section 5.2.3.2, and Appendix C.2.1.4 and C.2.1.5 develop the results presented in Section 5.2.4.

C.2.1.1 Recall of Assumptions in the Paper

Assumption 1. $\hat{\gamma} \in \mathcal{P}(\Omega)$ is a finite mixture of Diracs.

Assumption 2. $k \colon \Omega^2 \to \mathcal{R}$ is a symmetric positive semi-definite kernel with $k \in L^2(\Omega^2)$.

Assumption 3. a and b from Equation (S13) are differentiable with Lipschitz derivatives over $\mathcal{R}$.

Assumption 4 (Discriminator architecture). The discriminator is a standard architecture (fully connected, convolutional or residual) with activations that are smooth everywhere except on a closed set D of null Lebesgue measure.

Assumption 5 (Discriminator regularity). $0 \notin D$, or linear layers have non-null bias terms.

C.2.1.2 On the Solutions of Equation (S20)

The methods used in this section are adaptations to our setting of standard methods of proof. In particular, they can be easily adapted to slightly different contexts, the main ingredient being the structure of the kernel integral operator. Moreover, it is also worth noting that, although we relied on Assumption 1 for $\hat{\gamma}$, the results are essentially unchanged if we take a compactly supported measure γ instead.

Let us first prove the following two intermediate lemmas.

Lemma S1. *Let $\delta T > 0$ and $\mathcal{F}_{\delta T} = \mathcal{C}\big([0, \delta T], B_{L^2(\hat{\gamma})}(f_0, 1)\big)$ endowed with the norm:*

$$\forall u \in \mathcal{F}_{\delta T}, \ \|u\| = \sup_{t \in [0, \delta T]} \|u_t\|_{L^2(\hat{\gamma})}. \tag{S7}$$

Then $\mathcal{F}_{\delta T}$ is complete.

Proof. Let $(u^n)_n$ be a Cauchy sequence in $\mathcal{F}_{\delta T}$. For a fixed $t \in [0, \delta T]$:

$$\forall n, m, \|u_t^n - u_t^m\|_{L^2(\hat{\gamma})} \leq \|u^n - u^m\|, \tag{S8}$$

which shows that $(u_t^n)_n$ is a Cauchy sequence in $L^2(\hat{\gamma})$. $L^2(\hat{\gamma})$ being complete, $(u_t^n)_n$ converges to a $u_t^\infty \in L^2(\hat{\gamma})$. Moreover, for $\epsilon > 0$, because (u^n) is Cauchy, we can choose N such that:

$$\forall n, m \geq N, \|u^n - u^m\| \leq \epsilon. \tag{S9}$$

We thus have that:

$$\forall t, \forall n, m \geq N, \|u_t^n - u_t^m\|_{L^2(\hat{\gamma})} \leq \epsilon. \tag{S10}$$

Then, by taking m to ∞, by continuity of the $L^2(\hat{\gamma})$ norm:

$$\forall t, \forall n \geq N, \|u_t^n - u_t^\infty\|_{L^2(\hat{\gamma})} \leq \epsilon, \tag{S11}$$

which means that:

$$\forall n \geq N, \|u^n - u^\infty\| \leq \epsilon. \tag{S12}$$

so that $(u^n)_n$ tends to u^∞.

Moreover, as:

$$\forall n, \|u_t^n\|_{L^2(\hat{\gamma})} \leq 1, \tag{S13}$$

we have that $\|u_t^\infty\|_{L^2(\hat{\gamma})} \leq 1$.

Finally, let us consider $s, t \in [0, \delta T]$. We have that:

$$\forall n, \|u_t^\infty - u_s^\infty\|_{L^2(\hat{\gamma})} \leq \|u_t^\infty - u_t^n\|_{L^2(\hat{\gamma})} + \|u_t^n - u_s^n\|_{L^2(\hat{\gamma})} + \|u_s^\infty - u_s^n\|_{L^2(\hat{\gamma})}. \tag{S14}$$

The first and the third terms can then be taken as small as needed by definition of u^∞ by taking n high enough, while the second can be made to tend to 0 as t tends to s by continuity of u^n. This proves the continuity of u^∞ and shows that $u^\infty \in \mathcal{F}_{\delta T}$. □

Lemma S2. *For any $F \in L^2(\hat{\gamma})$, we have that $F \in L^2(\hat{\alpha})$ and $F \in L^2(\hat{\beta})$ with:*

$$\|F\|_{L^2(\hat{\alpha})} \leq \sqrt{2}\|F\|_{L^2(\hat{\gamma})} \text{ and } \|F\|_{L^2(\hat{\beta})} \leq \sqrt{2}\|F\|_{L^2(\hat{\gamma})}. \tag{S15}$$

Proof. For any $F \in L^2(\hat{\gamma})$, we have that

$$\|F\|^2_{L^2(\hat{\gamma})} = \frac{1}{2}\|F\|^2_{L^2(\hat{\alpha})} + \frac{1}{2}\|F\|^2_{L^2(\hat{\beta})}, \tag{S16}$$

so that $F \in L^2(\hat{\alpha})$ and $F \in L^2(\hat{\beta})$ with:

$$\|F\|^2_{L^2(\hat{\alpha})} = 2\|F\|_{L^2(\hat{\gamma})} - \|F\|_{L^2(\hat{\beta})} \leq 2\|F\|^2_{L^2(\hat{\gamma})} \tag{S17}$$

$$\text{and } \|F\|^2_{L^2(\hat{\beta})} = 2\|F\|_{L^2(\hat{\gamma})} - \|F\|_{L^2(\hat{\alpha})} \leq 2\|F\|^2_{L^2(\hat{\gamma})}, \tag{S18}$$

which allows us to conclude. □

From this, we can prove the existence and uniqueness of the initial value problem from Equation (S20).

Theorem S1. *Under Assumptions 1 to 3, Equation (S20) with initial value f_0 admits a unique solution $f_\cdot : \mathcal{R}_+ \to L^2(\Omega)$.*

Proof.

A few inequalities. We start this proof by proving a few inequalities.

Let $f, g \in L^2(\hat{\gamma})$. We have, by the Cauchy-Schwarz inequality, for all $z \in \Omega$:

$$\left|\left(\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f)\right) - \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\right)\right)(z)\right| \leq \|k(z,\cdot)\|_{L^2(\hat{\gamma})}\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) - \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\|_{L^2(\hat{\gamma})}. \tag{S19}$$

Moreover, by definition:

$$\left\langle \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) - \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g), h\right\rangle_{L^2(\hat{\gamma})} = \int (a'_f - a'_g)h \,\mathrm{d}\hat{\alpha} - \int (b'_f - b'_g)h \,\mathrm{d}\hat{\beta}, \tag{S20}$$

so that:

$$\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) - \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\|^2_{L^2(\hat{\gamma})} \leq \|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) - \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\|_{L^2(\hat{\gamma})}(\|a'_f - a'_g\|_{L^2(\hat{\alpha})} + \|b'_f - b'_g\|_{L^2(\hat{\beta})}) \tag{S21}$$

and then, along with Lemma S2:

$$\begin{aligned}\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) - \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\|_{L^2(\hat{\gamma})} &\leq \|a'_f - a'_g\|_{L^2(\hat{\alpha})} + \|b'_f - b'_g\|_{L^2\left(\hat{\beta}\right)} \\ &\leq \sqrt{2}\left(\|a'_f - a'_g\|_{L^2(\hat{\gamma})} + \|b'_f - b'_g\|_{L^2(\hat{\gamma})}\right).\end{aligned} \tag{S22}$$

By Assumption 3, we know that a', b' are Lipschitz with constants that we denote K_1, K_2. We can then write:

$$\forall x,\ |a'(f(x)) - a'(g(x))| \leq K_1|f(x) - g(x)| \tag{S23}$$

$$\text{and } \forall x,\ |b'(f(x)) - b'(g(x))| \leq K_2|f(x) - g(x)|, \tag{S24}$$

so that:

$$\|a'_f - a'_g\|_{L^2(\hat{\gamma})} \leq K_1\|f - g\|_{L^2(\hat{\gamma})}, \qquad \|b'_f - b'_g\|_{L^2(\hat{\gamma})} \leq K_2\|f - g\|_{L^2(\hat{\gamma})}. \tag{S25}$$

Finally, we can now write, for all $z \in \Omega$:

$$\left|\left(\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f)\right) - \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\right)\right)(z)\right| \leq \sqrt{2}\,(K_1 + K_2)\,\|f-g\|_{L^2(\hat{\gamma})}\|k(z,\cdot)\|_{L^2(\hat{\gamma})}, \tag{A}$$

and then:

$$\left\|\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f)\right) - \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(g)\right)\right\|_{L^2(\hat{\gamma})} \leq K\|f-g\|_{L^2(\hat{\gamma})}, \tag{B}$$

where $K = \sqrt{2}\,(K_1+K_2)\sqrt{\int \|k(z,\cdot)\|^2_{L^2(\hat{\gamma})}\,\mathrm{d}\hat{\gamma}(z)}$ is finite as a finite sum of finite terms from Assumptions 1 and 2. In particular, putting $g = 0$ and using the triangular inequality also gives us:

$$\left\|\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f)\right)\right\|_{L^2(\hat{\gamma})} \leq K\|f\|_{L^2(\hat{\gamma})} + M, \tag{B'}$$

where $M = \left\|\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(0)\right)\right\|_{L^2(\hat{\gamma})}$.

Existence and uniqueness in $L^2(\hat{\gamma})$. We now adapt the standard fixed point proof to prove existence and uniqueness of a solution to the studied equation in $L^2(\hat{\gamma})$.

We consider the family of spaces $\mathcal{F}_{\delta T} = \mathcal{C}\left([0,\delta T], B_{L^2(\hat{\gamma})}(f_0,1)\right)$. $\mathcal{F}_{\delta T}$ is defined, for $\delta T > 0$, as the space of continuous functions from $[0,\delta T]$ to the closed ball of radius 1 centered around f_0 in $L^2(\hat{\gamma})$ which we endow with the norm:

$$\forall u \in \mathcal{F}_{\delta T},\ \|u\| = \sup_{t\in[0,\delta T]} \|u_t\|_{L^2(\hat{\gamma})}. \tag{S26}$$

We now define the application Φ where $\Phi(u)$ is defined as, for any $u \in \mathcal{F}_{\delta T}$:

$$\Phi(u)_t = f_0 + \int_0^t \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(u_s)\right)\mathrm{d}s. \tag{S27}$$

We have, using Equation (B'):

$$\|\Phi(u)_t - f_0\|_{L^2(\hat{\gamma})} \leq \int_0^t K\|u_s\|_{L^2(\hat{\gamma})} + M\,\mathrm{d}s \leq (K+M)\delta T. \tag{S28}$$

Thus, taking $\delta T = (2(K+M))^{-1}$ makes Φ an application from $\mathcal{F}_{\delta T}$ into itself. Moreover, we have:

$$\forall u, v \in \mathcal{F}_{\delta T}, \|\Phi(u) - \Phi(v)\| \leq \frac{1}{2}\|u-v\|, \tag{S29}$$

which means that Φ is a contraction of $\mathcal{F}_{\delta T}$. Lemma S1 and the Banach-Picard theorem then tell us that Φ has a unique fixed point in $\mathcal{F}_{\delta T}$. It is then obvious that such a fixed point is a solution of Equation (S20) over $[0, \delta T]$.

Let us now consider the maximal $T > 0$ such that a solution f_t of Equation (S20) is defined over $[0, T[$. We have, using Equation (B′):

$$\forall t \in [0, T[, \|f_t\|_{L^2(\hat{\gamma})} \leq \|f_0\|_{L^2(\hat{\gamma})} + \int_0^t (K\|f_s\|_{L^2(\hat{\gamma})} + M)\,\mathrm{d}s, \tag{S30}$$

which, using Gronwall's lemma, gives:

$$\forall t \in [0, T[, \|f_t\|_{L^2(\hat{\gamma})} \leq \|f_0\|_{L^2(\hat{\gamma})} \mathrm{e}^{KT} + \frac{M}{K}(\mathrm{e}^{KT} - 1). \tag{S31}$$

Define $g^n = f_{T-\frac{1}{n}}$. We have, again using Equation (B′):

$$\begin{aligned} \forall m \geq n, \|g^n - g^m\|_{L^2(\hat{\gamma})} &\leq \int_{T-\frac{1}{n}}^{T-\frac{1}{m}} (K\|f_s\| + M)\,\mathrm{d}s \\ &\leq \left(\frac{1}{n} - \frac{1}{m}\right)\left(\|f_0\|_{L^2(\hat{\gamma})}\mathrm{e}^{KT} + \frac{M}{K}(\mathrm{e}^{KT} - 1)\right). \end{aligned} \tag{S32}$$

which shows that $(g^n)_n$ is a Cauchy sequence. $L^2(\hat{\gamma})$ being complete, we can thus consider its limit g^∞. Obviously, f_t tends to g^∞ in $L^2(\hat{\gamma})$. By considering the initial value problem associated with Equation (S20) starting from g^∞, we can thus extend the solution f_t to $[0, T + \delta T[$ thus contradicting the maximality of T which proves that the solution can be extended to $\mathcal{R}_+$.

Existence and uniqueness in $L^2(\Omega)$. We now conclude the proof by extending the previous solution to $L^2(\Omega)$. We keep the same notations as above and, in particular, f is the unique solution of Equation (S20) with initial value f_0.

Let us define $\tilde{f}$ as:

$$\forall t, \forall x, \tilde{f}_t(x) = f_0(x) + \int_0^t \mathcal{T}_{k,\hat{\gamma}}\big(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)\big)(x)\,\mathrm{d}s, \tag{S33}$$

where the r.h.s. only depends on f and is thus well-defined. By remarking that $\tilde{f}$ is equal to f on $\mathrm{supp}(\hat{\gamma})$ and that, for every s,

$$\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}\left(\tilde{f}_s\right)\right) = \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}\left((\tilde{f}_s)_{|\,\mathrm{supp}(\hat{\gamma})}\right)\right) = \mathcal{T}_{k,\hat{\gamma}}\big(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)\big), \tag{S34}$$

we see that $\tilde{f}$ is solution to Equation (S20). Moreover, from Assumption 2, we know that, for any $z \in \Omega$, $\int k(z,x)^2 \,\mathrm{d}\Omega(x)$ is finite and, from Assumption 1, that $\|k(z,\cdot)\|^2_{L^2(\hat{\gamma})}$ is a finite sum of terms $k(z,x_i)^2$ which shows that $\int \|k(z,\cdot)\|^2_{L^2(\hat{\gamma})} \,\mathrm{d}\Omega(z)$ is finite, again from Assumption 2. We can then say that $\tilde{f}_s \in L^2(\Omega)$ for any s by using the above with Equation (A) taken for $g = 0$.

Finally, suppose h is a solution to Equation (S20) with initial value f_0. We know that $h_{|\operatorname{supp}(\hat{\gamma})}$ coincides with f and thus with $\tilde{f}_{|\operatorname{supp}(\hat{\gamma})}$ in $L^2(\hat{\gamma})$ as we already proved uniqueness in the latter space. Thus, we have that $\|(h_s)_{|\operatorname{supp}(\hat{\gamma})} - (\tilde{f}_s)_{|\operatorname{supp}(\hat{\gamma})}\|_{L^2(\hat{\gamma})} = 0$ for any s. Now, we have:

$$\begin{aligned}\forall s, \forall z \in \Omega, &\left|\left(\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(h_s)\right) - \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}\left(\tilde{f}_s\right)\right)\right)(z)\right| \\ &= \left|\left(\mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}\left((h_s)_{|\operatorname{supp}(\hat{\gamma})}\right)\right) - \mathcal{T}_{k,\hat{\gamma}}\left(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}\left((\tilde{f}_s)_{|\operatorname{supp}(\hat{\gamma})}\right)\right)\right)(z)\right| \\ &\leq 0\end{aligned}$$

by Equation (A). This shows that $\partial_t(\tilde{f} - h) = 0$ and, given that $h_0 = \tilde{f}_0 = f_0$, we have $h = \tilde{f}$ which conludes the proof.

□

There only remains to prove for Theorem S3 the inversion between the integral over time and the integral operator. We first prove an intermediate lemma and then conclude with the proof of the inversion.

Lemma S3. *Under Assumptions 1 to 3, $\int_0^T \left(\|a'\|_{L^2\left((f_s)_\sharp\hat{\alpha}\right)} + \|b'\|_{L^2\left((f_s)_\sharp\hat{\beta}\right)}\right) \mathrm{d}s$ is finite for any $T > 0$.*

Proof. Let $T > 0$. We have, by Assumption 3 and the triangular inequality:

$$\forall x, |a'(f(x))| \leq K_1|f(x)| + M_1, \tag{S35}$$

where $M_1 = |a'(0)|$. We can then write, using Lemma S2 and the inequality from Equation (S31):

$$\begin{aligned}\forall s \leq T, \|a'\|_{L^2\left((f_s)_\sharp\hat{\alpha}\right)} &\leq K_1\sqrt{2}\|f_s\|_{L^2(\hat{\gamma})} + M_1 \\ &\leq K_1\sqrt{2}\left(\|f_0\|_{L^2(\hat{\gamma})}\mathrm{e}^{KT} + \frac{M}{K}(\mathrm{e}^{KT} - 1)\right) + M_1,\end{aligned} \tag{S36}$$

the latter being constant in s and thus integrable on $[0,T]$. We can then bound $\|b'\|_{L^2\left((f_s)_\sharp\hat{\beta}\right)}$ similarly which concludes the proof. □

Proposition S22. *Under Assumptions 1 to 3, the following integral inversion holds:*

$$f_t = f_0 + \int_0^t \mathcal{T}_{k_f,\hat{\gamma}}\Big(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha},\hat{\beta}}(f_s)\Big)\,\mathrm{d}s = f_0 + \mathcal{T}_{k_f,\hat{\gamma}}\bigg(\int_0^t \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha},\hat{\beta}}(f_s)\,\mathrm{d}s\bigg). \tag{S37}$$

Proof. By definition, a straightforward computation gives, for any function $h \in L^2(\hat{\gamma})$:

$$\big\langle \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f), h\big\rangle_{L^2(\hat{\gamma})} = \mathrm{d}\mathcal{L}_{\hat{\alpha}}(f)[h] = \int a'_f h\,\mathrm{d}\hat{\alpha} - \int b'_f h\,\mathrm{d}\hat{\beta}. \tag{S38}$$

We can then write:

$$\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\|^2_{L^2(\hat{\gamma})} = \big\langle \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t), \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\big\rangle_{L^2(\hat{\gamma})} = \int a'_{f_t}\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\,\mathrm{d}\hat{\alpha} - \int b'_{f_t}\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\,\mathrm{d}\hat{\beta} \tag{S39}$$

so that, with the Cauchy-Schwarz inequality and Lemma S2:

$$\begin{aligned}
\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\|^2_{L^2(\hat{\gamma})} &\leq \int |a'_{f_t}||\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)|\,\mathrm{d}\hat{\alpha} + \int |b'_{f_t}||\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)|\,\mathrm{d}\hat{\beta} \\
&\leq \|a'_{f_t}\|_{L^2(\hat{\alpha})}\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\|_{L^2(\hat{\alpha})} + \|b'_{f_t}\|_{L^2(\hat{\beta})}\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\|_{L^2(\hat{\beta})} \\
&\leq \sqrt{2}\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\|_{L^2(\hat{\gamma})}\Big[\|a'_{f_t}\|_{L^2(\hat{\alpha})} + \|b'_{f_t}\|_{L^2(\hat{\beta})}\Big],
\end{aligned} \tag{S40}$$

which then gives us:

$$\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\|_{L^2(\hat{\gamma})} \leq \sqrt{2}\Big[\|a'\|_{L^2((f_t)_\sharp\hat{\alpha})} + \|b'\|_{L^2((f_t)_\sharp\hat{\beta})}\Big]. \tag{S41}$$

By the Cauchy-Schwarz inequality and Equation (S41), we then have for all z:

$$\begin{aligned}
\int_0^t \int_x |k(z,x)\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)(x)|\,\mathrm{d}\hat{\gamma}(x)\,\mathrm{d}s &\leq \int_0^t \|k(z,\cdot)\|_{L^2(\hat{\gamma})}\|\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)\|_{L^2(\hat{\gamma})}\,\mathrm{d}s \\
&\leq \sqrt{2}\|k(z,\cdot)\|_{L^2(\hat{\gamma})}\int_0^t \Big[\|a'\|_{L^2((f_s)_\sharp\hat{\alpha})} + \|b'\|_{L^2((f_s)_\sharp\hat{\beta})}\Big]\,\mathrm{d}s.
\end{aligned} \tag{S42}$$

The latter being finite by Lemma S3, we can now use Fubini's theorem to conclude that:

$$\begin{aligned}
\int_0^t \mathcal{T}_{k_f,\hat{\gamma}}\big(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)\big)\,\mathrm{d}s &= \int_0^t \int_x k(\cdot,x)\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)(x)\,\mathrm{d}\hat{\gamma}(x)\,\mathrm{d}s \\
&= \int_x k(\cdot,x)\bigg[\int_0^t \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)(x)\,\mathrm{d}s\bigg]\,\mathrm{d}\hat{\gamma}(x) \\
&= \mathcal{T}_{k_f,\hat{\gamma}}\bigg(\int_0^t \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_s)(x)\,\mathrm{d}s\bigg).
\end{aligned} \tag{S43}$$

□

C.2.1.3 Differentiability of Neural Tangent Kernels

Neural Tangent Kernels and the initialization of infinite-width functions being closely related to Gaussian Processes (GP), we first prove the following lemma showing the regularity of samples of a GP from the regularity of the corresponding kernel.

Lemma S4. *Let $A : \mathcal{R}^n \times \mathcal{R}^n \to \mathcal{R}$, a symmetric kernel. Let V an open set such that A is $\mathcal{C}^\infty$ on $V \times V$. Then the Gaussian Process induced by the kernel A has a.s. $\mathcal{C}^\infty$ sample paths on V.*

Proof. Because A is $\mathcal{C}^\infty$ on $V \times V$, we know, from Theorem 2.2.2 of Adler 1981 for example, that the corresponding GP f is mean-square smooth on V. If we take α a $k-$th order multi-index, we also know, again from Adler 1981, that $\partial^\alpha f$ is also a GP with covariance kernel $\partial^\alpha A$. As A is $\mathcal{C}^\infty$, $\partial^\alpha A$ then is differentiable and $\partial^\alpha f$ has partial derivatives which are mean-square continuous. Then, by the Corollary 5.3.12 of Scheuerer 2009, we can say that $\partial^\alpha f$ has continuous sample paths a.s. which means that $\partial^\alpha f \in C^k(V)$. This proves the lemma. □

We then tackle the differentiability of a key kernel in the theory of infinite-width neural networks Jacot et al. 2018c.

Lemma S5. *Let $A : \mathcal{R}^n \times \mathcal{R}^n \to \mathcal{R}$, a symmetric, positive semi-definite kernel and $\phi : \mathcal{R} \to \mathcal{R}$. Define:*

$$\forall x, y \in \mathcal{R}^n, B(x, y) = \mathbb{E}_{f \sim \mathcal{GP}(0,A)} \left[\phi(f(x))\phi(f(y))\right]. \tag{S44}$$

Moreover, suppose ϕ is $\mathcal{C}^\infty$ on an open set $O \subset \mathcal{R}$ such that $\mathcal{R} - O$ is of Lebesgue measure 0. If $0 \in O$ and A is $\mathcal{C}^\infty$ everywhere, then B is $\mathcal{C}^\infty$ everywhere. If $0 \notin O$, then for every neighbourhood V of points (x, y) such that $A(x, x) > 0 A(y, y) > 0$, if A is $\mathcal{C}^\infty$ on V, then B is $\mathcal{C}^\infty$ on V.

Proof. We have:

$$\forall x, y \in \mathcal{R}^n, B(x, y) = \mathbb{E}_{f \sim \mathcal{GP}(0,A)} \left[[\mathbf{1}_{f(x) \in O} + \mathbf{1}_{f(x) \notin O}]\phi(f(x))\phi(f(y))\right]. \tag{S45}$$

Let us now study, putting $\Sigma_A^{(x,y)} = \begin{pmatrix} A(x,x) & A(x,y) \\ A(x,y) & A(y,y) \end{pmatrix}$:

$$\mathbb{E}_{f \sim \mathcal{GP}(0,A)} \left[\mathbf{1}_{f(x) \notin O} \phi(f(x))\phi(f(y))\right] = \mathbb{E}_{(z,z') \sim \mathcal{N}(0, \Sigma_A^{(x,y)})} \left[\mathbf{1}_{z \notin O} \phi(z)\phi(z')\right]. \tag{S46}$$

As z is sampled from a Gaussian distribution with zero mean, even if it is degenerate, the only negligible set which can have a non null probability for z to be in, is $\{0\}$.

First case: $0 \in O$. In this case, because $\mathcal{R} - O$ is of null Lebesgue measure, we have $\mathbb{P}(z \notin O) = 0$ which means that the last expected value, which is the integral of a function a.s. null, is also of value 0. In other words, we have:

$$\forall x, y, B(x, y) = \mathbb{E}_{f \sim \mathcal{GP}(0,A)} \left[\mathbf{1}_{f(x) \in O} \phi(f(x)) \phi(f(y)) \right]. \tag{S47}$$

Moreover, by the same reasoning as above applied to y, we also have:

$$\forall x, y, B(x, y) = \mathbb{E}_{f \sim \mathcal{GP}(0,A)} \left[\mathbf{1}_{f(x) \in O} \mathbf{1}_{f(y) \in O} \phi(f(x)) \phi(f(y)) \right]. \tag{S48}$$

Let $x_0, y_0 \in \mathcal{R}^n$ and consider open neighbourhoods of both, V_1, V_2, with compact closures which we denote $cl(V_1)$ and $cl(V_2)$. Let us now consider a sample path f of the GP of kernel A. Lemma S4 then tells us that we can take f to be $\mathcal{C}^\infty$ in V_1 and V_2 with probability one. Let us also denote $V_1' = V_1 \cap f^{-1}(O)$ and $V_2' = V_2 \cap f^{-1}(O)$ which are open as O is open. In other words, $\phi \circ f$ is $\mathcal{C}^\infty$ on V_1' and V_2'.

Let $\alpha = (\alpha_1, ..., \alpha_n), \beta = (\beta_1, ..., \beta_n) \in \mathbb{N}^n$ such that $\sum_i \alpha_i \leq l$ and $\sum_i \beta_i \leq k$ for given k, l.

Using the usual notations for multi-indexed partial derivatives, via a multivariate Faà di Bruno formula Leipnik et al. 2007, we can write the derivative $\partial^\alpha(\phi \circ f)$ at $x \in V_1'$ as a sum of terms of the form:

$$\phi^{(j)}(f(x)) g_1(x) \cdots g_N(x), \tag{S49}$$

where the g_is are partial derivatives of f at x. As A is $\mathcal{C}^\infty$ everywhere, each of the g_is is thus a GP with a $\mathcal{C}^\infty$ covariance function. We can also write for all $x \in V_1'$:

$$\begin{aligned} |\phi^{(j)}(f(x)) g_1(x) \cdots g_N(x)| &\leq \sup_{z \in \mathrm{cl}(V_1)} |\phi^{(j)}(f(z)) g_1(z) \cdots g_N(z)| \\ &\leq \sup_{z_0 \in \mathrm{cl}(V_1)} |\phi^{(j)}(f(z_0))| \sup_{z_1 \in \mathrm{cl}(V_1)} |g_1(z_1)| \cdots \sup_{z_N \in \mathrm{cl}(V_1)} |g_N(z_N)|. \end{aligned} \tag{S50}$$

For each i, because the covariance function of g_i is smooth over $\mathrm{cl}(V_1)$, its variance admits a maximum at in $\mathrm{cl}(V_1)$ and we take σ_i^2 the double of its value. We then know Adler 1990, that there is an M_i such that:

$$\forall n, \mathbb{E}_{f \sim \mathcal{GP}(0,A)} \left[\sup_{z_i \in \mathrm{cl}(V_1)} |g_N(z_i)| \right]^n \leq M_i^n \mathbb{E}|Y_i|^n, \tag{S51}$$

where Y_i is a Gaussian distribution which variance is σ_i^2, the r.h.s. thus being finite.

Now, by using Cauchy-Schwarz, we have that:

$$\mathbb{E}\left[\sup_{z_0\in\mathrm{cl}(V_1)}|\phi^{(j)}(f(z_0))|\sup_{z_1\in\mathrm{cl}(V_1)}|g_1(z_1)|\cdots\sup_{z_N\in\mathrm{cl}(V_1)}|g_N(z_N)|\right]$$
$$\leq\sqrt{\mathbb{E}\left[(\sup_{z\in\mathrm{cl}(V_1)}\phi^{(j)}(f(z))|)^2\right]}\sqrt{\mathbb{E}\left[\sup_{z_1\in\mathrm{cl}(V_1)}|g_1(z_1)|^2\cdots\sup_{z_N\in\mathrm{cl}(V_1)}|g_N(z_N)|^2\right]}. \quad \text{(S52)}$$

By iterated applications of the Cauchy-Schwarz inequality and using the previous arguments, we can then show that $\sup_{z_0\in\mathrm{cl}(V_1)}|\phi^{(j)}(f(z_0))|\sup_{z_1\in\mathrm{cl}(V_1)}|g_1(z_1)|\cdots\sup_{z_N\in\mathrm{cl}(V_1)}|g_N(z_N)|$ is indeed integrable.

The same reasoning applies to $\partial^\beta(\phi\circ f)$ and we can then write, by a standard corollary of the dominated convergence theorem:

$$\partial^{\alpha,\beta}B(x,y)_{|(x_0,y_0)}=\mathbb{E}_{f\sim\mathcal{GP}(0,A)}\left[\partial^\alpha(\phi\circ f)_{|x_0}\partial^\beta(\phi\circ f)_{|y_0}\right], \quad \text{(S53)}$$

which shows that B is $\mathcal{C}^\infty$ on (x_0,y_0).

Second case: $0\notin O$. In this case, taking (x_0,y_0) such that $A(x_0,x_0)A(y_0,y_0)>0$, supposing A is $\mathcal{C}^\infty$ on a neighbourhood of (x_0,y_0) such that $A(x,x)A(y,y)>0$ on the neighbourhood, we have that $\mathbb{P}(f(x)\notin O \text{ or } f(y)\notin O)=0$ for any (x,y) in the neighbourhood as $A(x,x)>0$, $A(y,y)>0$ and O is of null Lebesgue measure. We can then prove that B is $\mathcal{C}^\infty$ on that same neighbourhood with the same reasoning as above. □

From this, we can prove the results skimmed in Section 5.2.3.2.

Proposition S23 (Prop. S10)**.** *Let k be the NTK of an architecture such as in Assumption 4. Then k is smooth on every (x,y) such that $x\neq 0$ and $y\neq 0$. Moreover, if we suppose the architecture verifies Assumption 5, then k is smooth everywhere.*

Proof. We define the following kernel:

$$\Sigma_L^\phi(x,y)=\mathbb{E}_{f\sim\mathcal{GP}(0,\Sigma_{L-1}^\phi)}[\phi(f(x))\phi(f(y))]+\beta^2, \quad \text{(S54)}$$

with:

$$\Sigma_0^\phi(x,y)=x^Ty+\beta^2. \quad \text{(S55)}$$

According to the definitions of Jacot et al. (2018c), Arora et al. (2019b) and Huang et al. (2020), the smoothness of the kernel, for an architecture of depth L is guaranteed whenever the kernels Σ_L^σ and $\Sigma_L^{\sigma'}$, where σ denotes in this proof the

activation function, are smooth. Note that, in the case of ResNets, there is a slight adaptation of the formula defining Σ_L which does not change its regularity.

Under Assumption 5, if there are non-null bias terms, we have that $\Sigma_L^\phi(x,x) \geq \beta^2 > 0$ for every x and $\phi \in \sigma, \sigma'$ so that, by a recursion using Lemma S5 and as Σ_0^ϕ is clearly smooth, we have the smoothness of Σ_L. The same is true if the activation is smooth on 0.

Now let us consider (x, y) such that $x \neq 0$ and $y \neq 0$. Then, either σ is constant everywhere, which automatically proves smoothness, or, for $\phi \in \sigma, \sigma'$, as $\Sigma_0^\phi(x,x) \geq x^T x > 0$ and $\Sigma_1^\phi(x,x) \geq \mathbb{E}_{z\sim N(0,\Sigma_0^\phi(x,x))}[\phi(z)^2] > 0$. We can continue this reasoning by recursion thus proving that $\Sigma_L^\phi(x,x) > 0$. The same applies for y and we can then use Lemma S5 to prove the desired result. □

Theorem S2 (Theorem S4)**.** *Let f_t be a solution to Equation (S20) under Assumptions 1 and 3 by Theorem S3, with k the NTK of a neural network and f_0 an initialization of the latter.*

Then, under Assumption 4, if $0 \notin \operatorname{supp} \hat{\gamma}$,[2] f_t is smooth on any point $x \neq 0$. Under Assumption 5, f_t is smooth everywhere.

Proof. We observe that $\mathcal{T}_{k,\hat{\gamma}}(g)$ has a regularity which only depends on the regularity of $k(\cdot, x)$ for $x \in \operatorname{supp} \hat{\gamma}$: if $k(\cdot, x)$ is smooth in a certain neighbourhood V for every such x, we can bound $\partial^\alpha k(\cdot, x)$ on V for every x and then use dominated convergence to prove that $\mathcal{T}_{k,\hat{\gamma}}(g)(\cdot)$ is smooth on V. The theorem then follows from the previous results and the fact that f_0 has the same regularity as Σ_L^σ defined in the proof of the last proposition, which is the same as k, as well as the fact that $f_t - f_0 = \mathcal{T}_{k,\hat{\gamma}}\left(\int_0^t \nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f_s)\, \mathrm{d}s\right)$. □

C.2.1.4 Optimality in Concave Setting

We derive an optimality result for concave bounded loss functions of the discriminator and positive definite kernels.

Assumptions

We first assume that the NTK is positive definite over the training dataset.

Assumption 6. k is positive definite over $\hat{\gamma}$.

2. Note that this is verified with probability 1 on the sampling of the dataset except if γ is concentrated on 0. Moreover, as all distributions are supposed to be compactly supported, they can always be shifted so that this is verified.

This positive definite property equates for finite datasets to the invertibility of the mapping

$$\begin{aligned} \mathcal{T}_{k,\hat{\gamma}}|_{\operatorname{supp}\hat{\gamma}} \colon L^2(\hat{\gamma}) &\to L^2(\hat{\gamma}) \\ h &\mapsto \mathcal{T}_{k,\hat{\gamma}}(h)|_{\operatorname{supp}\hat{\gamma}} \end{aligned}, \tag{S56}$$

that can be seen as a multiplication by the invertible Gram matrix of k over $\hat{\gamma}$. We further discuss this hypothesis in Appendix C.2.2.5.

We also assume the following properties on the discriminator loss function.

Assumption 7. $\mathcal{L}_{\hat{\alpha}}$ is concave and bounded from above, and its supremum is reached on a unique point $y^\star$ in $L^2(\hat{\gamma})$.

Moreover, we need for the sake of the proof a uniform continuity assumption on the solution to Equation (S20).

Assumption 8. $t \mapsto f_t|_{\operatorname{supp}\hat{\gamma}}$ is uniformly continuous over $\mathcal{R}_+$.

Note that these assumptions are verified in the case of LSGAN, which is the typical application of the optimality results that we prove in the following.

Optimality Result

Proposition S24 (Asymptotic optimality)**.** *Under Assumptions 1 to 3 and 6 to 8, f_t converges pointwise when $t \to \infty$, and:*

$$\mathcal{L}_{\hat{\alpha}}(f_t) \xrightarrow[t\to\infty]{} \mathcal{L}_{\hat{\alpha}}(y^\star), \quad f_\infty = f_0 + \mathcal{T}_{k,\hat{\gamma}}\Big(\mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\Big(y^\star - f_0|_{\operatorname{supp}\hat{\gamma}}\Big)\Big), \quad f_\infty|_{\operatorname{supp}\hat{\gamma}} = y^\star, \tag{S57}$$

where we recall that:

$$y^\star = \operatorname*{arg\,max}_{y \in L^2(\hat{\gamma})} \mathcal{L}_{\hat{\alpha}}(y). \tag{S58}$$

This result ensures that, for concave losses such as LSGAN, the optimum for $\mathcal{L}_{\hat{\alpha}}$ in $L^2(\Omega)$ is reached for infinite training times by neural network training in the infinite-width regime when the NTK of the discriminator is positive definite. However, this also provides the expression of the optimal network outside $\operatorname{supp}\hat{\gamma}$ thanks to the smoothing of $\hat{\gamma}$.

In order to prove this proposition, we need the following intermediate results: the first one about the functional gradient of $\mathcal{L}_{\hat{\alpha}}$ on the solution f_t; the second one about a direct application of positive definite kernels showing that one can retrieve $f \in \mathcal{H}_k^{\hat{\gamma}}$ over all Ω from its restriction to $\operatorname{supp}\hat{\gamma}$.

Lemma S6. *Under Assumptions 1 to 3 and 6 to 8, $\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t) \to 0$ when $t \to \infty$. Since $\operatorname{supp}\hat{\gamma}$ is finite, this limit can be interpreted pointwise.*

Proof. Assumptions 1 to 3 ensure the existence and uniqueness of f_t, by Theorem S3.

$t \mapsto \hat{f}_t \triangleq f_t|_{\operatorname{supp}\hat{\gamma}}$ and $\mathcal{L}_{\hat{\alpha}}$ being differentiable, $t \mapsto \mathcal{L}_{\hat{\alpha}}(f_t)$ is differentiable, and:

$$\frac{d\mathcal{L}_{\hat{\alpha}}(f_t)}{dt} = \left\langle \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t), \frac{d\hat{f}_t}{dt} \right\rangle_{L^2(\hat{\gamma})} = \left\langle \nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t), \mathcal{T}_{k,\hat{\gamma}}\big(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\big) \right\rangle_{L^2(\hat{\gamma})}, \tag{S59}$$

using Equation (S20). This equates to:

$$\frac{d\mathcal{L}_{\hat{\alpha}}(f_t)}{dt} = \left\| \mathcal{T}_{k,\hat{\gamma}}\big(\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t)\big) \right\|^2_{\mathcal{H}^{\hat{\gamma}}_k} \geq 0, \tag{S60}$$

where $\|\cdot\|_{\mathcal{H}^{\hat{\gamma}}_k}$ is the semi-norm associated to the RKHS $\mathcal{H}^{\hat{\gamma}}_k$. Note that this semi-norm is dependent on the restriction of its input to $\operatorname{supp}\hat{\gamma}$ only. Therefore, $t \mapsto \mathcal{L}_{\hat{\alpha}}(f_t)$ is increasing. Since $\mathcal{L}_{\hat{\alpha}}$ is bounded from above, $t \mapsto \mathcal{L}_{\hat{\alpha}}(f_t)$ admits a limit when $t \to \infty$.

We now aim at proving from the latter fact that $\frac{d\mathcal{L}_{\hat{\alpha}}(f_t)}{dt} \to 0$ when $t \to \infty$. We notice that $\|\cdot\|^2_{\mathcal{H}^{\hat{\gamma}}_k}$ is uniformly continuous over $L^2(\hat{\gamma})$ since $\operatorname{supp}\hat{\gamma}$ is finite, $\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}$ is uniformly continuous over $L^2(\hat{\gamma})$ since a' and b' are Lipschitz-continuous, $\mathcal{T}_{k,\hat{\gamma}}|_{\operatorname{supp}\hat{\gamma}}$ is uniformly continuous as it amounts to a finite matrix multiplication, and Assumption 8 gives that $t \mapsto f_t|_{\operatorname{supp}\hat{\gamma}}$ is uniformly continuous over $\mathcal{R}_+$. Therefore, their composition $t \mapsto \frac{d\mathcal{L}_{\hat{\alpha}}(f_t)}{dt}$ (from Equation (S60)) is uniformly continuous over $\mathcal{R}_+$. Using Barbălat's Lemma Farkas et al. 2016, we conclude that $\frac{d\mathcal{L}_{\hat{\alpha}}(f_t)}{dt} \to 0$ when $t \to \infty$.

Furthermore, k is positive definite over $\hat{\gamma}$ by Assumption 6, so $\|\cdot\|_{\mathcal{H}^{\hat{\gamma}}_k}$ is actually a norm. Therefore, since $\operatorname{supp}\hat{\gamma}$ is finite, the following pointwise convergence holds:

$$\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t) \xrightarrow[t\to\infty]{} 0. \tag{S61}$$

□

Lemma S7. *Under Assumptions 1, 2 and 6, for all $f \in \mathcal{H}^{\hat{\gamma}}_k$, the following holds:*

$$f = \mathcal{T}_{k,\hat{\gamma}}\left(\mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\left(f|_{\operatorname{supp}\hat{\gamma}}\right)\right) \tag{S62}$$

Proof. Since k is positive definite by Assumption 6, then $\mathcal{T}_{k,\hat{\gamma}}|_{\operatorname{supp}\hat{\gamma}}$ from Equation (S56) is invertible. Let $f \in \mathcal{H}^{\hat{\gamma}}_k$. Then, by definition of the RKHS in Theorem S2, there exists $h \in L^2(\hat{\gamma})$ such that $f = \mathcal{T}_{k,\hat{\gamma}}(h)$. In particular, $f|_{\operatorname{supp}\hat{\gamma}} = \mathcal{T}_{k,\hat{\gamma}}|_{\operatorname{supp}\hat{\gamma}}(h)$, hence $h = \mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\left(f|_{\operatorname{supp}\hat{\gamma}}\right)$. □

We can now prove the desired proposition.

Proof of Prop. S24. Let us first show that f_t converges to the optimum $y^\star$ in $L^2(\hat{\gamma})$. By applying Lemma S6, we know that $\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f_t) \to 0$ when $t \to \infty$. Given that the supremum of the differentiable concave function $\mathcal{L}_{\hat{\alpha}} \colon L^2(\hat{\gamma}) \to \mathcal{R}$ is achieved at a unique point $y^\star \in L^2(\hat{\gamma})$ with finite $\operatorname{supp}\hat{\gamma}$, then the latter convergence result implies that $\hat{f}_t \triangleq f_t|_{\operatorname{supp}\hat{\gamma}}$ converges pointwise to $y^\star$ when $t \to \infty$.

Given this convergence in $L^2(\hat{\gamma})$, we can deduce convergence on the whole domain Ω by noticing that $f_t - f_0 \in \mathcal{H}_k^{\hat{\gamma}}$, from Corollary 1. Thus, using Lemma S7:

$$f_t - f_0 = \mathcal{T}_{k,\hat{\gamma}}\Big(\mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\Big((f_t - f_0)|_{\operatorname{supp}\hat{\gamma}}\Big)\Big). \tag{S63}$$

Again, since $\operatorname{supp}\hat{\gamma}$ is finite, and $\mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}$ can be expressed as a matrix multiplication, the fact that f_t converges to $y^\star$ over $\operatorname{supp}\hat{\gamma}$ implies that:

$$\mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\Big((f_t - f_0)|_{\operatorname{supp}\hat{\gamma}}\Big) \xrightarrow[t\to\infty]{} \mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\Big(y^\star - f_0|_{\operatorname{supp}\hat{\gamma}}\Big). \tag{S64}$$

Finally, using the definition of the integral operator in Theorem S2, the latter convergence implies the following desired pointwise convergence:

$$f_t \xrightarrow[t\to\infty]{} f_0 + \mathcal{T}_{k,\hat{\gamma}}\Big(\mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp}\hat{\gamma}}\Big(y^\star - f_0|_{\operatorname{supp}\hat{\gamma}}\Big)\Big). \tag{S65}$$

We showed at the beginning of this proof that f_t converges to the optimum $y^\star$ in $L^2(\hat{\gamma})$, so $\mathcal{L}_{\hat{\alpha}}(f_t) \to \mathcal{L}_{\hat{\alpha}}(y^\star)$ by continuity of $\mathcal{L}_{\hat{\alpha}}$ as claimed in the proposition. □

C.2.1.5 Case Studies of Discriminator Dynamics

We study in the remaining of this section the expression of the discriminators in the case of the IPM loss and LSGAN, as described in Section 5.2.4, and of the original GAN formulation.

Preliminaries

We first need to introduce some definitions.

The presented solutions to Equation (S20) leverage a notion of functions of linear operators, similarly to functions of matrices Higham 2008. We define such functions in the simplified case of non-negative symmetric compact operators with a finite number of eigenvalues, such as $\mathcal{T}_{k,\hat{\gamma}}$.

Definition S3. Let $\mathcal{A} \colon L^2(\hat{\gamma}) \to L^2(\Omega)$ be a non-negative symmetric compact linear operator with a finite number of eigenvalues, for which the spectral theorem guarantees the existence of an countable othornormal basis of eigenfunctions with

non-negative eigenvalues. If $\varphi\colon \mathcal{R}_+ \to \mathcal{R}$, we define $\varphi(\mathcal{A})$ as the linear operator with the same eigenspaces as $\mathcal{A}$, with their respective eigenvalues mapped by φ; in other words, if λ is an eigenvalue of $\mathcal{A}$, then $\varphi(\mathcal{A})$ admits the eigenvalue $\varphi(\lambda)$ with the same eigenspace.

In the case where $\mathcal{A}$ is a matrix, this amounts to diagonalizing $\mathcal{A}$ and transforming its diagonalization elementwise using φ. Note that $\mathcal{T}_{k,\hat{\gamma}}$ has a finite number of eigenvalues since it is generated by a finite linear combination of linear operators (see Theorem S2).

We also need to defined the following Radon–Nikodym derivatives with inputs in $\operatorname{supp}\hat{\gamma}$:

$$\rho = \frac{d\left(\hat{\beta} - \hat{\alpha}\right)}{d\left(\hat{\beta} + \hat{\alpha}\right)}, \qquad \rho_1 = \frac{d\hat{\alpha}}{d\hat{\gamma}}, \qquad \rho_2 = \frac{d\hat{\beta}}{d\hat{\gamma}}, \tag{S66}$$

knowing that

$$\rho = \frac{1}{2}(\rho_2 - \rho_1), \qquad \rho_1 + \rho_2 = 2. \tag{S67}$$

These functions help us to compute the functional gradient of $\mathcal{L}_{\hat{\alpha}}$, as follows.

Lemma S8. *Under Assumption 3:*

$$\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) = \rho_1 a'_f - \rho_2 b'_f = \rho_1 \cdot (a' \circ f) - \rho_2 \cdot (b' \circ f). \tag{S68}$$

Proof. We have from Equation (S13):

$$\mathcal{L}_{\hat{\alpha}}(f) = \mathbb{E}_{x\sim\hat{\alpha}}[a_f(x)] - \mathbb{E}_{y\sim\hat{\beta}}[b_f(y)] = \langle \rho_1, a_f\rangle_{L^2(\hat{\gamma})} - \langle \rho_2, b_f\rangle_{L^2(\hat{\gamma})}, \tag{S69}$$

hence by composition:

$$\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) = \rho_1 \cdot (a' \circ f) - \rho_2 \cdot (b' \circ f) = \rho_1 a'_f - \rho_2 b'_f. \tag{S70}$$

□

LSGAN

Proposition S12. Under Assumptions 1 and 2, the solutions of Equation (S20) for $a = -(\mathrm{id}+1)^2$ and $b = -(\mathrm{id}-1)^2$ are the functions defined for all $t \in \mathcal{R}_+$ as:

$$f_t = \exp(-4t\mathcal{T}_{k,\hat{\gamma}})(f_0 - \rho) + \rho = f_0 + \varphi_t(\mathcal{T}_{k,\hat{\gamma}})(f_0 - \rho), \tag{S71}$$

where

$$\varphi_t \colon x \mapsto \mathrm{e}^{-4tx} - 1. \tag{S72}$$

Proof. Assumptions 1 and 2 are already assumed and Assumption 3 holds for the given a and b in LSGAN. Thus, Theorem S3 applies, and there exists a unique solution $t \mapsto f_t$ to Equation (S20) over $\mathcal{R}_+$ in $L^2(\Omega)$ for a given initial condition f_0. Therefore, there remains to prove that, for a given initial condition f_0,

$$g \colon t \mapsto g_t = f_0 + \varphi_t(\mathcal{T}_{k,\hat{\gamma}})(f_0 - \rho) \tag{S73}$$

is a solution to Equation (S20) with $g_0 = f_0$ and $g_t \in L^2(\Omega)$ for all $t \in \mathcal{R}_+$.

Let us first express the gradient of $\mathcal{L}_{\hat{\alpha}}$. We have from Lemma S8, with $a_f = -(f+1)^2$ and $b_f = -(f-1)^2$:

$$\nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f) = \rho_1 a'_f - \rho_2 b'_f = -2\rho_1(f+1) - 2\rho_2(f-1) = 4\rho - 4f. \tag{S74}$$

So Equation (S20) equates to:

$$\partial_t f_t = 4\mathcal{T}_{k,\hat{\gamma}}(\rho - f_t). \tag{S75}$$

Now let us prove that g_t is a solution to Equation (S75). We have:

$$\partial_t g_t = -4(\mathcal{T}_{k,\hat{\gamma}} \circ \exp(-4t\mathcal{T}_{k,\hat{\gamma}}))(f_0 - \rho) = -4(\mathcal{T}_{k,\hat{\gamma}} \circ \exp(-4t\mathcal{T}_{k,\hat{\gamma}}))(f_0 - \rho). \tag{S76}$$

Restricted to $\operatorname{supp} \hat{\gamma}$, we can write from Equation (S73):

$$g_t = f_0 + \left(\exp\left(-4t\,\mathcal{T}_{k,\hat{\gamma}}|_{\operatorname{supp}\hat{\gamma}}\right) - \mathrm{id}_{L^2(\hat{\gamma})}\right)(f_0 - \rho), \tag{S77}$$

and plugging this in Equation (S76):

$$\partial_t g_t = -4\mathcal{T}_{k,\hat{\gamma}}(g_t - \rho), \tag{S78}$$

where we retrieve the differential equation of Equation (S75). Therefore, g_t is a solution to Equation (S75).

It is clear that $g_0 = f_0$. Moreover, $\mathcal{T}_{k,\hat{\gamma}}$ being decomposable in a finite orthonormal basis of elements of operators over $L^2(\Omega)$, its exponential has values in $L^2(\Omega)$ as well, making g_t belong to $L^2(\Omega)$ for all t. With this, the proof is complete. □

IPMs

Proposition S11. Under Assumptions 1 and 2, the solutions of Equation (S20) for $a = b = \mathrm{id}$ are the functions of the form $f_t = f_0 + t f^*_{\hat{\alpha}}$, where $f^*_{\hat{\alpha}}$ is the unnormalized MMD witness function, yielding:

$$f^*_{\hat{\alpha}} = \mathbb{E}_{x\sim\hat{\alpha}}[k(x,\cdot)] - \mathbb{E}_{y\sim\hat{\beta}}[k(y,\cdot)], \qquad \mathcal{L}_{\hat{\alpha}}(f_t) = \mathcal{L}_{\hat{\alpha}}(f_0) + t \cdot \mathrm{MMD}^2_k\Big(\hat{\alpha}, \hat{\beta}\Big). \tag{S79}$$

Proof. Assumptions 1 and 2 are already assumed and Assumption 3 holds for the given a and b of the IPM loss. Thus, Theorem S3 applies, and there exists a unique solution $t \mapsto f_t$ to Equation (S20) over $\mathcal{R}_+$ in $L^2(\Omega)$ for a given initial condition f_0. Therefore, in order to find the solution of Equation (S20), there remains to prove that, for a given initial condition f_0,

$$g\colon t \mapsto g_t = f_0 + t f^*_{\hat{\alpha}} \tag{S80}$$

is a solution to Equation (S20) with $g_0 = f_0$ and $g_t \in L^2(\Omega)$ for all $t \in \mathcal{R}_+$.

Let us first express the gradient of $\mathcal{L}_{\hat{\alpha}}$. We have from Lemma S8, with $a_f = b_f = f$:

$$\nabla^{\hat{\gamma}} \mathcal{L}_{\hat{\alpha}}(f) = \rho_1 a'_f - \rho_2 b'_f = -2\rho. \tag{S81}$$

So Equation (S20) equates to:

$$\partial_t f_t = -2\mathcal{T}_{k,\hat{\gamma}}(\rho) = 2\int_x k(\cdot,x)\rho(x)\,\mathrm{d}\hat{\gamma}(x) = \int_x k(\cdot,x)\,\mathrm{d}\hat{\alpha}(x) - \int_y k(\cdot,y)\,\mathrm{d}\hat{\beta}(y), \tag{S82}$$

by definition of ρ (see Equation (S66)), yielding:

$$\partial_t f_t = f^*_{\hat{\alpha}}. \tag{S83}$$

Clearly, $t \mapsto g_t = f_0 + t f^*_{\hat{\alpha}}$ is a solution of the latter equation, $g_0 = f_0$ and $g_t \in L^2(\Omega)$ given that $\operatorname{supp}\hat{\gamma}$ is finite and $k \in L^2(\Omega^2)$ by assumption. The set of solutions for the IPM loss is thus characterized.

Finally, let us compute $\mathcal{L}_{\hat{\alpha}}(f_t)$. By linearity of $\mathcal{L}_{\hat{\alpha}}$ for $a = b = \mathrm{id}$:

$$\mathcal{L}_{\hat{\alpha}}(f_t) = \mathcal{L}_{\hat{\alpha}}(f_0) + t \cdot \mathcal{L}_{\hat{\alpha}}(f^*_{\hat{\alpha}}) = \mathcal{L}_{\hat{\alpha}}(f_0) + t \cdot \mathcal{L}_{\hat{\alpha}}(\mathcal{T}_{k,\hat{\gamma}}(-2\rho)). \tag{S84}$$

But, from Equation (S69), $\mathcal{L}_{\hat{\alpha}}(f) = \langle -2\rho, f\rangle_{L^2(\hat{\gamma})}$, hence:

$$\mathcal{L}_{\hat{\alpha}}(f_t) = \mathcal{L}_{\hat{\alpha}}(f_0) + t \cdot \langle -2\rho, \mathcal{T}_{k,\hat{\gamma}}(-2\rho)\rangle_{L^2(\hat{\gamma})} = \mathcal{L}_{\hat{\alpha}}(f_0) + t \cdot \|\mathcal{T}_{k,\hat{\gamma}}(-2\rho)\|^2_{\mathcal{H}^{\hat{\gamma}}_k}. \tag{S85}$$

By noticing that $\mathcal{T}_{k,\hat{\gamma}}(-2\rho) = f^*_{\hat{\alpha}}$ and that $\|f^*_{\hat{\alpha}}\|_{\mathcal{H}^{\hat{\gamma}}_k} = \mathrm{MMD}_k\left(\hat{\alpha}, \hat{\beta}\right)$ since $f^*_{\hat{\alpha}}$ is the unnormalized MMD witness function, the expression of $\mathcal{L}_{\hat{\alpha}}(f_t)$ in the proposition is obtained. □

Vanilla GAN

Unfortunately, finding the solutions to Equation (S20) in the case of the original GAN formulation, i.e. $a = \log(1-\sigma)$ and $b = -\log\sigma$, remains to the extent of our knowledge an open problem. We provide in the remaining of this section some leads that might prove useful for more advanced analyses.

Let us first determine the expression of Equation (S20) for vanilla GAN.

Lemma S9. *For $a = \log(1-\sigma)$ and $b = -\log\sigma$, Equation (S20) equates to:*

$$\partial_t f_t = \mathcal{T}_{k,\hat{\gamma}}(\rho_2 - 2\sigma(f)). \tag{S86}$$

Proof. We have from Lemma S8, with $a_f = b_f = f$:

$$\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) = \rho_1 a'_f - \rho_2 b'_f = -\rho_1 \frac{\sigma'(f)}{1-\sigma(f)} + \rho_2 \frac{\sigma'(f)}{\sigma(f)}. \tag{S87}$$

By noticing that $\sigma'(f) = \sigma(f)(1-\sigma(f))$, we obtain:

$$\nabla^{\hat{\gamma}}\mathcal{L}_{\hat{\alpha}}(f) = \rho_1 a'_f - \rho_2 b'_f = -\rho_1\sigma(f) + \rho_2(1-\sigma(f)) = \rho_2 - 2\sigma(f). \tag{S88}$$

By plugging the latter expression in Equation (S20), the desired result is achieved. □

Note that Assumption 3 holds for these choices of a and b. Therefore, under Assumptions 1 and 2, there exists a unique solution to Equation (S86) in $\mathcal{R}_+ \to L^2(\Omega)$ with a given initialization f_0.

Let us first study Equation (S86) in the simplified case of a one-dimensional ordinary differential equation.

Proposition S25. *Let $r \in \{0, 2\}$ and $\lambda \in \mathcal{R}$. The set of differentiable solutions over $\mathcal{R}$ to this ordinary differential equation:*

$$\frac{dy_t}{dt} = \lambda(r - 2\sigma(y_t)) \tag{S89}$$

is the following set:

$$S = \left\{y\colon t \mapsto (1-r)\left(W\left(\mathrm{e}^{2\lambda t + C}\right) - 2\lambda t - C\right) \;\middle|\; C \in \mathcal{R}\right\}, \tag{S90}$$

where W the is principal branch of the Lambert W function Rob M. Corless et al. 1996.

Proof. The theorem of Cauchy-Lipschitz ensures that there exists a unique global solution to Equation (S89) for a given initial condition $y_0 \in \mathcal{R}$. Therefore, we only need to show that all elements of S are solutions of Equation (S89) and that they can cover any initial condition.

Let us first prove that $y\colon t \mapsto (1-r)\big(W\big(\mathrm{e}^{2\lambda t+C}\big) - 2\lambda t - C\big)$ is a solution of Equation (S89). Let us express the derivative of y:

$$\frac{1}{1-r}\frac{dy_t}{dt} = 2\lambda\big(\mathrm{e}^{2\lambda t+C}W'\big(\mathrm{e}^{2\lambda t+C}\big) - 1\big). \tag{S91}$$

$W'(z) = \frac{W(z)}{z(1+W(z))}$, so:

$$\frac{1}{1-r}\frac{dy_t}{dt} = 2\lambda\left(\frac{W\big(\mathrm{e}^{2\lambda t+C}\big)}{1+W(\mathrm{e}^{2\lambda t+C})} - 1\right) = -\frac{2\lambda}{1+W(\mathrm{e}^{2\lambda t+C})}. \tag{S92}$$

Moreover, $W(z) = z\mathrm{e}^{-W(z)}$, and with $r-1 \in \{1,-1\}$:

$$\frac{1}{1-r}\frac{dy_t}{dt} = -\frac{2\lambda}{1+\mathrm{e}^{2\lambda t+C}\mathrm{e}^{-W\left(\mathrm{e}^{2\lambda t+C}\right)}} = -\frac{2\lambda}{1+\mathrm{e}^{(r-1)y_t}}. \tag{S93}$$

Finally, we notice that, since $r \in \{0,2\}$:

$$\lambda(r-2\sigma(y_t)) = -\frac{2\lambda(1-r)}{1+\mathrm{e}^{(r-1)y_t}}. \tag{S94}$$

Therefore:

$$\frac{dy_t}{dt} = \lambda(r-2\sigma(y_t)) \tag{S95}$$

and y_t is a solution to Equation (S89).

Since $y_0 = (1-r)\big(W\big(\mathrm{e}^C\big) - C\big)$ and $z \mapsto W(\mathrm{e}^z) - z$ can be proven to be bijective over $\mathcal{R}$, the elements of S can cover any initial condition. With this, the result is proved. □

Suppose that $f_0 = 0$ in Equation (S86) and that ρ_2 has values in $\{0,2\}$ – i.e. $\hat{\alpha}$ and $\hat{\beta}$ have disjoint supports (which is the typical case for distributions with finite support). From Prop. S25, a candidate solution would be:

$$f_t = \varphi_t(x)(\rho_2 - 1) = -\varphi_t(x)(\rho), \tag{S96}$$

where

$$\varphi_t\colon x \mapsto W\big(\mathrm{e}^{2tx+1}\big) - 2tx - 1, \tag{S97}$$

since the initial condition $y_0 = 0$ gives the constant value $C = 1$ in Equation (S90). Note that the Lambert W function of a symmetric linear operator is well-defined, all the more so as we choose the principal branch of the Lambert function in our case; see the work of Robert M. Corless et al. (2007) for more details. Note also that the estimation of $W(\mathrm{e}^z)$ is actually numerically stable using approximations from Iacono et al. (2017).

However, Equation (S96) cannot be a solution of Equation (S86). Indeed, one can prove by following essentially the same reasoning as the proof of Prop. S25 that:

$$\partial_t f_t = 2\big(\mathcal{T}_{k,\hat{\gamma}} \circ (\psi_t(\mathcal{T}_{k,\hat{\gamma}}))^{-1}\big)(\rho_2 - 1), \tag{S98}$$

with

$$\psi_t \colon x \mapsto 1 + W\big(\mathrm{e}^{2tx+1}\big) > 0. \tag{S99}$$

However, this does not allow us to obtain Equation (S86) since in the latter the sigmoid is taken coordinate-wise, where the exponential in Equation (S98) acts on matrices.

Nonetheless, for t small enough, f_t as defined in Equation (S98) should approximate the solution of Equation (S86), since sigmoid is approximately linear around 0 and $f_t \approx 0$ when t is small enough. We find in practice that for reasonable values of t, e.g. $t \leq 5$, the approximate solution of Equation (S98) is actually close to the numerical solution of Equation (S86) obtained using an ODE solver. Thus, we provide here an candidate approximate expression for the discriminator in the setting of the original GAN formulation – i.e., for binary classifiers. We leave for future work a more in-depth study of this case.

C.2.2 Discussions and Remarks

We develop in this section some remarks and explanations referenced in the main paper.

C.2.2.1 From Finite to Infinite-Width Networks

The constancy of the neural tangent kernel during training when the width of the network becomes increasingly large is broadly applicable. As summarized by C. Liu et al. (2020), typical neural networks with the building blocks of multilayer perceptrons and convolutional neural networks comply with this property, as long as they end with a linear layer and they do not have any bottleneck – indeed, this constancy needs the minimum internal width to grow unbounded Arora et al. 2019b. This includes, for example, residual convolutional neural networks Kaiming He et al. 2016c. The requirement of a final linear activation can be

circumvented by transferring this activation into the loss function, as we did for the original GAN formulation in Section 5.2.2. This makes our framework encompass a wide range of discriminator architectures.

Indeed, many building blocks of state-of-the-art discriminators can be studied in this infinite-width regime with a constant NTK, as highlighted by the exhaustiveness of the Neural Tangents library Roman Novak et al. 2020. Assumptions about the used activation functions are mild and include many standard activations such as ReLU, sigmoid and $\tanh$. Beyond fully connected linear layers and convolutions, typical operations such as self-attention Hron et al. 2020, layer normalization and batch normalization G. Yang 2020. This variety of networks affected by the constancy of the NTK supports the generality of our approach, as it includes powerful discriminator architectures such as BigGAN Brock et al. 2019.

There are nevertheless some limits to this approximation, as we are not aware of works studying the application of the infinite-width regime to some operations such as spectral normalization, and networks in the regime of a constant NTK cannot perform feature learning as they are equivalent to kernel methods Geiger et al. 2020; G. Yang et al. 2020. However, this framework remains general and constitutes the most advanced attempt at theoretically modeling the discriminator's architecture in GANs.

C.2.2.2 Loss of the Generator and its Gradient

We highlight in this section the importance of taking into account discriminator gradients in the optimization of the generator. Let us focus on an example similar to the one of Arjovsky et al. (2017b, Example 1) and choose as β a single Dirac centered at 0 and as $\alpha_g = \alpha_\theta$ single Dirac centered at $x_\theta = \theta$ (the generator parameters being the coordinates of the generated point). Let us focus for the sake of simplicity on the case of LSGAN since it is a recurring example in this work, but a similar reasoning can be done for other GAN instances.

In the theoretical min-max formulation of GANs considered by Arjovsky et al. (2017b), the generator is trained to minimize the following quantity:

$$\mathcal{C}_{f^\star_{\alpha_\theta}}(\alpha_\theta) \triangleq \mathbb{E}_{x\sim\alpha_\theta}\left[c_{f^\star_{\alpha_\theta}}(x)\right] = f^\star_{\alpha_\theta}(x_\theta)^2, \tag{S100}$$

where:

$$\begin{aligned} f^\star_{\alpha_\theta} &= \underset{f\in L^2\left(\frac{1}{2}\alpha_\theta+\frac{1}{2}\beta\right)}{\arg\max} \left\{\mathcal{L}_{\alpha_\theta}(f) \triangleq \mathbb{E}_{x\sim\alpha_\theta}[a_f(x)] - \mathbb{E}_{y\sim\beta}[b_f(y)]\right\} \\ &= \underset{f\in L^2\left(\frac{1}{2}\alpha_\theta+\frac{1}{2}\beta\right)}{\arg\min} \left\{\left(f^\star_{\alpha_\theta}(x_\theta)+1\right)^2 + \left(f^\star_{\alpha_\theta}(0)-1\right)^2\right\}. \end{aligned} \tag{S101}$$

Consequently, $f^\star_{\alpha_\theta}(0) = 1$ and $f^\star_{\alpha_\theta}(x_\theta) = -1$ when $x_\theta \neq 0$, thus in this case:

$$\mathcal{C}_{f^\star_{\alpha_\theta}}(\alpha_\theta) = 1. \tag{S102}$$

This constancy of the generator loss would make it impossible to be learned by gradient descent, as pointed out by Arjovsky et al. (2017b).

However, the setting does not correspond to the actual optimization process used in practice and represented by Equation (S14). We do have $\nabla_\theta \mathcal{C}_{f^\star_{\alpha_\theta}}(\alpha_\theta) = 0$ when $x_\theta \neq 0$, but the generator never uses this gradient in standard GAN optimization. Indeed, this gradient takes into account the dependency of the optimal discriminator $f^\star_{\alpha_\theta}$ in the generator parameters, since the optimal discriminator depends on the generated distribution. Yet, in practice and with few exceptions such as Unrolled GANs Metz et al. 2017 and as done in Equation (S14), this dependency is ignored when computing the gradient of the generator, because of the alternating optimization setting – where the discriminator is trained in-between generator's updates. Therefore, despite being constant on the training data, this loss can yield non-zero gradients to the generator. However, this requires the gradient of $f^\star_{\alpha_\theta}$ to be defined, which is the issue addressed in Section 5.2.2.2.

C.2.2.3 Differentiability of the Bias-Free ReLU Kernel

Theorem S4 contradicts the results of Bietti et al. (2019a) on the regularity of the NTK of a bias-free ReLU MLP with one hidden layer, which can be expressed as follows (up to a constant scaling the matrix multiplication in linear layers):

$$k(x, y) = \|x\|\|y\|\kappa\left(\frac{\langle x, y\rangle}{\|x\|\|y\|}\right), \tag{S103}$$

where

$$\begin{aligned} \kappa\colon [0, 1] &\to \mathcal{R} \\ u &\mapsto \frac{2}{\pi}u(\pi - \arccos u) + \frac{1}{\pi}\sqrt{1 - u^2} \end{aligned}. \tag{S104}$$

More particularly, Bietti et al. (2019a, Proposition 3) claim that $k(\cdot, y)$ is not Lipschitz on y for all y in the unit sphere. By following their proof, it amounts to prove that $k(\cdot, y)$ is not Lipschitz on y for all y in any centered sphere. This would imply that k is not differentiable for all inputs (x, y), which contradicts our result. We highlight that this also contradicts empirical evidence, as we did observe the Lipschitzness of such NTK in practice using the Neural Tangents library Roman Novak et al. 2020.

We believe that the mistake in the proof of Bietti et al. (2019a) lies in the confusion between functions κ and $k_0\colon x, y \mapsto \kappa\left(\frac{\langle x,y\rangle}{\|x\|\|y\|}\right)$, which have different

geometries. Their proof relies on the fact that κ is indeed non-Lipschitz in the neighborhood of $u = 1$. However, this does not imply that $k_0 \colon x, y \mapsto \kappa\left(\frac{\langle x,y\rangle}{\|x\|\|y\|}\right)$ is not Lipschitz, or not derivable. We can prove that it is actually at least locally Lipschitz.

Indeed, let us compute the following derivative for $x \neq y \in \mathcal{R}^n$, $x \neq 0$ and $y \neq 0$:

$$\frac{\mathrm{d}k_0(x,y)}{\mathrm{d}x} = \frac{y\|x\| - \frac{x}{\|x\|}\langle x,y\rangle}{\|x\|^2\|y\|}\kappa'(u) = \frac{1}{\|x\|\|y\|}\left(y - \langle x,y\rangle\frac{x}{\|x\|^2}\right)\kappa'(u), \tag{S105}$$

where $u = \frac{\langle x,y\rangle}{\|x\|\|y\|}$ and:

$$\pi \cdot \kappa'(u) = \frac{u}{\sqrt{1-u^2}} + 2(\pi - \arccos u). \tag{S106}$$

Note that $\kappa'(u) \sim_{u\to 1^-} \frac{\pi u}{\sqrt{1-u^2}} \sim_{u\to 1^-} \frac{\pi}{\sqrt{2}\sqrt{1-u}}$. Therefore:

$$\begin{aligned}
\frac{\pi}{\sqrt{2}} \cdot \frac{\mathrm{d}k_0(x,y)}{\mathrm{d}x} &\sim_{x\to y} \frac{1}{\|y\|^2}\left(y - \langle x,y\rangle\frac{x}{\|x\|^2}\right)\frac{\sqrt{\|x\|\|y\|}}{\sqrt{\|x\|\|y\| - \langle x,y\rangle}} \\
&\sim_{x\to y} \frac{\|x\|^2 y - \langle x,y\rangle x}{\|y\|^3\sqrt{\|x\|\|y\| - \langle x,y\rangle}} \\
&\sim_{x\to y} \frac{\|y\|^2 - \langle x,y\rangle}{\|y\|^3\sqrt{\|y\|^2 - \langle x,y\rangle}} y \xrightarrow[x\to y]{} 0,
\end{aligned} \tag{S107}$$

which proves that k_0 is actually Lipschitz around points (y, y), as well as differentiable, and confirms our result.

C.2.2.4 Integral Operator and Instance Noise

Instance noise Kaae Sønderby et al. 2017 consists in adding random Gaussian noise to the input and target samples. This amounts to convolving the data distributions with a Gaussian density, which will have the effect of smoothing the discriminator. In the following, for the case of IPM losses, we link instance noise with our framework, showing that smoothing of the data distributions already occurs via the NTK kernel, stemming from the fact that the discriminator is a neural network trained with gradient descent.

More specifically, it can be shown that if k is an RBF kernel, the optimal discriminators in both case are the same. This is based on the fact that the density of a convolution of an empirical measure $\hat{\mu} = \frac{1}{N}\sum_i \delta_{x_i}$, where δ_z is the Dirac distribution centered on z, and a Gaussian density $\tilde{k}$ with associated RBF kernel k can be written as $\tilde{k} * \hat{\mu} = \frac{1}{N}\sum_i k(x_i, \cdot)$.

Let us considered the following regularized discriminator optimization problem in $L^2(\mathcal{R})$ smoothed from $L^2(\Omega)$ with instance noise, i.e. convolving $\hat{\alpha}$ and $\hat{\beta}$ with $\tilde{k}$.

$$\sup_{f \in L^2(\mathcal{R})} \left\{ \mathcal{L}^{\tilde{k}}_{\hat{\alpha}}(f) \triangleq \mathbb{E}_{x \sim \tilde{k} * \hat{\alpha}}[f(x)] - \mathbb{E}_{y \sim \tilde{k} * \hat{\beta}}[f(y)] - \lambda \left\| f \right\|^2_{L^2} \right\} \tag{S108}$$

The optimum f^{IN} can be found by taking the gradient:

$$\nabla_f \left(\mathcal{L}^{\tilde{k}}_{\hat{\alpha}} f^{\text{IN}} - \lambda \left\| f^{\text{IN}} \right\|^2_{L^2} \right) = 0 \qquad \Leftrightarrow \qquad f^{\text{IN}} = \frac{1}{2\lambda} \left(\tilde{k} * \hat{\alpha} - \tilde{k} * \hat{\beta} \right). \tag{S109}$$

If we now study the resolution of the optimization problem in $\mathcal{H}^{\hat{\gamma}}_k$ as in Section 5.2.4.1 with $f_0 = 0$, we find the following discriminator:

$$f_t = t \left(\mathbb{E}_{x \sim \hat{\alpha}}[k(x, \cdot)] - \mathbb{E}_{y \sim \hat{\beta}}[k(y, \cdot)] \right) = t \left(\tilde{k} * \hat{\alpha} - \tilde{k} * \hat{\beta} \right). \tag{S110}$$

Therefore, we have that $f^{\text{IN}} \propto f_t$, i.e. instance noise and regularization by neural networks obtain the same smoothed solution.

This analysis was done using the example of an RBF kernel, but it also holds for stationary kernels, i.e. $k(x, y) = \tilde{k}(x - y)$, which can be used to convolve measures. We remind that this is relevant, given that NTKs are stationary over spheres Jacot et al. 2018c; G. Yang et al. 2019, around where data can be concentrated in high dimensions.

C.2.2.5 Positive Definite NTKs

Optimality results in the theory of NTKs usually rely on the assumption that the considered NTK k is positive definite over the training dataset $\hat{\gamma}$ Jacot et al. 2018c; Yaoyu Zhang et al. 2020. This property offers several theoretical advantages.

Indeed, this gives sufficient representational power to its RKHS to include the optimal solution over $\hat{\gamma}$. Moreover, this positive definite property equates for finite datasets to the invertibility of the mapping

$$\begin{aligned} \mathcal{T}_{k,\hat{\gamma}}|_{\operatorname{supp} \hat{\gamma}} \colon L^2(\hat{\gamma}) &\to L^2(\hat{\gamma}) \\ h &\mapsto \mathcal{T}_{k,\hat{\gamma}}(h)|_{\operatorname{supp} \hat{\gamma}} \end{aligned}, \tag{S111}$$

that can be seen as a multiplication by the invertible Gram matrix of k over $\hat{\gamma}$. From this, one can retrieve the expression of $f \in \mathcal{H}^{\hat{\gamma}}_k$ from its restriction $f|_{\operatorname{supp} \hat{\gamma}}$ to $\operatorname{supp} \hat{\gamma}$ in the following way:

$$f = \mathcal{T}_{k,\hat{\gamma}} \circ \mathcal{T}_{k,\hat{\gamma}}|^{-1}_{\operatorname{supp} \hat{\gamma}} \left(f|_{\operatorname{supp} \hat{\gamma}} \right), \tag{S112}$$

as shown in Lemma S7. Finally, as shown by Jacot et al. (2018c) and in Appendix C.2.1.4, this makes the discriminator loss function strictly increase during training.

One may wonder whether this assumption is reasonable for NTKs. Jacot et al. (2018c) proved that it indeed holds for NTKs of non-shallow MLPs with non-polynomial activations if data is supported on the unit sphere, supported by the fact that the NTK is stationary over the unit sphere. Others, such as Fan et al. (2020), have observed positive definiteness of the NTK subject to specific asumptions on the networks and data. We are not aware of more general results of this kind. However, one may conjecture that, at least for specific kind of networks, NTKs are positive definite for any training data.

Indeed, besides global convergence results Allen-Zhu et al. 2019, prior work indicate that MLPs are universal approximators Hornik et al. 1989; Leshno et al. 1993. This property can be linked in our context to universal kernels Steinwart 2001, which are guaranteed to be positive definite over any training data Sriperumbudur et al. 2011. Universality is linked to the density of the kernel RKHS in the space of continuous functions. In the case of NTKs, previously cited approximation properties can be interpreted as signs of expressive RKHSs, and thus support the hypothesis of universal NTKs. Furthermore, beyond positive definiteness, universal kernels are also characteristic Sriperumbudur et al. 2011, which is interesting when they are used to compute MMDs, as we do in Section 5.2.4.1. Note that for the standard case of ReLU MLPs, Ji et al. (2020) showed universal approximation results in the infinite-width regime, and works such as the one of L. Chen et al. (2021) observed that their RKHS is close to the one of the Laplace kernel, which is positive definite.

Bias-Free ReLU NTKs are not Characteristic. As already noted by Leshno et al. (1993), the presence of bias is important when it comes to representational power of MLPs. We can retrieve this observation in our framework. In the case of a ReLU shallow network with one hidden layer and without bias, Bietti et al. (2019a) determine its associated NTK as follows (up to a constant scaling the matrix multiplication in linear layers):

$$k(x,y) = \|x\|\|y\|\kappa\left(\frac{\langle x,y\rangle}{\|x\|\|y\|}\right), \tag{S113}$$

with in particular $k(x,0) = 0$ for all $x \in \Omega$; suppose that $0 \in \Omega$. This expression of the kernel implies that k is not positive definite for all datasets: take for example $x = 0$ and $y \in \Omega \setminus \{0\}$; then the Gram matrix of k has a null row, hence k is not strictly positive definite over $\{x, y\}$. Another consequence is that k is not characteristic. Indeed, take probability distributions $\mu = \delta_{\frac{y}{2}}$ and $\nu = \frac{1}{2}(\delta_x + \delta_y)$

Table S1. – Sinkhorn divergence averaged over three runs between the final generated distribution and the target dataset for the Density problem.

Loss	RBF kernel	ReLU	ReLU (no bias)	Sigmoid
IPM (inf.)	$(2.37 \pm 0.32) \times 10^{-3}$	$(3.34 \pm 0.49) \times 10^{-9}$	$(7.34 \pm 0.34) \times 10^{-2}$	$(6.25 \pm 0.31) \times 10^{-3}$
IPM	—	$(5.02 \pm 1.19) \times 10^{-3}$	$(9.25 \pm 0.30) \times 10^{-2}$	$(3.06 \pm 0.57) \times 10^{-2}$
LSGAN (inf.)	$(7.53 \pm 0.59) \times 10^{-3}$	$(1.49 \pm 0.11) \times 10^{-3}$	$(2.80 \pm 0.03) \times 10^{-1}$	$(2.21 \pm 0.01) \times 10^{-1}$
LSGAN	—	$(1.53 \pm 1.08) \times 10^{-2}$	$(1.64 \pm 0.19) \times 10^{-1}$	$(5.88 \pm 0.80) \times 10^{-2}$

with δ_z being the Dirac distribution centered on $z \in \Omega$, and where $x = 0$ and $y \in \Omega \setminus \{0\}$. Then:

$$\mathbb{E}_{z\sim\mu} k(z, \cdot) = k\left(\frac{1}{2}y, \cdot\right) = \frac{1}{2}k(y, \cdot) = \frac{1}{2}(k(y, \cdot) + k(x, \cdot)) = \mathbb{E}_{z\sim\nu} k(z, \cdot), \tag{S114}$$

i.e., kernel embeddings of μ and $\nu \neq \mu$ are identical, making k not characteristic by definition.

C.2.3 GAN(TK)^2 and Further Empirical Analysis

We present in this section additional experimental results that complement and explain some of the results already exposed in Section 5.3.4. All these experiments were conducted using the proposed general toolkit GAN(TK)^2.

We focus in this article on particular experiments for the sake of clarity and as an illustration of the potential of analysis of our framework, but GAN(TK)^2 is a general-purpose toolkit centered around the infinite-width of the discriminator and could be leveraged for an even more extensive empirical analysis. We specifically focused on the IPM and LSGAN losses for the discriminator since they are the two losses for which we know the analytic behavior of the discriminator in the infinite-width limit, but other losses can be studied as well in GAN(TK)^2. We leave a large-scale empirical study of our framework, which is out of the scope of this paper, for future work.

C.2.3.1 Other Two-Dimensional Datasets

We present additional experimental results on two other two-dimensional problems, Density and AB; see, respectively, Figures S11 and S12. Numerical results are detailed in Tables S1 and S2. We globally retrieve the same conclusions that we developed in Section 5.3.4 on this datasets with more complex shapes.

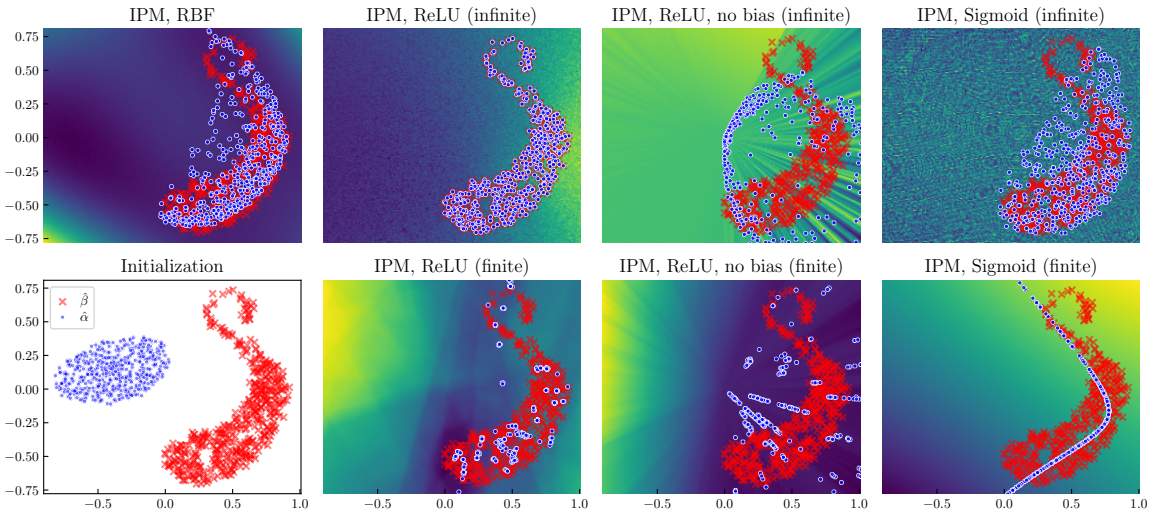


Figure S11. – Generator (●) and target (×) samples for different methods applied to the Density problem. In the background, $c_{f^\star}$.

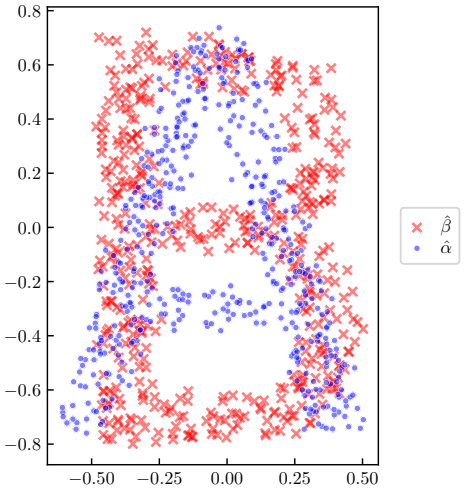


Figure S12. – Initial generator (●) and target (×) samples for the AB problem.

Table S2. – Sinkhorn divergence averaged over three runs between the final generated distribution and the target dataset for the AB problem.

Loss	RBF kernel	ReLU	ReLU (no bias)	Sigmoid
IPM (inf.)	$(4.65 \pm 0.82) \times 10^{-3}$	$(2.64 \pm 2.13) \times 10^{-9}$	$(6.11 \pm 0.19) \times 10^{-3}$	$(5.69 \pm 0.38) \times 10^{-3}$
IPM	—	$(2.75 \pm 0.20) \times 10^{-3}$	$(3.65 \pm 1.44) \times 10^{-2}$	$(1.25 \pm 0.32) \times 10^{-2}$
LSGAN (inf.)	$(1.13 \pm 0.05) \times 10^{-2}$	$(8.63 \pm 2.24) \times 10^{-3}$	$(1.02 \pm 0.40) \times 10^{-1}$	$(1.40 \pm 0.06) \times 10^{-2}$
LSGAN	—	$(1.32 \pm 1.30) \times 10^{-1}$	$(2.57 \pm 0.73) \times 10^{-2}$	$(8.78 \pm 2.23) \times 10^{-2}$

Table S3. – Sinkhorn divergence averaged over three runs between the final generated distribution and the target dataset for the 8 Gaussians problem.

Loss	RBF kernel	ReLU	ReLU (no bias)	Sigmoid
IPM (inf.)	$(2.60 \pm 0.06) \times 10^{-2}$	$(9.40 \pm 2.71) \times 10^{-7}$	$(9.70 \pm 1.88) \times 10^{-2}$	$(8.40 \pm 0.02) \times 10^{-2}$
IPM	—	$(1.21 \pm 0.14) \times 10^{-1}$	1.20 ± 0.60	$(7.40 \pm 1.30) \times 10^{-1}$
LSGAN (inf.)	$(4.21 \pm 0.10) \times 10^{-1}$	$(7.56 \pm 0.45) \times 10^{-2}$	$(1.27 \pm 0.01) \times 10^{1}$	7.35 ± 0.11
LSGAN	—	3.07 ± 0.68	7.52 ± 0.01	7.41 ± 0.54

C.2.3.2 ReLU vs. Sigmoid Activations

We additionally introduce a new baseline for the 8 Gaussians, Density and AB problems, where we replace the ReLU activation in the discriminator by a sigmoid. Results are given in Tables S1 to S3 and an illustration is available in Figure S11.

We observe that the sigmoid baseline is consistently outperformed by the RBF kernel and ReLU activation (with bias) for all regimes and losses. This is in accordance with common experimental practice, where internal sigmoid activations are found less effective than ReLU because of the potential activation saturation that they can induce.

We provide a qualitative explanation to this underperformance of sigmoid via our framework in Section C.2.3.4.

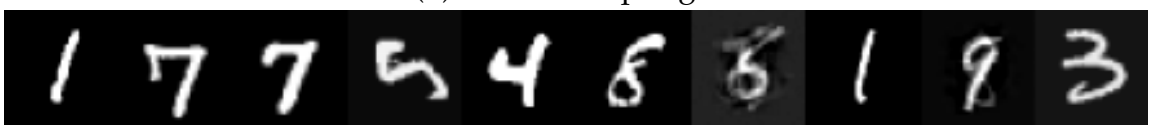
C.2.3.3 Qualitative MNIST Experiment

An experimental analysis of our framework on complex image datasets is out the scope of our study – we leave it for future work. Nonetheless, we present an experiment on MNIST images LeCun et al. 1998 in a similar setting as the experiments on two-dimensional point clouds of the previous sections. We make a point cloud $\hat{\alpha}$, initialized to a standard Gaussian, move towards a subset of the MNIST dataset following the gradients of the IPM loss in the infinite-width regime. Qualitative results are presented in Figure S13.

We notice, similarly to the two-dimensional datasets, that the ReLU network with bias outperforms its bias-free counterpart and a standard RBF kernel in terms of sample quality. The difference between the RBF kernel and ReLU NTK is even more flagrant in this complex high-dimensional setting, as the RBF kernel is unable to produce accurate samples.

(a) RBF kernel: blurry digits.

(b) ReLU: sharp digits.



(c) ReLU (no bias): mostly sharp digits with some artifacts and blurry images.

Figure S13. – Uncurated samples from the results of the descent of a set of 1024 particules over a subset of 1024 elements of MNIST, starting from a standard Gaussian. Training is done using the IPM loss in the infinite-width kernel setting.

C.2.3.4 Visualizing the Gradient Field Induced by the Discriminator

We raise in Section 5.2.4 the open problem of studying the convergence of the generated distribution towards the target distribution with respect to the gradients of the discriminator. We aim in this section at qualitatively studying these gradients in a simplified case that could shed some light on the more general setting and explain some of our experimental results. These gradient fields can plotted using the provided GAN(TK)² toolkit.

Setting

Since we study gradients of the discriminator expressed in Equation (S21), we assume that $f_0 = 0$ – for instance, using the anti-symmetrical initialization Yaoyu Zhang et al. 2020 – in order to ignore residual gradients from the initialization.

By Theorem S3, for any loss and any training time, the discriminator can be expressed as $f^\star_{\hat\alpha} = \mathcal{T}_{k,\hat\gamma}(h_0)$, for some $h_0 \in L^2(\hat\gamma)$. Thus, there exists $h_1 \in L^2(\hat\gamma)$ such that:

$$f^\star_{\hat\alpha} = \sum_{x \in \operatorname{supp}\hat\gamma} h_1(x) k(x, \cdot). \tag{S115}$$

Consequently,

$$\nabla f^\star_{\hat\alpha} = \sum_{x \in \operatorname{supp}\hat\gamma} h_1(x) \nabla k(x, \cdot), \qquad \nabla c_{f^\star_{\hat\alpha}} = \sum_{x \in \operatorname{supp}\hat\gamma} h_1(x) \nabla k(x, \cdot) c'(f^\star_{\hat\alpha}(\cdot)). \tag{S116}$$

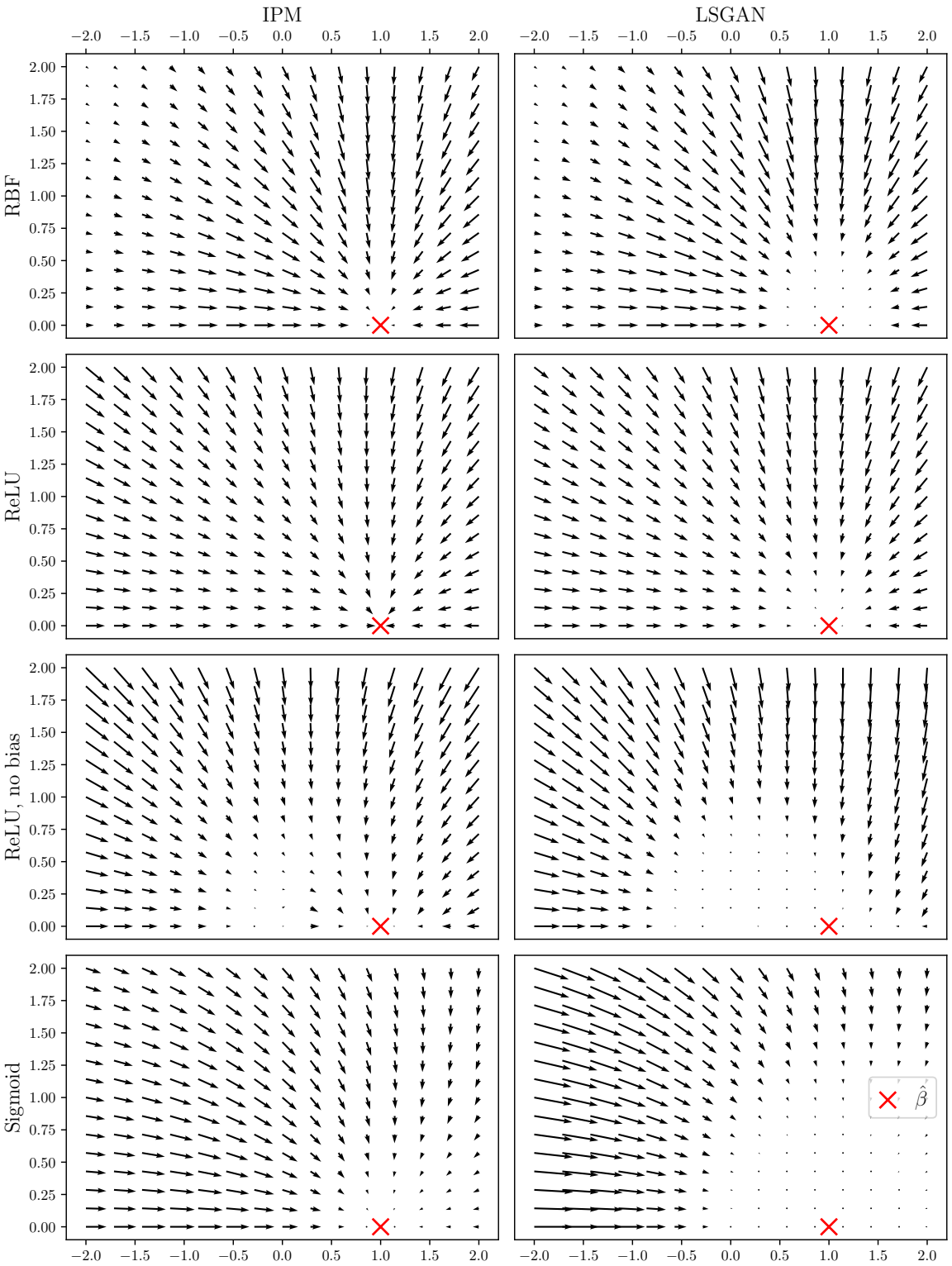


Figure S14. – Gradient field $\nabla c_{f^\star_{\hat{\alpha}_x}}(x)$ received by a generated sample $x \in \mathcal{R}^2$ (i.e. $\hat{\alpha} = \hat{\alpha}_x = \delta_x$) initialized to x_0 with respect its coordinates in $\text{Span}\{x_0, y\}$ where y, marked by a $\times$, is the target distribution (i.e. $\hat{\beta} = \delta_y$), with $\|y\| = 1$. Arrows correspond to the movement of x in $\text{Span}\{x_0, y\}$ following $\nabla c_{f^\star_{\hat{\alpha}_x}}(x)$, for different losses and networks; scales are specific for each pair of loss and network. The ideal case is the convergence of x along this gradient field towards the target y. Note that in the chosen orthonormal coordinate system, without loss of generality, y has coordinate $(1, 0)$; moreover, the gradient field is symmetrical with respect to the horizontal axis.

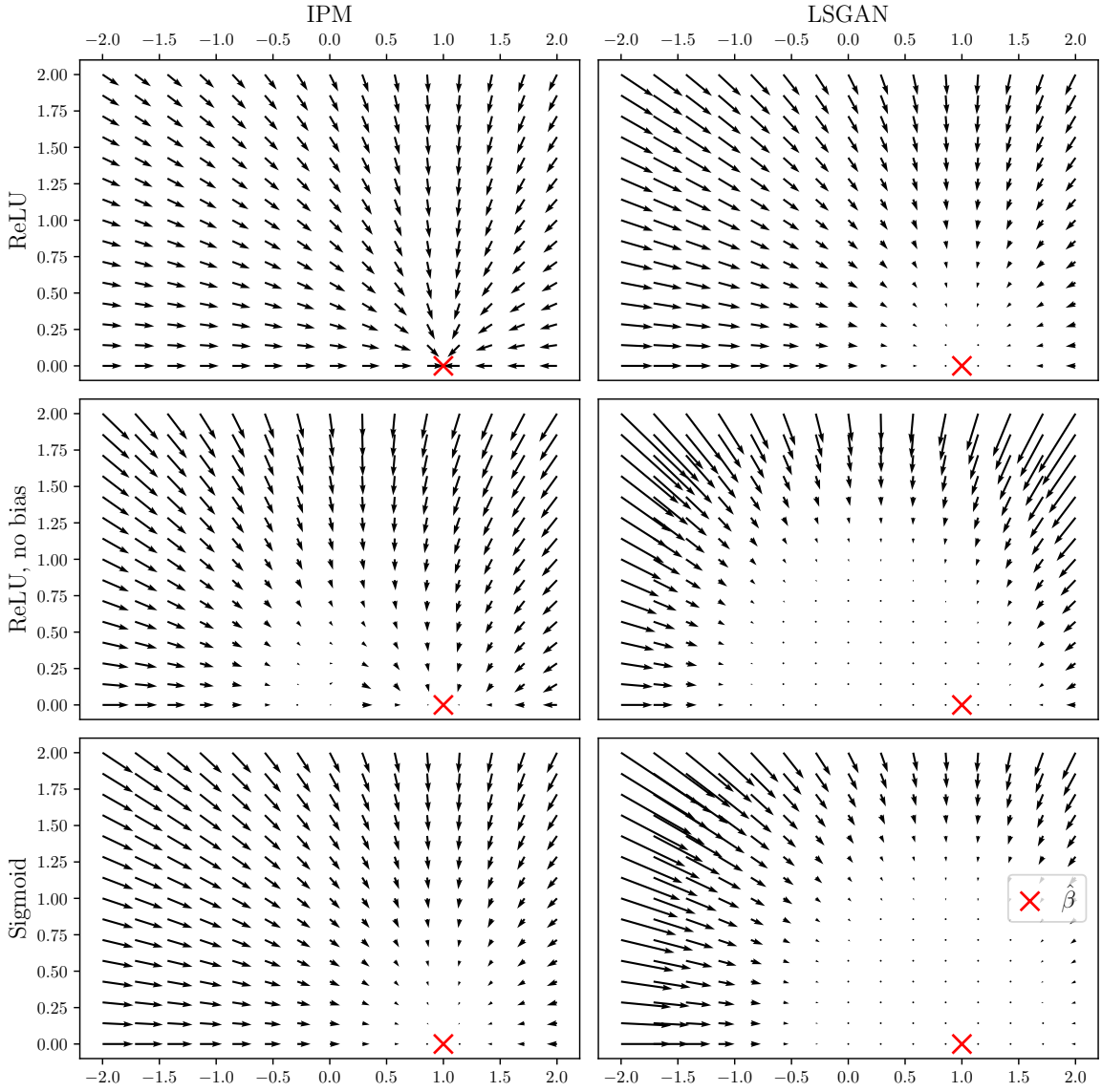


Figure S15. – Same plot as Figure S14 but with underlying points $x, y \in \mathcal{R}^{512}$.

Dirac-GAN Setting. The latter linear combination of gradients indicates that, by examining gradients of $c_{f^\star_{\hat{\alpha}}}$ for pairs of $(x, y) \in (\operatorname{supp} \hat{\alpha}) \times \left(\operatorname{supp} \hat{\beta}\right)$, one could already develop potentially valid intuitions that can hold even when multiple points are considered. This is especially the case for the IPM loss, as h_0, h_1 have a simple form: $h_1(x) = 1$ if $x \in \operatorname{supp} \hat{\alpha}$ and $h_1(y) = -1$ if $y \in \operatorname{supp} \hat{\alpha}$ (assuming points from $\hat{\alpha}$ and $\hat{\beta}$ are uniformly weighted); moreover, note that $c'(f^\star_{\hat{\alpha}}(\cdot)) = 1$. Thus, we study here $\nabla c_{f^\star_{\hat{\alpha}}}$ when $\hat{\alpha}$ and $\hat{\beta}$ are only comprised of one point, i.e. the setting of Dirac GAN Mescheder et al. 2018, with $\hat{\alpha} = \delta_x$ and $\hat{\beta} = \delta_y$.

Visualizing High-Dimensional Inputs. Unfortunately, the gradient field is difficult to visualize when the samples live in a high-dimensional space. Interestingly, the NTK $k(x, y)$ for any architecture starting with a fully connected layer only depends on $\|x\|$, $\|y\|$ and $\langle x, y \rangle$ G. Yang et al. 2019, and therefore all the information of $\nabla c_{f^\star_{\hat{\alpha}}}$ is contained in $\operatorname{Span}\{x, y\}$. From this, we show in Figures S14 and S15 the gradient field $\nabla c_{f^\star_{\hat{\alpha}}}$ in the two-dimensional space $\operatorname{Span}\{x, y\}$ for different architectures and losses in the infinite-width regime described in Section 5.3.4 and in this section. Figure S14 corresponds to two-dimensional $x, y \in \mathcal{R}^2$, and Figure S15 to high-dimensional $x, y \in \mathcal{R}^{512}$. Note that in the plots, the gradient field is symmetric w.r.t. the horizontal axis and for this reason we have restricted ourselves to the case where the second coordinate is positive.

Convergence of the Gradient Flow. In the last paragraph, we have seen that the gradient field in the Dirac-GAN setting lives in the two-dimensional $\operatorname{Span}\{x, y\}$, independently of the dimensionality of x, y. This means that when training the generated distribution, as in Section 5.3.4, the position of the particle x during training always remains in this two-dimensional space, and hence (non-)convergence in this setting can be easily checked by studying this gradient field. This is what we do in the following, for different architectures and losses.

Qualitative Analysis of the Gradient Field

x is far from y. When generated outputs are far away from the target, it is essential that their gradient has a large enough magnitude in order to *pull* these points towards the target. The behavior of the gradients for distant points can be observed in the plots. For ReLU networks, for both losses, the gradients for distant points seem to be well behaved and large enough. Note that in the IPM case, the magnitude of the gradients is even larger when x is further away from y. This is not the case for the RBF kernel when the variance parameter is too small, as the magnitude of the gradient becomes prohibitively small. Note that we selected a large variance parameter in order to avoid such a behavior, but

diminishing magnitudes can still be observed. Note that choosing an overly large variance may also have a negative impact on the points that are closer to target.

x is close to y. A particularity of the NTK of ReLU discriminators with bias that arises from this study is that the gradients vanish more slowly when the generated x tends to the target y, compared to NTKs of ReLU without bias and sigmoid networks, and to the RBF kernel. We hypothesize that this is also another distinguishing feature that helps the generated distribution to converge more easily to the target distribution, especially when they are not far apart. On the contrary, this gradient vanishes more rapidly for NTKs of ReLU without bias and sigmoid networks, compared to the RBF kernel. This can explain the worse performance of such NTKs compared to the RBF kernel in our experiments (see Tables S1 to S3). Note that this phenomenon is even more pronounced in high-dimensional spaces such as in Figure S15.

x is close to 0. Finally, we highlight gradient vanishing and instabilities around the origin for ReLU networks without bias. This is related to its differentiability issues at the origin exposed in Section 5.2.3.2, and to its lack of representational power discussed in Appendix C.2.2.5. This can also be retrieved on larger scale experiments of Figures S10 and S11 where the origin in the source of instabilities in the descent.

Sigmoid Network. It is also possible to evaluate the properties of the discriminator's gradient for architectures that are not used in practice, such as networks with the sigmoid activation. Figures S10 and S11 provide a clear explanation: as stated above, the magnitudes of the gradients become too small when $x \to y$, and heavily depend on the direction from which x approaches y. Ideally, the induced gradient flow should be insensitive to the direction in order for the convergence to be reliable and robust, which seems to be the case for ReLU networks.

C.2.4 Experimental Details

We detail in this section experimental parameters needed to reproduce our experiments.

C.2.4.1 GAN(TK)^2 Specifications and Computing Resources

GAN(TK)^2 is implemented in Python (tested on Python Python 3.8.1 and 3.9.2) and based on JAX Bradbury et al. 2018 for tensor computations and Neural Tan-

gents Roman Novak et al. 2020 for NTKs. Sinkhorn divergences between cloud points are computed using Geomloss Feydy et al. 2019. We refer to the code provided in the supplementary material for detailed specifications and instructions.

All experiments presented in this paper were run on Nvidia GPUs (Nvidia Titan RTX – 24GB of VRAM – with CUDA 11.2 as well as Nvidia Titan V – 12GB – and Nvidia GeForce RTX 2080 Ti – 11 GB – with CUDA 10.2). All two-dimensional experiments require only a few minutes of computations on a single GPU. Experiments on MNIST were run using simultaneously four GPUs for parallel computations, for at most a couple of hours.

C.2.4.2 Datasets

8 Gaussians. The target distribution is composed of 8 Gaussians with their means being evenly distributed on the centered sphere of radius 5, and each with a standard deviation of 0.5. The input fake distribution is drawn at initialization from a standard normal distribution $\mathcal{N}(0, 1)$. We sample in our experiments 500 points from each distribution at each run to build $\hat{\alpha}$ and $\hat{\beta}$.

AB and Density. These two datasets are taken from the Geomloss library examples Feydy et al. 2019 [3] and are licensed under the MIT license. To sample a point from a distribution based on these greyscale images files, we sample a pixel (considered to lie in $[-1, 1]^2$) in the image from a distribution where each pixel probability is proportional to the darkness of this pixel, and then apply a Gaussian noise centered at the chosen pixel coordinates with a standard deviation equal to the inverse of the image size. We sample in our experiments 500 points from each distribution at each run to build $\hat{\alpha}$ and $\hat{\beta}$.

MNIST. MNIST LeCun et al. 1998 is a standard dataset containing white digits over a dark frame, with no known license to the best of our knowledge. [4] We preprocess each MNIST image by extending it from 28×28 frames to 32×32 frames (by padding it with black pixels) and normalizing pixels in the $[-1, 1]$ range. For our experiments, we consider a subset of 1024 elements of MNIST, which are randomly sampled for each run.

3. They can be found at `https://github.com/jeanfeydy/geomloss/tree/master/geomloss/examples/optimal_transport/data`: AB corresponds to files `A.png` (source) and `B.png` (target), and Density corresponds to files `density_a.png` (source) and `density_a.png` (target).

4. No license is provided on the official webpage: `http://yann.lecun.com/exdb/mnist/`.

C.2.4.3 Parameters

Sinkhorn divergence The Sinkhorn divergence is computed using the Geomloss library Feydy et al. 2019, with a blur parameter of 0.001 and a scaling of 0.95, making it close to the Wasserstein $\mathcal{W}_2$ distance.

RBF kernel. The RBF kernel used in our experiments is the following:

$$k(x, y) = \mathrm{e}^{\frac{\|x-y\|^2}{2n}}, \tag{S117}$$

where n is the dimension of x and y, i.e. the dimension of the data.

Architecture. We used for the neural networks of our experiments the standard NTK paramaterization Jacot et al. 2018c, with a scaling factor of 1 for matrix multiplications and, when bias in enabled, a multiplicative constant of 1 for biases (except for sigmoid where this bias factor is lowered to 0.2 to avoid saturating the sigmoid). All considered networks are composed of 3 hidden layers and end with a linear layer. In the finite-width case, the width of these hidden layers is 128. We additionally use antisymmetric initialization Yaoyu Zhang et al. 2020 when using the IPM loss.

Discriminator optimization. Discriminators in the finite-width regime are trained using full-batch gradient descent without momentum, with one step per update to the distributions and the following learning rates ε:

- for the IPM loss: $\varepsilon = 0.01$;
- for the IPM loss with reset and LSGAN: $\varepsilon = 0.1$.

In the infinite-width limit, we use the analytic expression derived in Section 5.2.4 with training time $\tau = 1$ (except for MNIST where $\tau = 1000$).

Point cloud descent. The multiplicative constant η over the gradient applied to each datapoint for two-dimensional problems is chosen as follows:

- for the IPM loss in the infinite-width regime: $\eta = 1000$;
- for the IPM loss in the finite-width regime: $\eta = 100$;
- for the IPM loss in the finite-width regime and discriminator reset: $\eta = 1000$;
- for LSGAN in the infinite-width regime: $\eta = 1000$;

- for LSGAN in the finite-width regime: $\eta = 1$.

We multliply η by 1000 when using sigmoid activations, because of the low magnitude of the gradients it provides. We choose for MNIST $\eta = 100$.

Training is performed for the following number of iterations:

- for 8 Gaussians: $20\,000$;
- for Density and AB: $10\,000$;
- for MNIST: $50\,000$.

C.3 Appendix 5.3: Normalizing Kalman Filters for Multivariate Time Series Forecasting

C.3.1 Proofs

For completeness, we restate the *NKF* model:

$$\begin{aligned} \mathbf{l}_t &= F_t \mathbf{l}_{t-1} + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \Sigma_t), \\ \mathbf{y}_t &= f_t(A_t^T \mathbf{l}_t + \beta\varepsilon_t), \quad \beta\varepsilon_t \sim \mathcal{N}(0, \Gamma_t). \end{aligned} \tag{S118}$$

C.3.1.1 Filtering

Proposition S13 (Filtering)**.** *The filtered distributions of the NKF model are Gaussian and are given by the filtered distributions of the corresponding LGM with pseudo-observations* $z_t := f_t^{-1}(y_t),\ t \geq 1$. *That is,* $p(l_t|y_{1:t}; \Theta, \Lambda) = p_{LGM}\left(l_t|z_{1:t}; \Theta\right)$ *where* p_{LGM} *refers to the distribution given by the LGM.*

Proof. Note that for simplicity we omit conditioning on the parameters. Let us first recall the recursive Bayesian estimation, consisting of two distinct steps, predict and update of the latent state l_t given by

$$\begin{aligned} \text{predict:} \quad & p(l_t|y_{1:t-1}) = \int p(l_t|l_{t-1}) p(l_{t-1}|y_{1:t-1}) \mathrm{d}l_{t-1}, \\ \text{update:} \quad & p(l_t|y_{1:t}) = \frac{p(y_t|l_t) p(l_t|y_{1:t-1})}{\int p(y_t|l_t) p(l_t|y_{1:t-1}) \mathrm{d}l_t}. \end{aligned} \tag{S119}$$

The filtered distribution $p(l_t|y_{1:t})$ is obtained recursively applying both these steps until time t, starting from a prior on the state $p(l_1)$. However, in its current form one could expect that computing the latter would require approximating integrals, as equation S118 is non-linear and non-Gaussian, which is prohibitive in the high-dimensional setting. But we show by induction that filtered distributions are Gaussian and in fact coincide with the filtered distributions of the underlying linear Gaussian state space model.

Let $z_t := A_t l_t + \beta\varepsilon_t$ so that $z_t = f_t^{-1}(y_t), \forall t = 1, 2, \ldots, T$. Since y_t is observed and f_t is an invertible, deterministic function, we can view z_t as the pseudo-observation generated from the underlying linear Gaussian state space model (*LGM*),

$$\begin{aligned} \mathbf{l}_t &= F_t \mathbf{l}_{t-1} + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \Sigma_t), \\ \mathbf{z}_t &= A_t^T \mathbf{l}_t + \beta\varepsilon_t, \quad \beta\varepsilon_t \sim \mathcal{N}(0, \Gamma_t), \end{aligned} \tag{S120}$$

By making use of the change of variables formula, the likelihood term in the update step of (S119) can be written as:

$$p(y_t|l_t) = p_{\mathbf{z}_t}(z_t|l_t) D f_t^{-1}(y_t), \tag{S121}$$

where $Dg(x) := |\det[\text{Jac}_x(g)]|$ is the absolute value of the determinant of the Jacobian of g evaluated at x. By Eq. (S120), $p_{\mathbf{z}_t}(z_t|l_t)$ is the density of the Gaussian variable $Z_t = A_t^T l_t + \beta\varepsilon_t$ conditioned on l_t, and from this we obtain $\forall t = 1, 2, \ldots T$,

$$p(y_t|l_t) = \mathcal{N}(z_t|\, A_t^T l_t, \Gamma_t) D f_t^{-1}(y_t). \tag{S122}$$

We proceed with inductive proof, first showing that the filtered distribution for the base case $t = 1$, $p(l_1|y_1)$, is Gaussian and the same as that of the underlying *LGM*. For this, we start with the first prediction step $p(l_1) = \mathcal{N}(\mu_1, \Sigma_1)$, which is assumed to be Gaussian. From Eq. (S121), assuming $Df_1^{-1}(y_1)$ is non-zero, we get:

$$p(l_1|y_1) = \frac{p(l_1)p(y_1|l_1)}{\int p(l_1)p(y_1|l_1)dl_1} = \frac{p(l_1)p_{\mathbf{z}_1}(z_1|l_1)Df_1^{-1}(y_1)}{\int p(l_1)p_{\mathbf{z}_1}(z_1|l_1)Df_1^{-1}(y_1)dl_1} = \frac{p(l_1)p_{\mathbf{z}_1}(z_1|l_1)}{\int p(l_1)p_{\mathbf{z}_1}(z_1|l_1)dl_1} = p_{LGM}(l_1|z_1).$$

This is exactly the first filtered distribution for *LGM* with observation z_1 and is in fact Gaussian since $p(z_1|l_1)$ is Gaussian by Eq. (S122) and $p(l_1)$ is assumed to be Gaussian. Let μ_1^f, Σ_1^f denote the mean and covariance of this filtered distribution. For the inductive step assume that $p(l_{t-1}|y_{1:t-1})$ is Gaussian with mean μ_{t-1}^f and variance Σ_{t-1}^f, and is the same as $p_{LGM}(l_{t-1}|z_{1:t-1})$. We will prove that $p(l_t|y_{1:t}) = p_{LGM}(l_t|z_{1:t})$.

As both $p(l_{t-1}|y_{1:t-1})$ and $p(l_t|l_{t-1}) = \mathcal{N}(l_t|F_t l_{t-1}, \Sigma_t)$ are Gaussian, it follows that the distribution obtained from the prediction step must also be Gaussian, i.e., $p(l_t|y_{1:t-1}) = \mathcal{N}(l_t|\mu_t^p, \Sigma_t^p), \forall t = 2, \ldots, T$, with:

$$\begin{aligned} \mu_t^p &= F_t \mu_{t-1}^f, \\ \Sigma_t^p &= F_t \Sigma_{t-1}^f F_t^T + \Sigma_t. \end{aligned} \tag{S123}$$

In fact, this coincides with $p_{LGM}(l_t|z_{1:t-1})$ given that $\mu_{t-1}^f, \Sigma_{t-1}^f$ are the same for both *LGM* and our model and both use the same transition for the latent state.

Similar to the base case, the filtered distribution for $t > 1$ is

$$\begin{aligned} p(l_t|y_{1:t}) &= \frac{p(y_t|l_t)p(l_t|y_{1:t-1})}{\int p(y_t|l_t)p(l_t|y_{1:t-1})\mathrm{d}l_t} \\ &= \frac{Df_t^{-1}(y_t)p_{\mathbf{z}_t}(z_t|l_t)p(l_t|y_{1:t-1})}{\int Df_t^{-1}(y_t)p_{\mathbf{z}_t}(f_t^{-1}(y_t)|l_t)p(l_t|y_{1:t-1})\mathrm{d}l_t} \\ &= \frac{p_{\mathbf{z}_t}(z_t|l_t)p_{LGM}(l_t|z_{1:t-1})}{\int p_{\mathbf{z}_t}(z_t|l_t)p_{LGM}(l_t|z_{1:t-1})\mathrm{d}l_t} \\ &= p_{LGM}(l_t|z_{1:t}), \end{aligned} \tag{S124}$$

which is the same as the filtered distribution of the corresponding *LGM*. In fact, one can deduce this filtered distribution in closed-form:

$$\begin{aligned} p(l_t|y_{1:t}) &= \frac{\mathcal{N}(z_t|\ A_t^T l_t, \Gamma_t)\mathcal{N}(l_t|\mu_t^p, \Sigma_t^p)}{\int \mathcal{N}(z_t|\ A_t^T l_t, \Gamma_t)\mathcal{N}(l_t|\mu_t^p, \Sigma_t^p)\mathrm{d}l_t} \\ &= \mathcal{N}(l_t|\ \mu_t^f, \Sigma_t^f), \end{aligned} \tag{S125}$$

where $\mu_t^f = \mu_t^p + K_t[f^{-1}(y_t) - A_t^T \mu_t^p]$, $\Sigma_t^f = (I - K_t A_t^T)\Sigma_t^p$, and $K_t = \Sigma_t^p A_t (A_t^T \Sigma_t^p A_t + \Gamma_t)^{-1}$. This recursive formula for the filtered distribution is valid for $t > 1$ and the same for the base case $t = 1$ is obtained by noting that $\mu_1^p = \mu_1$ and $\Sigma_1^p = \Sigma_1$. □

C.3.1.2 Smoothing

Proposition S14 (Smoothing)**.** *The smoothed distributions of the NKF model are Gaussian and are given by the smoothed distributions of the corresponding LGM with pseudo-observations* $z_t := f_t^{-1}(y_t), t = 1, 2 \ldots, T$*. That is,* $p(l_t|y_{1:T}; \Theta, \Lambda) = p_{LGM}(l_t|z_{1:T}; \Theta)$.

Proof. Note that for simplicity, we omit conditioning on the parameters. This can again be proved by induction starting with the base case $t = T$ and running backwards. For the base case the smoothed distribution $p(l_T|y_{1:T})$ coincides with the filtered distribution for the final time step T, which was already shown to be equal to that of the standard *LGM*. For the inductive step assume that in time

step $t+1$ it holds that $p(l_{t+1}|y_{1:T}) = p_{LGM}(l_{t+1}|z_{1:T})$. We will prove that the same is true in time step t, i.e., $p(l_t|y_{1:T}) = p_{LGM}(l_t|z_{1:T})$.

$$\begin{aligned}
p(l_t|y_{1:T}) &= \int p(l_t, l_{t+1}|y_{1:T})\mathrm{d}l_{t+1} \\
&= \int p(l_{t+1}|y_{1:T})p(l_t|l_{t+1}, y_{1:T})\mathrm{d}l_{t+1} \\
&= \int p(l_{t+1}|y_{1:T})p(l_t|l_{t+1}, y_{1:t})\mathrm{d}l_{t+1} \\
&= p(l_t|y_{1:t}) \int \frac{p(l_{t+1}|y_{1:T})p(l_{t+1}|l_t)}{p(l_{t+1}|y_{1:t})}\mathrm{d}l_{t+1} \\
&= p_{LGM}(l_t|z_{1:t}) \int \frac{p_{LGM}(l_{t+1}|z_{1:T})p(l_{t+1}|l_t)}{p_{LGM}(l_{t+1}|z_{1:t})}\mathrm{d}l_{t+1} \\
&= p_{LGM}(l_t|z_{1:T}),
\end{aligned} \tag{S126}$$

where in the penultimate step we used the fact that the predictive distribution $p(l_{t+1}|y_{1:t})$ and the filtered distribution $p(l_t|y_{1:t})$ of our model are the same as those of the standard *LGM* (see Proposition S13 and eq. S123). The smoothed distribution of the standard *LGM* is Gaussian with mean μ_t^s and variance Σ_t^s such that, starting from $\mu_T^s = \mu_T^f, \Sigma_T^s = \Sigma_T^f$:

$$\mu_t^s = \mu_t^f + G_t[\mu_{t+1}^s - \mu_{t+1}^p], \tag{S127a}$$
$$\Sigma_t^s = \Sigma_t^f + G_t[\Sigma_{t+1}^s - \Sigma_{t+1}^p], \tag{S127b}$$
$$G_t = \Sigma_t^f A_t[\Sigma_{t+1}^p]^{-1}, \tag{S127c}$$

where μ_t^f, Σ_t^f correspond to mean and covariance computed during the update step of the NKF, and μ_t^p, Σ_t^p computed in the prediction step (refer to Eq. (S123)).

□

C.3.1.3 Likelihood

Proposition S15 (Likelihood)**.** *The likelihood of the parameters (Θ, Λ) of the NKF model given the observations $\{y_{1:T}\}$ can be computed as*

$$\ell(\Theta, \Lambda) = p(y_{1:T}; \Theta, \Lambda) = \prod_{t=1}^{T} p_{LGM}(z_t|z_{1:t-1}; \Theta) \left|\det\left[Jac_{z_t}(f_t)\right]\right|^{-1}, \tag{S27}$$

where $z_t = f_t^{-1}(y_t)$ and $p_{LGM}(z_t|z_{1:t-1}; \Theta)$ denotes the predictive distribution of LGM.

Proof. We can compute the likelihood by decomposing it into telescoping conditional distributions and using the substitution $z_t = f^{-1}(y_t)$,

$$\begin{aligned}
p(y_{1:T};\Theta,\Lambda) &= \prod_{t=1}^{T} p(y_t|y_{1:t-1};\Theta,\Lambda) \\
&= \prod_{t=1}^{T} \int \left|\det\left[\text{Jac}_{y_t}(f_t^{-1})\right]\right| p_{\mathbf{z}_t}(z_t|l_t)p(l_t|y_{1:t-1};\Theta)\mathrm{d}l_t \\
&= \prod_{t=1}^{T} \left|\det\left[\text{Jac}_{y_t}(f_t^{-1})\right]\right| \int p_{\mathbf{z}_t}(z_t|l_t)p_{LGM}(l_t|z_{1:t-1};\Theta)\mathrm{d}l_t \qquad \text{(S128)} \\
&= \prod_{t=1}^{T} \left|\det\left[\text{Jac}_{y_t}(f_t^{-1})\right]\right| p_{LGM}(z_t|z_{1:t-1};\Theta) \\
&= \prod_{t=1}^{T} \left|\det\left[\text{Jac}_{y_t}(f_t^{-1})\right]\right| \mathcal{N}(z_t;\nu_t^p,\Gamma_t^p),
\end{aligned}$$

where in the third step we used the fact that predictive distributions of our model (S118) and the standard *LGM* are the same. This is true because the filtered distributions of our model and the *LGM* are the same since both use the same transition for the latent state, as shown in Proposition S13 (see (S123)). In the final step we used the fact that the predictive distribution $p(z_t|z_{1:t-1})$ of the standard *LGM* is Gaussian with mean ν_t^p and covariance Γ_t^p given by the analytical expressions Barber 2011:

$$\begin{aligned}
&\nu_t^p = A_t^T F_t \mu_{t-1}^f, && \Gamma_t^p = A_t^T\left(F_t\Sigma_{t-1}^f F_t^T + \Sigma_t\right)A_t + \Gamma_t, && t>1, \\
&\nu_1^p = A_t^T\mu_1, && \Gamma_1^p = A_t^T\Sigma_1 A_t + \Gamma_1, && t=1.
\end{aligned}$$

□

C.3.1.4 Forecasting Distribution

The joint forecasting distribution of the future time steps $T+1:T+\tau$ can be obtained in terms of the predictive distribution of the corresponding *LGM* with $z_t = f^{-1}(y_t)$,

$$p(y_{T+1:T+\tau}|y_{1:T},x_{1:T+\tau};\Theta,\Lambda) = \prod_{t=T+1}^{T+\tau} p_{LGM}(z_t|z_{1:t-1};\Theta)\left|\det\left[\text{Jac}_{z_t}(f_t)\right]\right|^{-1}. \qquad \text{(S129)}$$

The exact analytical expressions for the forecasting distribution is given by:

$$p(y_{T+1},\dots,y_{T+\tau}|y_1,\dots,y_T,\Theta) = \prod_{t=T+1}^{T+\tau} \mathcal{N}(f^{-1}(y_t);\nu_t^p,\Gamma_t^p)\left|\det\left[\text{Jac}_{y_t}(f_t^{-1})\right]\right|, \tag{S130}$$

$$\nu_t^p = A_t^T\mu_t^p, \qquad \mu_t^p = F_t\mu_{t-1}^f, \tag{S131}$$

$$\Gamma_t^p = A_t^T\Sigma_t^pA_t+\Gamma_t, \qquad \Sigma_t^p = F_t\Sigma_{t-1}^fF_t^T+\Sigma_t, \tag{S132}$$

where (μ_t^p,Σ_t^p) and (ν_t^p,Γ_t^p) are the parameters of the predictive distributions for the latent state and observations given in Propositions S13 and S15, respectively.

C.3.1.5 Handling Missing Data

Given a subset of observed targets $y_{\mathbf{t}^{\text{obs}}}$, what can we say about the missing observations? Said otherwise, what is the probability of $p(y|y_{\mathbf{t}^{\text{obs}}})$? This data imputation problem is of central importance in numerous applications. This problem can be solved by first solving the smoothing problem in order to obtain the posterior $p(l_t|y_{\mathbf{t}^{\text{obs}}})$:

$$\begin{aligned} p(y|y_{\mathbf{t}^{\text{obs}}}) &= \int p(y,l|y_{\mathbf{t}^{\text{obs}}})\mathrm{d}l \\ &= \int p(y|l)p(l|y_{\mathbf{t}^{\text{obs}}})\mathrm{d}l \\ &= \prod_t p_{LGM}(z_t|z_{\mathbf{t}^{\text{obs}}})\left|\det\left[\text{Jac}_{z_t}(f_t)\right]\right|^{-1} \\ &= \prod_t \mathcal{N}(z_t|A_t^T\mu_t^{u,s},A_t^T\Sigma_t^{u,s}A_t+\Gamma_t)\left|\det\left[\text{Jac}_{z_t}(f_t)\right]\right|^{-1}, \end{aligned} \tag{S133}$$

where $(\mu_t^{u,s},\Sigma_t^{u,s})$ corresponds to the parameters of the smoothed distribution in the presence of missing data. Once again, this distribution admits an analytical expression and can be readily sampled from.

C.3.2 Model: Additional Details

C.3.2.1 Encoding of the LGM Parameters

In order to constrain the real-valued outputs h_t at time t of the RNN Ψ to the parameter domains of the *LGM*, we apply a sequence of transformations. For the j-th state space model parameter, Θ_t^j, we initially compute the affine transformation

$\tilde{\Theta}_t^j = w_j^\top h_t + b_j$, where the weights and biases are different for each parameter and are all included in Φ and learned. We then transform $\tilde{\Theta}_t^j$ to the domain of the parameter by applying:

- for real-valued parameters: no transformation i.e., $\Theta_t^j = \tilde{\Theta}_t^j$,
- for positive parameters: the softplus function $\Theta_t^j = \log(1 + \exp(\tilde{\Theta}_t^j))$,
- for bounded parameters in $[a, b]$: a scaled and shifted sigmoid $\Theta_t^j = (b - a)\frac{1}{1+\exp(-\tilde{\Theta}_t^j)} + a$.

In practice, it is often advisable to impose stricter bounds than theoretically required, e.g., enforcing an upper bound on the observation noise variance or a lower bound on the innovation strengths can stabilize the training procedure in the presence of outliers.

C.3.2.2 Local-Global Instantiation

As in Rangapuram et al. 2018, the evolution of the latent state $\mathbf{l}_t^{(i)}$ for each time series i is captured using a *composition* of level-trend and seasonality model, described below.

Local Level-Trend Model In the level-trend model, the latent state has two dimensions and is characterized by:

$$\begin{aligned} F_t^{(i)} &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad A_t^{(i)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \\ \Sigma_t^{(i)} &= \begin{bmatrix} \alpha_t^2 & 0 \\ 0 & \beta_t^2 \end{bmatrix}, \quad \Gamma_t^{(i)} \in \mathcal{R}_{++}. \end{aligned} \tag{S134}$$

Local Seasonality Model In the case of seasonality-based models, each seasonality pattern can be described by a set of seasonal factors (or seasons) associated with it. For example, in the day-of-week pattern there are seven factors, one for each day of the week. We can represent each factor as a component of the latent state $\mathbf{l}_t \in \mathcal{R}^7$. Then, for the day-of-week seasonality model, we have

$$\begin{aligned} F_t^{(i)} &= I, \quad A_t^{(i)} = \mathbb{I}_{\{\text{day(t)=j}\}_{j=1}^7}, \\ \Sigma_t^{(i)} &= \sigma_t^2 \text{diag}(A_t^{(i)}), \quad \Gamma_t^{(i)} \in \mathcal{R}_{++}. \end{aligned} \tag{S135}$$

Composite Model We concatenate the state of the level-trend model and the seasonality model in order to take into account both types of dynamics.

Note that in order to take into account the correlations across time series, the *pseudo-observations* generated by the composite state space model are given to the normalizing flow f_t, which will implicitly capture these correlations.

C.3.3 Experimental Details

We use the same hyperparameters for the model architecture across all datasets. For the RNN, we use an LSTM with the same architecture and hyperparameters as those proposed in *DeepState* Rangapuram et al. 2018, based on the open-source implementation from Alexandrov et al. 2020. To avoid numerical issues arising during the *NKF* update step, we find it useful to lower bound the observation noise $\Gamma_t^{(i)}$ to 0.1 for `elec`, `solar`, `wiki`, and 0.01 for the rest. The numerical issues are perhaps due to the overfitting of *LGM* parameters to the training data; the open-source implementation Alexandrov et al. 2020 of *DeepState* Rangapuram et al. 2018 also recommends such safe guards on the noise terms. Note that in the experiments f_t is the same across all time steps; however time-dependant noise may still be captured as the parameters of the *LGM* are time-dependent. An interesting research avenue for future work may be to consider a time-varying normalizing flow by conditioning it on temporal features, thus bringing us to consider conditional NFs Rezende et al. 2015 .

We evaluated various NF proposed in the literature: *RealNVP* Dinh et al. 2017, *Glow* Kingma et al. 2018 and *iResnet* Behrmann et al. 2019. We have finally opted for RealNVP which is straightforward to implement and does not suffer from drawbacks of the other methods. In particular, the number of parameters in Glow scales quadratically in the number of dimensions due to the fully connected layers in the permutation step: for the wiki dataset, as the dimension of the observations is 2×10^3, this would imply that just one permutation layer would have 4×10^6 parameters. In iResnet, the estimator of the Jacobian term yielded a high variance in the high dimensional setting and requires a number of forward passes in order to compute the inverse. For all the experiments, we have set the number of blocks of the RealNVP to 9. Each affine-coupling is parameterized by a 2-layer neural network with the numbers of hidden dimensions set to 16, without batch-normalization.

During training we tune the number of epochs for each dataset on a separate validation set of equal size to the test set. We use the Adam optimizer with 2×10^{-4} learning rate and 1×10^{-6} weight-decay.

As in Salinas et al. 2019a, the evaluation is done in a rolling fashion: for hourly datasets accuracy is measured on 7 rolling time windows where each roll corre-

sponds to 24 hours, thus covering 7 days of the test set. For all the other datasets we use 5 windows. More details on the forecast horizon τ, domain, frequency, dimension N and length of training timesteps T are given in the Appendix C.3.3.2.

C.3.3.1 Qualitative Experiments

Datasets

Both datasets are composed of 2 daily univariate time series with weekly seasonality, correlated in different ways. The time series observations $y_{1:T}$, with $T = 120$, are generated according to the following state space model:

$$\begin{aligned} \mathbf{l}_t &= \mathbf{l}_{t-1} + \beta\epsilon_t, \quad \beta\epsilon_t \sim \mathcal{N}(0, 10^{-3} \times I), \\ \mathbf{y}_t &= m(A_t^T \mathbf{l}_t) + \beta\eta_t, \quad \beta\eta_t \sim \mathcal{D}(t), \end{aligned} \tag{S136}$$

where $\mathbf{l}_t = [\mathbf{l}_t^{(1)}; \mathbf{l}_t^{(2)}] \in \mathcal{R}^{14}$ and each component of $\mathbf{l}_t^{(i)} \in \mathcal{R}^7$ is associated to a day of the week, $A_t \in \mathcal{R}^{14\times 2}$ is block diagonal where $A_t^{(i)} = \mathbb{I}_{\{\text{day(t)=j}\}_{j=1}^7}$ *selects* the corresponding state component based on t, $m : \mathcal{R}^2 \to \mathcal{R}^2$ is a *mixing* function, and $\mathcal{D}(t)$ is a highly non-Gaussian, time-dependent distribution. The initial state $l_1^{(i)}$ is sampled from a Gaussian distribution $\mathcal{N}(\mu_0, 10^{-3} \times I)$, with 3 different mean levels according to the day of the week[5]: $\mu_1 = [-10, -10, 0, 0, 0, 10, 10]^T$.

For the first experiment (corresponding to Figure S12a) we consider the simple case where no mixing occurs, i.e., m is the identity function. The observational noise $\beta\eta_t$ is the same for every t and highly non-Gaussian. To this respect, $\beta\eta_t^{(1)}$ follows an 1D Uniform distribution and $\beta\eta_t^{(2)}$ is the image of $\beta\eta_t^{(1)}$ when mapped through a cosine function.

For the second experiment (corresponding to Figure S12b) we consider a more complex setting where mixing occurs and the observational noise $\beta\eta_t$ is time-dependent. Artificial dependencies across time series are induced, setting the *mixing* function m to $m([x, y]^T) = [x\,y, x + y]^T$. The observational noise $\beta\eta_t$ follows a day-of-week pattern, where three non-Gaussian distributions are selected based on the day of the week, similarly as for μ_1: for the first two days $\beta\eta_t$ is the same in the first experiment, for the three next days $\beta\eta_t$ follows a mixture of two Gaussians, and for the weekend $\beta\eta_t$ is a simple line generated from an 1D Uniform distribution.

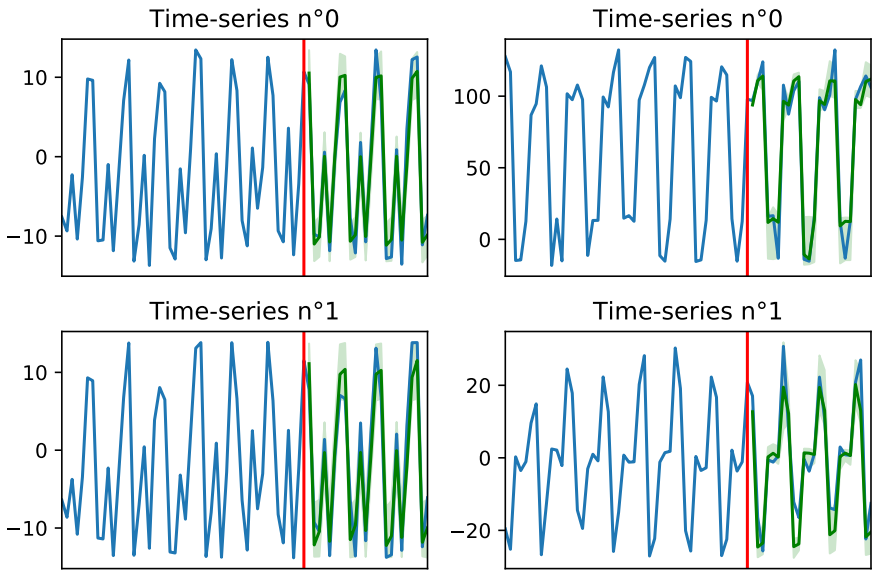


Figure S16. – Forecast results with NF (*NKF* model) for both time-series of dataset `S1` (left) and `S2` (right). For each forecast: In blue, the values of the input (left of red line) and target (right of red line) time series. In dark green, the forecasts mean, and quantiles $[0.1, 0.9]$ are filled light green.

Forecast Samples

C.3.3.2 Quantitative Experiments

Datasets

We evaluate on the public datasets (with the same training and test splits) used in Salinas et al. 2019a: `exchange`: daily exchange rate between 8 currencies as used in Lai et al. 2017; `solar`: hourly photo-voltaic production of 137 stations in Alabama State used in Lai et al. 2017; `elec`: hourly time series of the electricity consumption of 370 customers Dheeru et al. 2017; `traffic`: hourly occupancy rate, between 0 and 1, of 963 San Francisco car lanes Dheeru et al. 2017; `wiki`: daily page views of 2000 Wikipedia pages used in Gasthaus et al. 2019. Similar to Salinas et al. 2019a, the forecasts of different methods are evaluated by splitting each dataset in the following fashion: all data prior to a fixed *forecast start date* compose the training set and the remainder is used as the test set. We measure the accuracy of the forecasts of various methods on all the time points of the test set.

5. Note that in our notation (see Eq. (S24)), the prior state $\mathbf{l}_1$ is indexed by 1 not 0.

In Table S4 we summarize the details of the datasets used to evaluate the models.

dataset	τ (num steps predicted)	domain	frequency	dimension N	time steps T
`exchange`	30	$\mathcal{R}^+$	daily	8	6071
`solar`	24	$\mathcal{R}^+$	hourly	137	7009
`elec`	24	$\mathcal{R}^+$	hourly	370	5790
`traffic`	24	$\mathcal{R}^+$	hourly	963	10413
`wiki`	30	$\mathbb{N}$	daily	2000	792

Table S4. – Summary of the datasets used to test the models. Number of steps forecasted, data domain $\mathcal{D}$, frequency of observations, dimension of series N, and number of time steps T.

Ablation Study NKF-Local

Here we give additional details for the local variant of our model *NKF-Local*. This variant uses the same form for the state-space model (Eq. (S31a) and (S31b)), with an alternative local normalizing flow $u_t : \mathcal{R} \to \mathcal{R}$, applied to each time-series independently:

$$f_t(z_t) = (u_t(z_{1,t}), \dots, u_t(z_{N,t})). \tag{S137}$$

In this case, the conditional density in Eq. (S25) of variables formula can easily be expressed in terms of normalizing flow u_t, and reduces to:

$$\begin{aligned} p(y_t|l_t;\Theta,\Lambda) &= p(y_{1,t},\dots,y_{N,t}|l_t;\Theta,\Lambda) \\ &= p_{z_t}(u_t^{-1}(y_{1,t}),\dots,u_t^{-1}(y_{N,t});\Theta)\prod_{i=1}^{N}\left|\det\left[\mathrm{Jac}_{y_{t,i}}(u_t^{-1})\right]\right|. \end{aligned} \tag{S138}$$

Written in this form, we can see that the computation of the Jacobian term scales linearly in the dimension, as $y_{i,t} \in \mathcal{R}$ and can be calculated in parallel.

For the univariate u_t , we use non-time dependent iResnet Behrmann et al. 2019, with 6 invertible blocks, LipSwish activation, spectral normalizing coefficient of 0.9, and 10 fixed-point iterations for the computation of the inverse, and 1 iteration for the power iteration method. We set the learning rate associated to the RNN to 0.001, and 0.00001 for the normalizing flow.

C.3.3.3 Handling Missing Data

Here we report the exact numbers for the missing data experiment, Table S5.

	elec10%	elec30%	elec50%	elec70%	elec90%
KVAE	0.046 ± 0.0048	0.051 ± 0.0062	0.13 ± 0.059	0.15 ± 0.039	0.16 ± 0.051
DeepAR	0.045±0.001	0.068±0.001	0.079±0.009	0.097±0.010	0.145±0.023
GP-Copula	0.036±0.0013	0.045±0.002	0.046±0.0046	0.092±0.0091	0.094±0.0077
NKF	**0.015±0.001**	**0.016±0.001**	**0.016±0.000**	**0.018±0.001**	**0.026±0.0019**

Table S5. – CRPS-Sum-N (lower is better), averaged over 3 runs.

C.3.3.4 A Note on Evaluation Multivariate Metrics for Probabilistic Forecasting

CRPS relies on the pinball loss that measures the fit at each quantile between the quantile function F^{-1} and an observation:

$$\Lambda_\alpha(q, y) = (\alpha - \mathbb{I}_{\{y<q\}})(y - q), \tag{S139}$$

where $\alpha \in [0, 1]$ is the quantile level and q the respective quantile of the probability distribution.

The integrated pinball loss over all quantile levels $\alpha \in [0, 1]$ is defined as the CRPS:

$$\text{CRPS}(F^{-1}, y) = \int_0^1 2\Lambda_\alpha(F^{-1}(\alpha), y)\, d\alpha. \tag{S140}$$

We estimate F^{-1} by drawing 100 samples and sorting them.

Although CRPS is a widely accepted metric for assessing the quality of probabilistic forecasts in the univariate case, it is not applicable in the multivariate setting as it does not capture correlations across time-series. Instead, Salinas et al. 2019a introduced CRPS-Sum an extension of CRPS to the multivariate case:

$$\text{E}_t[\text{CRPS}(F^{-1}, \sum_i y_{i,t})],$$

where F^{-1} is estimated by summing the samples across dimensions and then computing the quantiles by sorting. However, Salinas et al. 2019a does not give a theoretical justification of CRPS-Sum.

Below, we prove that CRPS-Sum is a proper scoring rule.

We restate fundamental definitions and results from Gneiting et al. 2007 on proper scoring rules first before showing that CRPS-Sum-N is a proper scoring rule.

Let Ω be the sample space, let $\mathcal{A}$ be a σ-algebra on Ω, and let $\mathcal{P}$ be a convex class of probability measures on $(\Omega, \mathcal{A})$.

Definition 1. A *scoring rule* is a function $S : \mathcal{P} \times \Omega \to [-\infty, \infty]$, such that for every $P \in \mathcal{P}$ we have that $E(S(P, \cdot))$ exists (and is possibly non-finite). For such an S, and for every $P, Q \in \mathcal{P}$, define

$$S(P, Q) := \int S(P, \cdot) dQ.$$

Properness is defined thusly:

Definition 2. A scoring rule is proper with respect to $\mathcal{P}' \subset \mathcal{P}$ if $\forall P, Q \in \mathcal{P}'$ we have that $S(P, P) \geq S(Q, P)$. It is called strictly proper with respect to $\mathcal{P}'$ if equality holds iff $P = Q$ a.s.

Theorem 3. *Let $\Omega = \mathbb{R}$, and $\mathcal{A}$ be the Borel σ-algebra. Then the function*

$$\mathrm{CRPS}(P, x) := -\int_{\mathbb{R}} (P(\{t | t \leq y\}) - \mathbb{I}_{x \leq y})^2 dy$$

is a proper scoring rule with respect to the set $\mathcal{P}$ of probability measures on $\mathcal{A}$. Furthermore, if we restrict to the subclass $\mathcal{P}'$ of probability measures with finite first moment, then it can be shown that

$$\mathrm{CRPS}(P, x) = \frac{1}{2} E_P(|X - X'|) - E_P(|X - x|),$$

where X and X' are understood as independent random variables that have the distribution P, and CRPS *is strictly proper with respect to $\mathcal{P}'$.*

1. The term "$E_P(|X - X'|)$ where X and X' are understood as independent random variables that have the distribution P" means, by definition: $\int_{\Omega \times \Omega} |p_1 - p_2| d(P \times P)$, where p_i is the projection to the i^{th} coordinate.
2. It is crucial in the definition of properness that we require that $\forall P, Q \in \mathcal{P}'$ we have that $S(P, P) \geq S(Q, P)$ rather than $S(P, P) \geq S(P, Q)$. Indeed, it is easy to see that if P follows a standard normal distribution and Q is the constant distribution 0, then

$$\mathrm{CRPS}(P, Q) - \mathrm{CRPS}(P, P) =$$

$$E_P(|X - X'|) - E_P(|X|) =$$

$$\frac{2}{\sqrt{\pi}} - \sqrt{\frac{2}{\pi}} > 0.$$

3. The proof of strict properness in Theorem 3 with respect to $\mathcal{P}'$ boils down to the statement that for any independent X and X' following a probability distribution $P \in \mathcal{P}'$, and Y and Y' independent of each other and of X and X', following a probability distribution $Q \in \mathcal{P}'$, we have that:

$$2E(|X - Y|) - E(|X - X'|) - E(|Y - Y'|) =$$

$$\int (P(\{t|t \leq y\}) - Q(\{t|t \leq y\}))^2 dy,$$

and is therefore non-negative, and zero iff $P = Q$ a.s. An elementary proof can be found in pages 5 and 6 of Székely 2003.

In the case that $\Omega = \mathbb{R}^d$ and $\mathcal{A}$ is its associated Borel σ-algebra, we introduce the following new scoring rule.

Definition 4. For any choice of a measurable function $L : \mathbb{R}^d \to \mathbb{R}$ w.r.t to the Borel σ-algebras, define:

$$\text{CRPS}(L, P, x) := \text{CRPS}(L_*P, L(x)),$$

where L_*P is the pushforward measure of P by L.

It is trivial to verify that for every such choice of L, the function $\text{CRPS}(L, \cdot, \cdot)$ is a scoring rule. Also note that the scoring rule $CRPS - Sum - N$ from Appendix G.1 of Salinas et al. 2019a is none other than $\text{CRPS}(L, \cdot, \cdot)$ for L defined by

$$L(x_1, ..., x_d) = x_1 + ... + x_d.$$

In what follows we will not restrict to this choice of L.

Lemma S10. *The following equality holds for all probability measures P and Q:*

$$\text{CRPS}(L, P, Q) = \text{CRPS}(L_*P, L_*Q).$$

Proof. This boils down to the change-of-variables formula in measure theory:

$$\text{CRPS}(L, P, Q) = \int_{\mathbb{R}^d} \text{CRPS}(L_*P, L(x)) dQ =$$

$$\int_{\mathbb{R}} \text{CRPS}(L_*P, x) dL_*Q = \text{CRPS}(L_*P, L_*Q)$$

□

Therefore, we get the easy corollary:

Corollary S2. $\mathrm{CRPS}(L, \cdot, \cdot)$ is proper with respect to the Borel measurable sets.

Proof. For any two probabilty distributions on the Borel measurable sets on $\mathbb{R}^d$:

$$\mathrm{CRPS}(L, Q, P) = \mathrm{CRPS}(L_* Q, L_* P) \leq$$

$$\mathrm{CRPS}(L_* P, L_* P) = \mathrm{CRPS}(L, P, P).$$

□

If $d > 1$, then for any reasonable large convex set of probability measures $\mathcal{P}'$ (e.g., the class of probabilty measures with finite first moments), and for any $P \in \mathcal{P}'$, it is always possible to find multiple Q's in $\mathcal{P}'$ such that $L_* Q = L_* P$. Therefore $d > 1$ implies that $\mathrm{CRPS}(L, \cdot, \cdot)$ is proper but not strictly proper.