Proceedings of Machine Learning Research vol 144:1–14, 2021

Bridging Physics-based and Data-driven modeling for Learning Dynamical Systems

Rui Wang RUW020@UCSD.EDU
UC San Diego

Danielle Maddix DMMADDIX@AMAZON.COM
Amazon Research

Christos Faloutsos FALOUTSO@AMAZON.COM
Amazon and Carnegie Mellon University

Yuyang Wang YUYAWANG@AMAZON.COM
Amazon Research

Rose Yu ROSEYU@ENG.UCSD.EDU
UC San Diego

Abstract

How can we learn a dynamical system to make forecasts, when some variables are unobserved? For instance, in COVID-19, we want to forecast the number of infected and death cases but we do not know the count of susceptible and exposed people. While mechanics compartment models are widely-used in epidemic modeling, data-driven models are emerging for disease forecasting. We first formalize the learning of physics-based models as AutoODE, which leverages automatic differentiation to estimate the model parameters. Through a benchmark study on COVID-19 forecasting, we notice that physics-based mechanistic models significantly outperform deep learning. Such performance differences highlight the generalization problem in dynamical system learning due to distribution shift. We identify two scenarios where distribution shift can occur: changes in data domain and changes in parameter domain (system dynamics). Through systematic experiments on several dynamical systems, we found that deep learning models fail to forecast well under both scenarios. While much research on distribution shift has focused on changes in data domain, our work calls attention to rethink generalization for learning dynamical systems.

Keywords: Dynamical system, deep learning, generalization, COVID-19 forecasting.

1. Introduction

Dynamical systems (Strogatz, 2018) describe the evolution of phenomena occurring in nature, in which a differential equation, $d\boldsymbol{y}/dt = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{y}, t)$, models the dynamics of a d-dimensional state $\boldsymbol{y} \in \mathbb{R}^d$. Here $\boldsymbol{f}_{\boldsymbol{\theta}}$ is a non-linear operator parameterized by parameters $\boldsymbol{\theta}$. Learning dynamical systems is to search a good model for a dynamical system in the hypothesis space guided by some criterion for performance. In this work, we study the forecasting problem of predicting an sequence of future states $(\boldsymbol{y}_k, ..., \boldsymbol{y}_{k+q-1})$ given an sequence of historic states $(\boldsymbol{y}_0, ..., \boldsymbol{y}_{k-1})$.

A plethora of work has been devoted to learning dynamical systems. When $\boldsymbol{f}_{\boldsymbol{\theta}}$ is known, physics-based methods based on numerical integration are commonly used for parameter estimation (Houska et al., 2012). Raissi and Karniadakis (2018); Al-Aradi et al. (2018); Sirignano and Spiliopoulos (2018) propose to directly solve $\boldsymbol{y}$ by approximating $\boldsymbol{f}$ with neural networks that take the coordinates and time as input. When $\boldsymbol{f}_{\boldsymbol{\theta}}$ is unknown and the data is abundant, data-driven methods are preferred. For example, deep learning (DL), especially deep sequence models (Benidis et al., 2020; Sezer et al., 2019; Flunkert et al., 2017; Rangapuram et al., 2018) have demonstrated success in time series forecasting. In addition, Wang et al. (2020b); Ayed et al. (2019b); Wang et al. (2020c); Chen et al. (2018) have developed hybrid DL models based on differential equations for spatiotemporal dynamics forecasting.

Our study was initially motivated by the need to forecast COVID-19 dynamics. While mechanics compartment models are widely used in epidemic modeling, data-driven deep learning models are emerging for disease forecasting. We formalize automatic differentiation for estimating mechanistic model parameters, which we name as `AutoODE`. Specifically, we use numerical integration to generate state estimate and minimize the difference between estimate and the ground truth with automatic differentiation (Baydin et al., 2017; Paszke et al., 2017). We also propose a novel compartmental model, `ST-SuEIR`, which obtains a 57.4% reduction in mean absolute errors for 7-days ahead prediction. We perform a comprehensive benchmark study of various methods to forecast the cumulative number of confirmed, removed and death cases.[1] To our surprise, we found that physics-based models significantly outperform deep learning methods for COVID-19 forecasting.

To understand the inferior performance of DL, we experiment with several other dynamical systems: *Sine*, *SEIR*, *Lotka-Volterra* and *FitzHugh–Nagumo*. We observe that DL models suffer from distribution shift, often leading to poor generalization performance. In particular, two distribution shift scenarios may occur: changes in data distribution and changes in parameters that govern the dynamics of the system. Our findings highlight the unique challenge of using DL models for learning dynamical systems. To bridge the gap between physics-based and data-driven modeling, we need robust methods that can deal with both distribution shift scenarios for the forecasting task. To summarize, our contributions are the following:

1. We study dynamical systems for forecasting, even with unobserved variables. We formalize auto-differentiation for physics-based mechanistic models as `AutoODE` and propose a novel compartmental model for forecasting COVID-19. Our method obtains a 57.4% reduction in mean absolute error for 7-days ahead prediction compared to the best DL competitor.
2. We perform a benchmark study of physics-based mechanistic models and data-driven DL methods for predicting the COVID-19 dynamics among the cumulative confirmed, removed and death cases. We notice that physics-based models significantly outperform DL methods for COVID-19 forecasting, especially for the number of infected and removed cases.
3. We expand our study to other systems including *Lotka-Volterra*, *FitzHugh–Nagumo* and *SEIR* dynamics. We observe that the inferior performance of DL is mainly due to distribution shift. Many widely used DL models often fail to learn the correct dynamics when there is distribution shift either in the data or the parameters of the dynamical system.

2. Related Work

Learning Dynamical System The seminal work by Steven L. Brunton (2015) proposed to solve ODEs by creating a dictionary of possible terms and applying sparse regression to select appropriate terms. But it assumes that the chosen library is sufficient. Physics-informed deep learning directly solves differential equations with neural nets given space $\boldsymbol{x}$ and time t as input (Raissi and Karniadakis, 2018; Al-Aradi et al., 2018; Sirignano and Spiliopoulos, 2018). This type of methods cannot be used for forecasting since future t would always lie outside of the training domain and

[1] Reproducibility: We open-source our code https://github.com/Rose-STL-Lab/AutoODE-DSL; the COVID-19 data we use is from John Hopkins Dataset https://github.com/CSSEGISandData/COVID-19.

neural nets cannot extrapolate to unseen domain (Kouw and Loog, 2018; Amodei et al., 2019). Local methods, such as ARIMA and Gaussian SSMs (Salinas et al., 2019; Rasmussen and Williams., 2006; Du et al., 2016) learn the parameters individually for each time series. Hybrid DL models, e.g. Ayed et al. (2019b); Wang et al. (2020c); Chen et al. (2018); Ayed et al. (2019a) integrate differential equations in DL for temporal dynamics forecasting.

Deep Sequence Models Since accurate numerical computation requires lots of manual engineering and theoretical properties may have not been well understood, deep sequence models have been widely used for learning dynamical systems. Sequence to sequence models and the Transformer, have an encoder-decoder structure that can directly map input sequences to output sequences with different lengths (Vaswani et al., 2017; Wu et al., 2020; Li et al., 2020; Rangapuram et al., 2018; Flunkert et al., 2017). Fully connected neural networks can also be used autoregressively to produce multiple time-step forecasts (Benidis et al., 2020; Lim and Zohren, 2020). Neural ODE (Chen et al., 2018) is based on the assumption that the data is governed by an ODE system and able to generate continuous predictions. When the data is spatially correlated, deep graph models, such as graph convolution networks and graph attention networks (Velickovic et al., 2017), have also been used.

Epidemic Forecasting Compartmental models are commonly used for modeling epidemics. Chen et al. (2020) proposes a time-dependent SIR model that uses ridge regression to predict the transmission and recovery rates over time. A potential limitation with this method is that it does not consider the incubation period and unreported cases. Pei and Shaman (2020) modified the compartments in the SEIR model into the subpopulation commuting among different places, and estimated the model parameters using iterated filtering methods. Wang et al. (2020a) proposes a population-level survival-convolution method to model the number of infectious people as a convolution of newly infected cases and the proportion of individuals remaining infectious over time. Zou et al. (2020) proposes the `SuEIR` model that incorporates the unreported cases, and the effect of the exposed group on susceptibles. Davis et al. (2020) shows the importance of simultaneously modeling the transmission rate among the fifty U.S. states since transmission between states is significant.

3. Learning Dynamical Systems

3.1. Problem Formulation

Denote $\boldsymbol{y} \in \mathbb{R}^d$ as observed variables and $\boldsymbol{u} \in \mathbb{R}^p$ as the unobserved variables, we aim to learn a dynamical system given as

$$\begin{cases} \dfrac{d\boldsymbol{y}}{dt} = f_{\boldsymbol{\theta}}(t, \boldsymbol{y}, \boldsymbol{u}) \\ \dfrac{d\boldsymbol{u}}{dt} = g_{\boldsymbol{\theta}}(t, \boldsymbol{y}, \boldsymbol{u}) \\ \boldsymbol{y}(t_0) = \boldsymbol{y}_0 \\ \boldsymbol{u}(t_0) = \boldsymbol{u}_0. \end{cases} \tag{3.1}$$

In practice, we have observations $(\boldsymbol{y}_0, \boldsymbol{y}_1, ..., \boldsymbol{y}_{k-1})$ as inputs. The task of learning dynamical systems is to learn $f_{\boldsymbol{\theta}}$ and $g_{\boldsymbol{\theta}}$, and produce accurate forecasts $(\boldsymbol{y}_k, ..., \boldsymbol{y}_{k+q-1})$, where q is called the forecasting horizon.

3.2. Data-Driven Modeling

For data-driven models, we assume both f and g are unknown. We are given training and test samples either as sliced sub-sequences from a long sequence (same parameters, different initial conditions) or independent samples from the system (different parameters, same initial conditions). In particular, let $p_{\mathcal{S}}$ be the training data distribution and $p_{\mathcal{T}}$ be the test data distribution. DL seeks a hypothesis $h \in \mathcal{H} : \mathbb{R}^{d\times k} \mapsto \mathbb{R}^{d\times q}$ that maps a sequence of past values to future values:

$$h(\boldsymbol{y}_0^{(i)}, ..., \boldsymbol{y}_{k-1}^{(i)}) = \hat{\boldsymbol{y}}_k^{(i)}, ..., \hat{\boldsymbol{y}}_{k+q-1}^{(i)} \tag{3.2}$$

where (i) denotes individual sample, k is the input length and q is the output length.

Following the standard statistical learning setting, a deep sequence model minimizes the training loss $\hat{L_1}(h) = \frac{1}{n}\sum_{i=1}^{n} l(\boldsymbol{y}^{(i)}, h)$, where $\boldsymbol{y}^{(i)} = (\boldsymbol{y}_0^{(i)}, ..., \boldsymbol{y}_{k+q-1}^{(i)}) \sim p_{\mathcal{S}}$ is the i^{th} training sample, l is a loss function. For example, for square loss, we have

$$l(\boldsymbol{y}^{(i)}, h) = ||h(\boldsymbol{y}_0^{(i)}, ..., \boldsymbol{y}_{k-1}^{(i)}) - (\boldsymbol{y}_k^{(i)}, ..., \boldsymbol{y}_{k+q-1}^{(i)})||_2^2$$

The test error is given as $L_1(h) = \mathbb{E}_{\boldsymbol{y}\sim p_{\mathcal{T}}}[l(\boldsymbol{y}, h)]$. The goal is to achieve small test error $L_1(h)$ and small $|\hat{L_1}(h) - L_1(h)|$ indicates good generalization ability.

A fundamental difficulty of forecasting in dynamical system is the distributional shift that naturally occur in learning dynamical systems (Kouw and Loog, 2018; Amodei et al., 2019). In forecasting, the data in the future $p_{\mathcal{T}}$ often lie outside the training domain $p_{\mathcal{S}}$, and requires methods to extrapolate to the unseen domain. This is in contrast to classical machine learning theory, where generalization refers to model adapting to unseen data drawn from the same distribution (Hastie et al., 2009; Poggio et al., 2012).

3.3. Physics-Based Modeling

Physics-based modeling assumes we already have an appropriate system of ODEs to describe the underlying dynamics. We know the function f and g, but not the parameters. We can use automatic differentiation to estimate the unknown parameters $\boldsymbol{\theta}$ and the initial values $\boldsymbol{u}_0$. We coin this procedure as `AutoODE`. Similar approaches have been used in other papers (Rackauckas et al., 2020; Zou et al., 2020) but have not been well formalized. The main procedure is described in the algorithm 1. In the meanwhile, we need to ensure $\boldsymbol{u}$ and $\boldsymbol{y}$ have enough correlation that `AutoODE` can correctly learn all the parameters based on the observable $\boldsymbol{y}$ only. If $\boldsymbol{u}$ and $\boldsymbol{y}$ are not correlated or loosely correlated, we may be not able to estimate $\boldsymbol{u}$ solely based on the observations of $\boldsymbol{y}$.

Algorithm 1: AutoODE

```
0: Initialize the unknown parameters $\boldsymbol{\theta}, \boldsymbol{u}_0$ in Eqn. 3.1 randomly.
1: Discretize Eqn. 3.1 and apply 4-th order Runge Kutta (RK4) Method.
2: Generate estimation for $\boldsymbol{y}$: $(\hat{\boldsymbol{y}}_0, ..., \hat{\boldsymbol{y}}_k)$
3: Minimize the forecasting loss with the Adam optimizer,
   $L_2(\boldsymbol{\theta}, \boldsymbol{u}_0) = \frac{1}{k}\sum_{i=0}^{k-1} ||\hat{\boldsymbol{y}}_i(\boldsymbol{\theta}, \boldsymbol{u}, t) - \boldsymbol{y}_i(\boldsymbol{\theta}, \boldsymbol{u}, t)||^2$.
4: After convergence, use estimated $\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{u}_0}$ and 4-th order Runge Kutta Method to
   generate final prediction, $(\boldsymbol{y}_k, ..., \boldsymbol{y}_{t+q-1})$.
```

NeuralODE (Chen et al., 2018) uses the adjoint method to differentiate through the numerical solver. Adjoint methods are more efficient in higher dimensional neural network models which require complex numerical integration. In our case, since we are dealing with low dimension ordinary differential equations and the RK4 is sufficient to generate accurate predictions. We can directly implement the RK4 in Pytorch and make it fully differentiable.

4. Case study: COVID-19 Forecasting

We benchmark different methods for predicting the COVID-19 dynamics among the cumulative confirmed, removed and death cases. We propose an extension of the `SuEIR` compartmental model, and estimate the model parameters with `AutoODE`.

$$\begin{cases} dS_i/dt = -[\sum_j \beta_i(t)A_{ij}(I_j + E_j)S_i]/N_i, \\ dE_i/dt = [\sum_j \beta_i(t)A_{ij}(I_j + E_j)S_i]/N_i - \sigma_i E_i, \\ dU_i/dt = (1-\mu_i)\sigma_i E_i, \\ dI_i/dt = \mu_i \sigma_i E_i - \gamma_i I_i, \\ dR_i/dt = \gamma_i I_i, \\ dD_i/dt = r_i(t) dR_i/dt. \\ N_i = S_i + E_i + U_i + I_i + R_i \end{cases} \tag{4.1}$$

4.1. Spatiotemporal-SuEIR

We present the `Spatiotemporal-SuEIR (ST-SuEIR)` model given in Eqn. (4.1) and details are listed as below. We estimate the unknown parameters β_i, σ_i, μ_i, and γ_i, which correspond to the transmission, incubation, discovery, and recovery rates, respectively. We also need to estimate unobserved variables $\boldsymbol{u} = \{S, E, U\}$, the cumulative numbers of susceptibles, exposed and unreported cases, and predict the observed variables $\boldsymbol{y} = \{I, R, D\}$, cumulative numbers of infected, removed and death cases. The total population $N_i = S_i + E_i + U_i + I_i + R_i$ is assumed to be constant for each U.S. states i.

Low Rank Approximation to the Sparse Transmission Matrix A_{ij} We introduce a sparse transmission matrix A to model the transmission rate among the 50 U.S. states. A is the element-wise product of the U.S. states adjacency matrix M and the transmission matrix C, that is, $A = C \odot M \in \mathbb{R}^{50\times 50}$. M is a sparse 1-0 matrix that indicates whether two states are adjacent to each other. C learns the transmission rates among the all 50 states from the data. $A = C \odot M$ means that we omit the transmission between the states that are not adjacent to each other. We make C sparse with M as C contains too many parameters and we want to avoid overfitting. To further reduce the number of parameters and improve the computational efficiency to $\mathcal{O}(kn)$, we use a low rank approximation to generate the correlation matrix $C = B^T D$, where $B, D \in \mathbb{R}^{k\times n}$ for $k << n$.

Piecewise Linear Transmission Rate $\beta_i(t)$ Most compartmental models assume the transmission rate β_i is constant. For COVID-19, the transmission rate of COVID-19 changes over time due to government regulations, such as school closures and social distancing. Even though we focus on short-term forecasting (7 days ahead), it is possible that the transmission rate may change during the training period. Instead of a constant approximation to β_i, we use a piece-wise linear function over time $\beta_i(t)$, and set the breakpoints, slopes and biases as trainable parameters.

MAE	07/13 ∼ 07/19			08/23 ∼ 08/29			09/06 ∼ 09/12		
	I	R	D	I	R	D	I	R	D
FC	8379	5330	257	559	701	**30**	775	654	**33**
Seq2Seq	5172	2790	**99**	781	700	40	728	787	35
Transformer	8225	2937	2546	1282	1308	46	1301	1253	41
NeuralODE	7283	5371	173	682	661	43	858	791	35
GCN	6843	3107	266	1066	923	55	1605	984	44
GAT	4155	2067	153	1003	898	51	1065	833	40
SuEIR	1746	1984	136	639	778	39	888	637	47
ST-SuEIR	**818**	**1079**	109	**514**	**538**	41	**600**	**599**	39

Table 1: Proposed ST-SuEIR wins in predicting I and R: 7-day ahead prediction MAEs on COVID-19 trajectories of accumulated number of infectious, removed and death cases.

Death Rate Modeling: $r_i(t)$ The relationship between the numbers of accumulated removed and death cases can be close to linear, exponential or concave in different states. We assume the death rate $r_i(t)$ as a linear combination of $a_i t + b_i$ to cover both the convex and concave functions, where a_i and b_i are set as learnable parameters.

Weighted Loss Function We set the unknown parameters in Eqn. (4.1) as trainable, and apply AutoODE to minimize the following weighted loss function:

$$L(\boldsymbol{A},\boldsymbol{\beta},\boldsymbol{\sigma},\boldsymbol{\mu},\boldsymbol{\gamma},\boldsymbol{r}) = \frac{1}{k}\sum_{t=0}^{k-1} w(t)\Big[l(\hat{I}_t, I_t) + \alpha_1 l(\hat{R}_t, R_t) + \alpha_2 l(\hat{D}_t, D_t)\Big],$$

with weights α_1, α_2 and loss function $l(\cdot,\cdot)$ which we will specify in the following section. We utilize these weights to balance the loss of the three states due to scaling differences, and also reweigh the loss at different time steps. We give larger weights to more recent data points by setting $w(t) = \sqrt{t}$. The constants, α_1, α_2 and k are tuned on the validation set.

4.2. Experiments on forecasting COVID-19 Dynamics

We use the COVID-19 data from Apr 14 to Sept 12 provided by Johns Hopkins University (Dong et al., 2020). It contains the cumulative numbers of infected (I), recovered (R) and death (D) cases.

We investigate six DL models on forecasting COVID-19 trajectories: sequence to sequence with LSTMs (Seq2Seq), Transformer, autoregressive fully connected neural nets (FC), NeuralODE, graph convolution networks (GCN) and graph attention networks (GAT). We standardize I, R and D time series of each state individually to avoid one set of features dominating another. We use sliding windows to generate samples of sequences before the target week and split them into training and validation sets. We perform exhaustive search to tune the hyperparameters on the validation set.

We also compare SuEIR (Zou et al., 2020) and ST-SuEIR. We rescale the trajectories of the number of cumulative cases of each state by the population of that state. We use the quantile

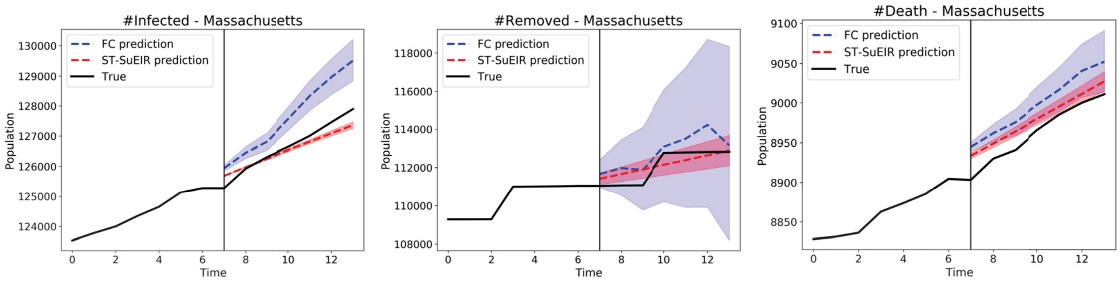

Figure 1: Proposed ST−SuEIR wins: I, R and D predictions for week $08/23 \sim 08/29$ in Massachusetts by our proposed ST−SuEIR and the best performing DL model FC.

regression loss (Wen et al., 2018) for all models. All the DL models are trained to predict the number of daily new cases instead of the number of cumulative cases because we want to detread the time series, and put the training and test samples in the same approximate range. All experiments were conducted on Amazon Sagemaker (Liberty et al., 2020).

Table 1 shows the 7-day ahead forecasting mean absolute errors of three features I, R and D for the weeks of July 13, Aug 23 and Sept 6. We can see that ST−SuEIR overall performs better than SuEIR and all the DL models. FC and Seq2Seq have better prediction accuracy of death counts but all DL models have much bigger errors on the prediction of week July 13. Figure 1 visualizes the 7-day ahead COVID-19 predictions of I, R and D in Massachusetts by ST−SuEIR and the best performing DL model, FC. The prediction by ST−SuEIR is closer to the target and has smaller confidence intervals. This demonstrates the effectiveness of our hybrid model, as well as the benefits of our novel compartmental model design.

5. Generalization in Learning Dynamical Systems

We explore the potential reasons behind the inadequate performance of DL models on COVID-19 forecasting. It is known that distribution shift often leads to poor generalization in DL. A model's performance deteriorates quickly when the test data distribution is different from training. We explore two distribution shift scenarios: changes in the data where the observation domain differs; and changes in parameters where the dynamics of the system in training and test differs.

5.1. Dynamical Systems

Apart from COVID-19 trajectories, we also investigate the following three non-linear dynamics.

Lotka-Volterra (*LV*) system (Ahmad, 1993) of Eqn.(5.1) describes the dynamics of biological systems in which predators and preys interact, where d denotes the number of species interacting and p_i denotes the population size of species i at time t. The unknown parameters $r_i \geq 0$, $k_i \geq 0$ and A_{ij} denote the intrinsic growth rate of species i, the carrying capacity of species i when the other species are absent, and the interspecies competition between two different species, respectively.

FitzHugh–Nagumo (*FHN*) FitzHugh (1961) and, independently, Nagumo et al. (1962) derived the Eqn.(5.2) to qualitatively describe the behaviour of spike potentials in the giant axon of squid neurons. The system describes the reciprocal dependencies of the voltage x across an axon membrane and a recovery variable y summarizing outward currents. The unknown parameters a, b, and c are dimensionless and positive, and c determines how fast y changes relative to x.

SEIR system of Eqn.(5.3) models the spread of infectious diseases (HE, 1992). It has four compartments: Susceptible (S) denotes those who potentially have the disease, Exposed (E) models the incubation period, Infected (I) denotes the infectious who currently have the disease, and Removed/Recovered (R) denotes those who have recovered from the disease or have died. The total population N is assumed to be constant and the sum of these four states. The unknown parameters β, σ and γ denote the transmission, incubation, and recovery rates, respectively.

$$\frac{dp_i}{dt} = r_i p_i \Big(1 - \frac{\sum_{j=1}^{d} A_{ij} p_j}{k_i}\Big), \quad i = 1, 2, \dots, d. \tag{5.1}$$

$$\begin{cases} \dfrac{dx}{dt} = c(x + y - \dfrac{x^3}{3}), \\ \dfrac{dy}{dt} = -\dfrac{1}{c}(x + by - a). \end{cases} \tag{5.2}$$

$$\begin{cases} dS/dt = -\beta SI/N, \\ dE/dt = \beta SI/N - \sigma E, \\ dI/dt = \sigma E - \gamma I, \\ dR/dt = \gamma I, \\ N = S + E + I + R. \end{cases} \tag{5.3}$$

5.2. Distribution Shift: Interpolation vs. Extrapolation

We define $p_{\mathcal{S}}$ and $p_{\mathcal{T}}$ as the training and the test data distributions. And the $\theta_{\mathcal{S}}$ and $\theta_{\mathcal{T}}$ denote *parameter* distributions of training and test sets, where the *parameter* here refers to the coefficients and the initial values of dynamical systems. A distribution is a function that map a sample space to the interval [0,1] if it a continuous distribution, or a subset of that interval if it is a discrete distribution. The domain of a distribution p, i.e. Dom(p), refers to the set of values (sample space) for which that distribution is defined.

We define two types of interpolation and extrapolation tasks. Regarding the data domain, we define a task as an interpolation task when the data domain of the test data is a subset of the domain of the training data, i.e., Dom$(p_{\mathcal{T}}) \subseteq$ Dom$(p_{\mathcal{S}})$, and then extrapolation occurs Dom$(p_{\mathcal{T}}) \nsubseteq$ Dom$(p_{\mathcal{S}})$. Regarding the *parameter* domain, an interpolation task indicates that Dom$(\theta_{\mathcal{T}}) \subseteq$ Dom$(\theta_{\mathcal{S}})$, and an extrapolation task indicates that Dom$(\theta_{\mathcal{T}}) \nsubseteq$ Dom$(\theta_{\mathcal{S}})$. Many machine learning setups focus on the interpolation tasks. The extrapolation tasks correspond to the situations with distribution shift.

5.3. Scenario 1: unseen data in the different data domain

Through a simple experiment on learning the *Sine* curves, we show deep sequence models have poor generalization on extrapolation tasks regarding the data domain, i.e. Dom$(p_{\mathcal{T}}) \nsubseteq$ Dom$(p_{\mathcal{S}})$. Specifically, we generate 2k *Sine* samples of length 60 with different frequencies and phases, and randomly split them into training, validation and interpolation-test sets. The extrapolation-test set is the interpolation-test set shifted up by 1. We investigate four models, including `Seq2Seq` (sequence to sequence with LSTMs), `Transformer`, `FC` (autoregressive fully connected neural nets) and `NeuralODE`. All models are trained to make 30 steps ahead prediction given the previous 30 steps.

Table 2 shows that all models have substantially larger errors on the extrapolation test set. Figure 2 shows `Seq2Seq` predictions on an interpolation (left) and an extrapolation (right) test samples. We can see that `Seq2Seq` makes accurate predictions on the interpolation-test sample, while it fails to generalize when the same samples are shifted up only by 1.

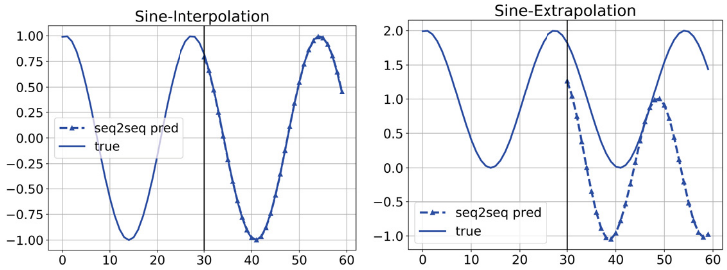

Figure 2: `Seq2Seq` predictions on an interpolation (left) and an extrapolation (right) test samples of Sine dynamics, the vertical black line in the plots separates the input and forecasting period.

RMSE	`Inter`	`Extra`
`Seq2Seq`	0.012	1.242
`Auto-FC`	0.009	1.554
`Transformer`	0.016	1.088
`NeuralODE`	0.012	1.214

Table 2: RMSEs of the interpolation and extrapolation tasks of Sine dynamics.

5.4. Scenario 2: unseen data with different system parameters

Even when $\text{Dom}(p_{\mathcal{T}}) \subseteq \text{Dom}(p_{\mathcal{S}})$, deep sequence models can still fail to learn the correct dynamics if there is a distributional shift in the parameter domain, i.e., $\text{Dom}(\theta_{\mathcal{T}}) \not\subseteq \text{Dom}(\theta_{\mathcal{S}})$. For example, the recovery rate γ in the epidemic may increase as more people are vaccinated. A tighter social distancing policy would lead to smaller contact rate β. Thus, there is a great chance that new test samples are outside of the parameter domain of training data. In that case, the DL models would not make accurate prediction for COVID-19.

For each of the three dynamics in section 5.1, we generate 6k synthetic time series samples with different system parameters and initial values. The training/validation/interpolation-test sets for each dataset have the same range of system parameters while the extrapolation-test set contains samples from a different range. Table 3 shows the *parameter* distribution of test sets. For each dynamics, we perform two experiments to evaluate the models' extrapolation generalization ability on initial values and system parameters. All samples are normalized so that $\text{Dom}(p_{\mathcal{T}}) = \text{Dom}(p_{\mathcal{S}})$.

	System Parameters		Initial Values	
	Interpolation	Extrapolation	Interpolation	Extrapolation
LV	$\boldsymbol{k} \sim U(0, 250)^4$	$\boldsymbol{k} \sim U(250, 300)^4$	$\boldsymbol{p_0} \sim U(30, 200)^4$	$\boldsymbol{p_0} \sim U(0, 30)^4$
FHN	$c \sim U(1.5, 5)$	$c \sim U(0.5, 1.5)$	$x_0 \sim U(2, 10)$	$x_0 \sim U(0, 2)$
SEIR	$\beta \sim U(0.45, 0.9)$	$\beta \sim U(0.3, 0.45)$	$I_0 \sim U(30, 100)$	$I_0 \sim U(10, 30)$

Table 3: The initial values and system parameters ranges of interpolation and extrapolation test sets.

Table 4 shows the prediction RMSEs of the models on initial values and system parameter interpolation and extrapolation test sets. We observe that the models' prediction errors on extrapolation test sets are much larger than the error on interpolation test sets. Figures 3-4 show that `Seq2Seq` and `FC` fail to make accurate prediction when tested outside of the parameter distribution even though they make accurate predictions for parameter interpolation test samples.

RMSE	*LV*				*FHN*				*SEIR*			
	$\boldsymbol{k}$		$\boldsymbol{p_0}$		c		x_0		β		I_0	
	Int	Ext	Int	Ext	Int	Ext	Int	Ext	Int	Ext	Int	Ext
Seq2Seq	**0.050**	0.215	0.028	0.119	0.093	0.738	0.079	0.152	1.12	4.14	2.58	7.89
FC	0.078	0.227	0.044	0.131	**0.057**	0.402	**0.057**	0.120	1.04	3.20	1.82	5.85
Transformer	0.074	0.231	0.067	0.142	0.102	0.548	0.111	0.208	1.09	4.23	2.01	6.13
NeuralODE	0.091	0.196	0.050	0.127	0.163	0.689	0.124	0.371	1.25	3.27	2.01	5.82
AutoODE	0.057	**0.054**	**0.018**	**0.028**	0.059	**0.058**	0.066	**0.069**	**0.89**	**0.91**	**0.96**	**1.02**

Table 4: RMSEs on initial values and system parameter interpolation and extrapolation test sets.

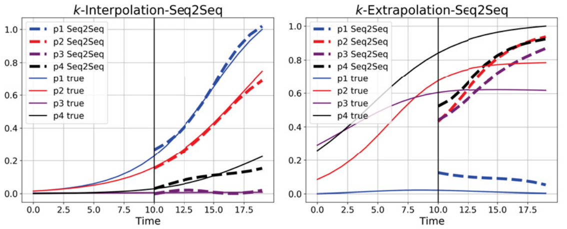

Figure 3: Seq2Seq predictions on a $\boldsymbol{k}$-interpolation and a $\boldsymbol{k}$-extrapolation test samples of *LV* dynamics, the vertical black line separates the input and forecasting period.

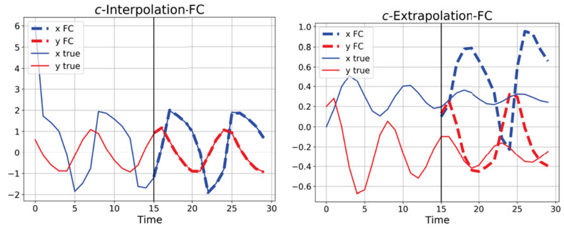

Figure 4: FC predictions on a c-interpolation and a c-extrapolation test samples of *FHN* dynamics, the vertical black line in the plots separates the input and forecasting period.

AutoODE always obtains the lowest errors as it would be not affected by the range of parameters or the initial values. However, it is a local method and we need to train one model for each sample. In contrast, DL models can only mimic the behaviors of SEIR, LV and FHN dynamics rather than understanding the underlying mechanisms.

6. Conclusion

We study the problem of forecasting non-linear dynamical systems. From a benchmark study of DL and physics-based models, we find that FC and Seq2Seq have better prediction accuracy on the number of deaths, while the DL methods generally much worse than the physics-based models on the number of infected and removed cases. This is mainly due to the distribution shift in the COVID-19 dynamics. On several other non-linear dynamical systems, we experimentally show that four DL models fail to generalize under shifted distributions in both the data and the parameter domains. Even though these models are powerful enough to memorize the training data, and perform well on the interpolation tasks. Our study provides important insights on learning real world dynamical systems: to achieve accurate forecasts with DL, we need to ensure that both the data and dynamical system parameters in the training set can cover the domains of the test set. Future works include incorporating compartmental models into deep learning models and derive theoretical generalization bounds of dynamics forecasting for deep learning models.

References

S. Ahmad. On the nonautonomous volterra-lotka competition equations. 1993.

Ali Al-Aradi, Adolfo Correia, Danilo Naiff, Gabriel Jardim, and Yuri Saporito. Solving nonlinear and high-dimensional partial differential equations via deep learning. *arXiv preprint arXiv:1811.08782*, 2018.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2019.

I. Ayed, Emmanuel de Bézenac, A. Pajot, J. Brajard, and P. Gallinari. Learning dynamical systems from partial observations. *ArXiv*, abs/1902.11136, 2019a.

Ibrahim Ayed, Emmanuel De Bézenac, Arthur Pajot, and Patrick Gallinari. Learning partially observed PDE dynamics with neural networks, 2019b. URL https://openreview.net/forum?id=HyefgnCqFm.

Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Radul, and J. Siskind. Automatic differentiation in machine learning: a survey. *ArXiv*, abs/1502.05767, 2017.

Konstantinos Benidis, Syama Sundar Rangapuram, V. Flunkert, Bernie Wang, Danielle C. Maddix, A. Türkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, L. Stella, L. Callot, and Tim Januschowski. Neural forecasting: Introduction and literature overview. *ArXiv*, abs/2004.10240, 2020.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6571–6583. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7892-neural-ordinary-differential-equations.pdf.

Yi-Cheng Chen, Ping-En Lu, Cheng-Shang Chang, and Tzu-Hsuan Liu. A time-dependent sir model for covid-19 with undetectable infected persons. *arXiv preprint arXiv:2003.00122*, 2020.

Jessica T Davis, Matteo Chinazzi, Nicola Perra, Kunpeng Mu, Ana Pastore y Piontti, Marco Ajelli, Natalie E Dean, Corrado Gioannini, Maria Litvinova, Stefano Merler, Luca Rossi, Kaiyuan Sun, Xinyue Xiong, M. Elizabeth Halloran, Ira M Longini Jr., Cécile Viboud, and Alessandro Vespignani. Estimating the establishment of local transmission and the cryptic phase of the covid-19 pandemic in the usa. *medRXiv preprint https://doi.org/10.1101/2020.07.06.20140285*, 2020.

Ensheng Dong, Hongru Du, and Lauren Gardner. An interactive web-based dashboard to track covid-19 in real time. *Lancet Inf Dis.*, 20(5):533–534. doi:10.1016/S1473–3099(20)30120–1, 2020. doi: 10.1016/S1473-3099(20)30120-1. URL https://github.com/CSSEGISandData/COVID-19.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Richard FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophyiscal Journal.*, 1:445–466, 1961. doi: https://doi.org/10.1016/S0006-3495(61)86902-6.

V. Flunkert, David Salinas, and Jan Gasthaus. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *ArXiv*, abs/1704.04110, 2017.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Springer, 2009.

Tillett HE. Infectious diseases of humans; dynamics and control. *Epidemiol Infect*, 1992.

B. Houska, F. Logist, M. Diehl, and J. Van Impe. A tutorial on numerical methods for state and parameter estimation in nonlinear dynamic systems. In D. Alberer, H. Hjalmarsson, and L. Del Re, editors, *Identification for Automotive Systems, Volume 418, Lecture Notes in Control and Information Sciences*, page 67–88. Springer, 2012.

Wouter M. Kouw and Marco Loog. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*, 2018.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *arXiv preprint arXiv:1907.00235*, 2020.

Edo Liberty, Zohar Karnin, Bing Xiang, Laurence Rouesnel, Baris Coskun, Ramesh Nallapati, Julio Delgado, Amir Sadoughi, Yury Astashonok, Piali Das, Can Balioglu, Saswata Chakravarty, Madhav Jha, Philip Gautier, David Arpin, Tim Januschowski, Valentin Flunkert, Yuyang Wang, Jan Gasthaus, Lorenzo Stella, Syama Rangapuram, David Salinas, Sebastian Schelter, and Alex Smola. Elastic machine learning algorithms in amazon sagemaker. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, page 731–737, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450367356. doi: 10.1145/3318464.3386126. URL https://doi.org/10.1145/3318464.3386126.

Bryan Lim and Stefan Zohren. Time series forecasting with deep learning: A survey. *ArXiv*, abs/2004.13408, 2020.

Jinichi Nagumo, Suguru Arimoto, and Shuji Yoshizawa. An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070, 1962.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. *NIPS 2017 Autodiff Workshop*, 2017.

Sen Pei and Jeffrey Shaman. Initial simulation of sars-cov2 spread and intervention effects in the continental us. *medRXiv preprint https://doi.org/10.1101/2020.03.21.20040303*, 2020.

Tomaso Poggio, Lorenzo Rosasco, Charlie Frogner, and Guille D. Canas. Statistical learning theory and applications. 2012.

Christopher Rackauckas, Y. Ma, Julius Martensen, Collin Warner, K. Zubov, Rohit Supekar, D. Skinner, and Ali Ramadhan. Universal differential equations for scientific machine learning. *ArXiv*, abs/2001.04385, 2020.

Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

Syama Sundar Rangapuram, Matthias W. Seeger, Jan Gasthaus, L. Stella, Y. Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *NeurIPS*, 2018.

Carl Edward Rasmussen and Christopher KI Williams. Gaussian process for machine learning. *MIT press*, 2006.

David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processe. In *In Advances in Neural Information Processing Systems 32.*, 2019.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning : A systematic literature review: 2005-2019. *arXiv preprint arXiv:1911.13288*, 2019.

Justin Sirignano and Konstantinos Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. *arXiv preprint arXiv:1708.07469*, 2018.

J. Nathan Kutz Steven L. Brunton, Joshua L. Proctor. Discovering governing equations from data: Sparse identification of nonlinear dynamical systems. *arXiv preprint arXiv:1509.03580*, 2015.

Steven H. Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC press, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, 2017.

Petar Velickovic, Guille mCucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Qinxia Wang, Shanghong Xie, Yuanjia Wang, and Donglin Zeng. Survival-convolution models for predicting covid-19 cases and assessing effects of mitigation strategies. *medRXiv preprint https://doi.org/10.1101/2020.04.16.20067306*, 2020a.

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery and data mining*, 2020b.

Rui Wang, Robin Walters, and Rose Yu. Incorporating symmetry into deep dynamics models for improved generalization. *arXiv preprint arXiv:2002.03061*, 2020c.

Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2018.

Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint arXiv:2001.08317*, 2020.

Difan Zou, Lingxiao Wang, Pan Xu, Jinghui Chen, Weitong Zhang, and Quanquan Gu. Epidemic model guided machine learning for covid-19 forecasts in the united states. *medRXiv preprint https://doi.org/10.1101/2020.05.24.20111989*, 2020.