Imperial College London

Department of Aeronautics

Physics-informed LSTM for Lorenz Equation

Author:
Elise Özalp

Supervisor:
Dr. Luca Magri

May 1, 2022

Chapter 1: Lorenz Equation

$$\frac{dx}{dt} = \sigma(y - x) \tag{1.1}$$
$$\frac{dy}{dt} = x(\rho - z) - y \tag{1.2}$$
$$\frac{dz}{dt} = xy - \beta z \tag{1.3}$$

- $\sigma = 10, \beta = 8/3, \rho = 28$
- Initial condition $(0, 1, 1)$

For the sake of brevity, we write (1.1) as

$$u_t = \mathcal{N}(u) \tag{1.4}$$

where $\mathcal{N}$ is our non-linear differential operator.

For the Lorenz system with the given parameters, the **Lyapunov exponent** is given by $\approx \mathbf{0.90566}$, resulting in a **Lyapunov time** of $\approx \mathbf{1.1042}$.

1.1 Training Data

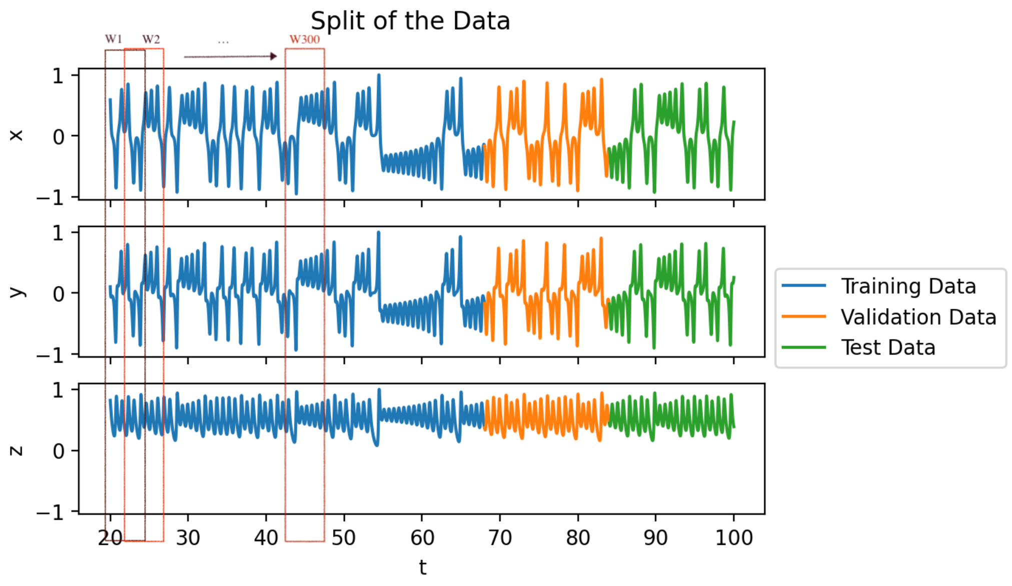


Figure 1.1: We removed the transient time $[0, 20]$. With a step size of 0.01, this dataset contains 8000 data points. We normalized the data and split it using a $60 - 20 - 20$ split. For training, we split the data in time windows, see section below.

Chapter 2: Architectures

2.1 Many-to-one

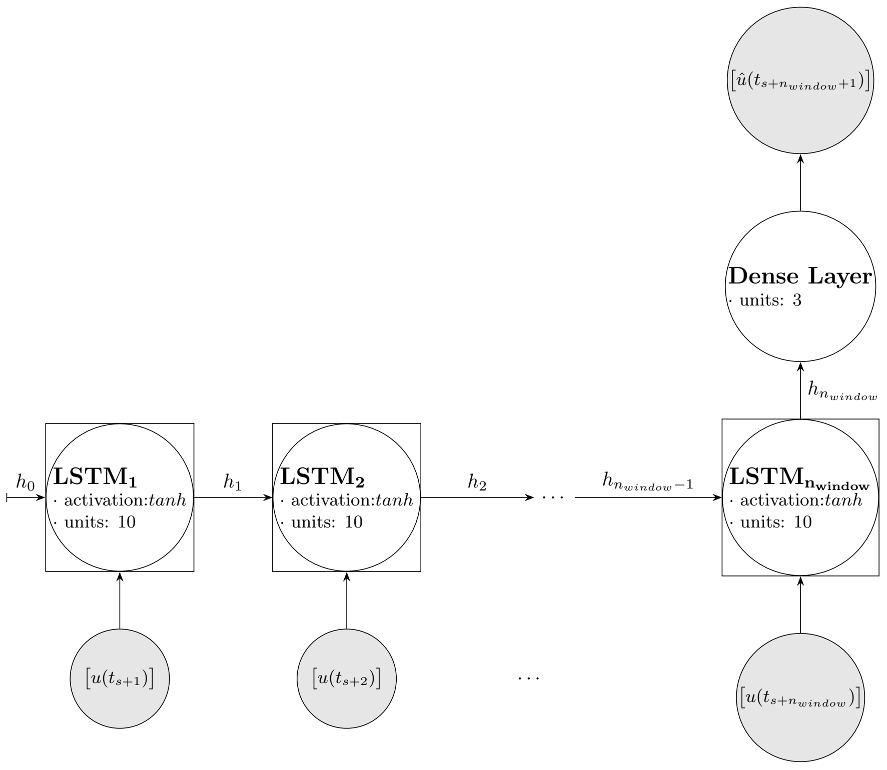


Figure 2.1: Many-to-one training. While each epoch is quicker, it takes longer for the network to reach a certain accuracy. Very sensitive on the window length but can achieve similar results to many-to-many.

2.2 Many-to-many: Data-driven Loss

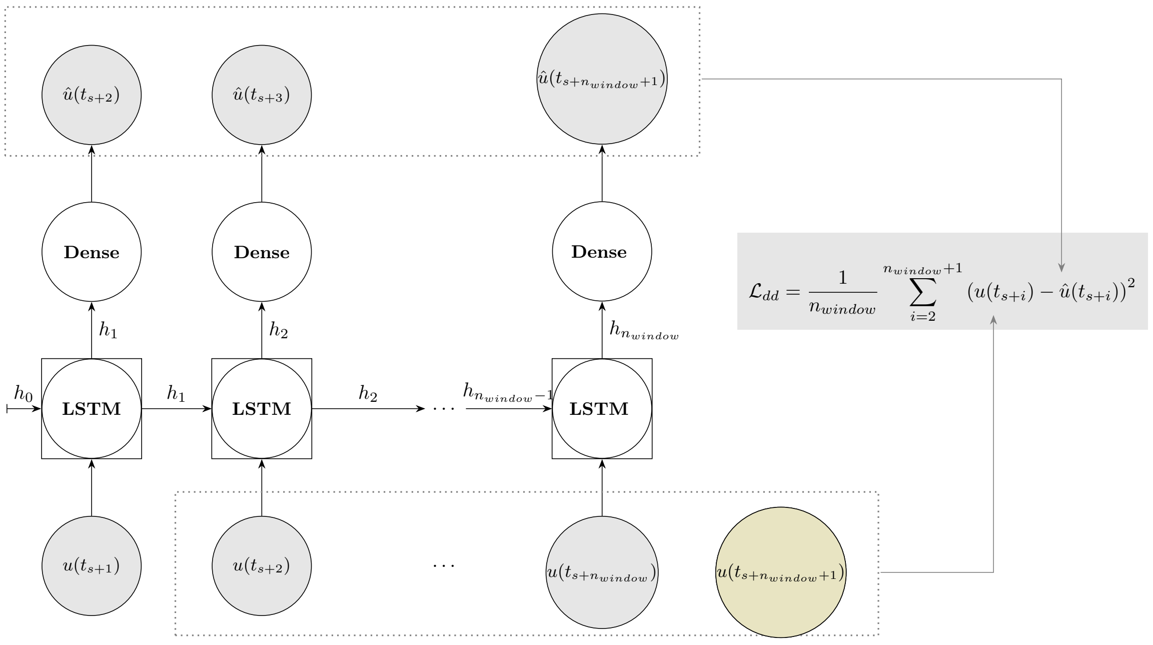


Figure 2.2: Many-to-many training data-driven only. Quick training, not as dependent on window length. Still kept window length 100 to capture around 1 LT.

2.3 Physics-informed Loss

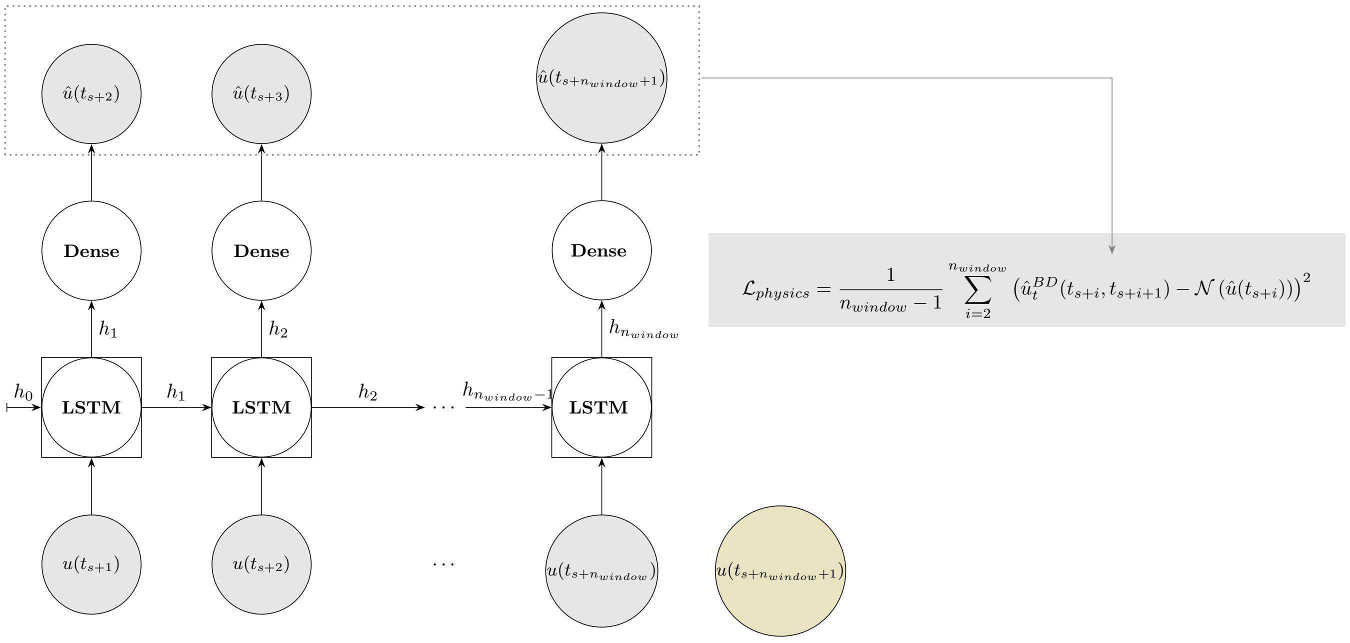


Figure 2.3: For the physics-informed loss in the many-to-many architecture, we only use the predictions to compute the loss. Trough the architecture, we automatically have many predictions.

Physics-informed Finite-Difference

Alternatively, we can use only the network outputs to compute a *forward difference scheme*. For this approach, we need to compute input the first prediction back into the network to receive the next prediction step

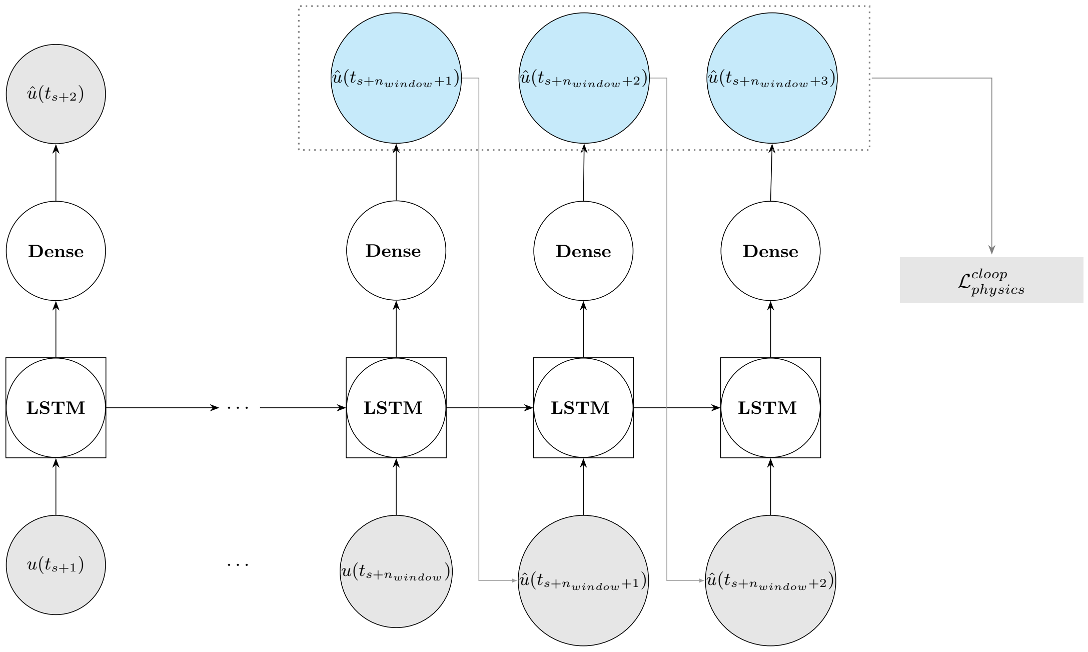
$$\frac{d}{dt}\hat{u}(t_{i+1}) \approx \frac{\overbrace{\hat{u}(t_{i+2})}^{\text{two step network prediction}} - \overbrace{\hat{u}(t_{i+1})}^{\text{one step network prediction}}}{\Delta t}. \tag{2.1}$$

Physics-informed Loss

This will then result in the *physics-informed* loss

$$\mathcal{L}_{physics} = \frac{1}{N_{batch}}\sum_{i=0}^{N_{batch}}\left(\underbrace{\hat{u}_t(t_{i+window+1})}_{\text{FD approx}} - \underbrace{\mathcal{N}(\hat{u}(t_{i+window+1}))}_{\text{prediction}}\right)^2. \tag{2.2}$$

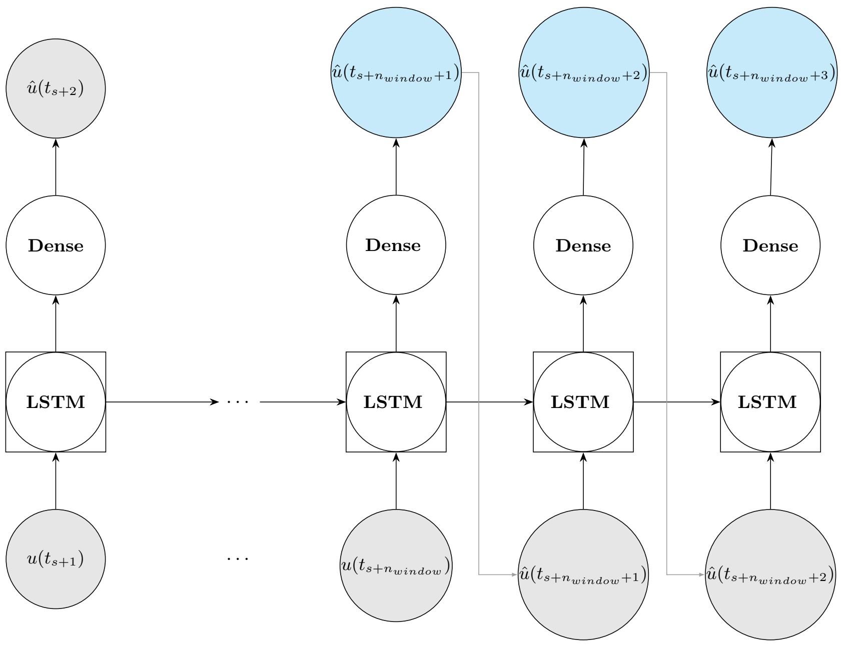
2.4 Closed-loop Physics-informed Training



$\hat{u}(t_{s+2})$
$\hat{u}(t_{s+n_{window}+1})$
$\hat{u}(t_{s+n_{window}+2})$
$\hat{u}(t_{s+n_{window}+3})$
Dense
Dense
Dense
Dense
$\mathcal{L}^{cloop}_{physics}$
LSTM
LSTM
LSTM
LSTM
$u(t_{s+1})$
$u(t_{s+n_{window}})$
$\hat{u}(t_{s+n_{window}+1})$
$\hat{u}(t_{s+n_{window}+2})$

Figure 2.4: Alternatively, we can also train using a multiple-step prediction. Then, we can easily compute the physics loss on between these predictions. I varied different number of prediction steps. The results always become worse than just training data-driven (physics) as in Section 2.2.

2.5 Prediction for Test Data



$\hat{u}(t_{s+2})$
$\hat{u}(t_{s+n_{window}+1})$
$\hat{u}(t_{s+n_{window}+2})$
$\hat{u}(t_{s+n_{window}+3})$
Dense
Dense
Dense
Dense
LSTM
LSTM
LSTM
LSTM
$u(t_{s+1})$
$u(t_{s+n_{window}})$
$\hat{u}(t_{s+n_{window}+1})$
$\hat{u}(t_{s+n_{window}+2})$

Figure 2.5: To evaluate the performance of the network, we used a closed-loop approach. Using the first n_{window} steps of the test data, we continue with a multi-step prediction. By inputting previous predictions, we can predict as many time steps as we want.

We then calculate the loss by taking

$$\mathcal{L} = \mathcal{L}_{dd} + \mathcal{L}_{physics} + \mathcal{L}^{cloop}_{physics} \tag{2.3}$$

Alternative Label to Avoid Normalization

As a suggestion to avoid the normalization and the scaling, we can rewrite the desired prediction

$$u(t_{n+window+1}) = u(t_{n+window}) + \delta(t_{n+window+1}).$$

Then we can use the following input with label

$$[u(t_n), \dots, u(t_{n+window})] \longrightarrow [\delta(t_{n+window+1})] .$$

As a result, we can normalize the data but we do not have to rescale the physics-loss.

Chapter 3: Measures

- u true/numerical solution; $\hat{u}$ network solution

3.1 Prediction Horizon

The prediction horizon is given by $t_{N_{PH}}$ s.t.

$$\frac{\|u(t_{N_{PH}}) - \hat{u}(t_{N_{PH}})\|}{\sqrt{\frac{1}{N_{PH}} \sum_{i=test_0}^{N_{PH}} \|u(t_i)\|^2}} < k. \tag{3.1}$$

This allows us to analyze if the network shows improvements on the test e.g. throughout different epoch evaluations. Ideally, we would like to see the prediction horizon increasing until it eventually stabilizes, e.g. However, choosing an appropriate threshold k is still unclear.

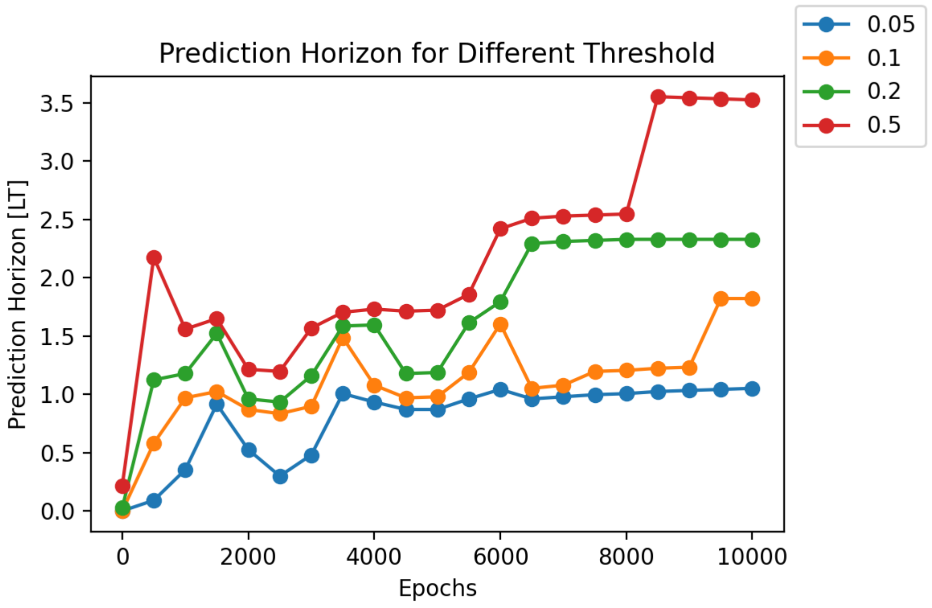


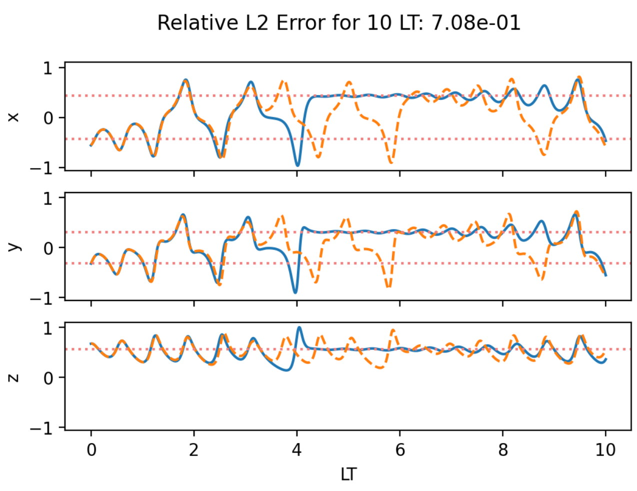
Figure 3.1: Prediction Horizon for many-to-many data-driven training.

3.2 Relativ $\mathcal{L}_2$ Error

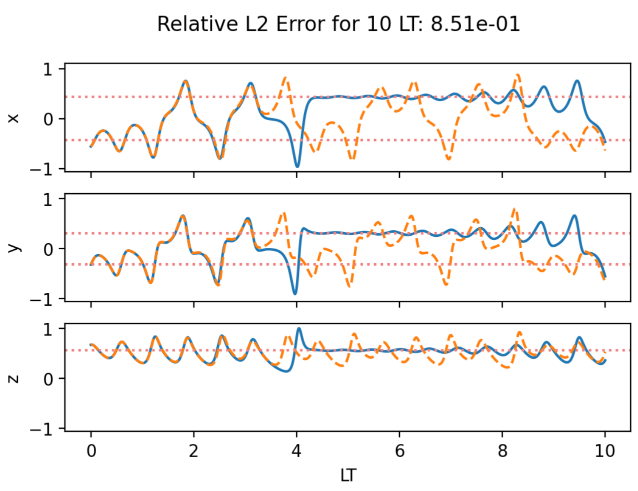
For a fixed prediction horizon $t_{N_{PH}}$, compute the relative $\mathcal{L}_2$ error

$$\epsilon(t_{N_{PH}}) = \frac{\|u(t_i) - \hat{u}(t_i)\|_2}{\|u(t_i)\|_2}_{i=0,\ldots,N_{PH}}. \tag{3.2}$$

By evaluating the relative $\mathcal{L}_2$ error on a fixed time span, we can compare which prediction is closer to the numerical error. This information is not directly contained in the prediction horizon measure.



(a) Data-driven only network



(b) Additional training epochs using the physics-informed loss

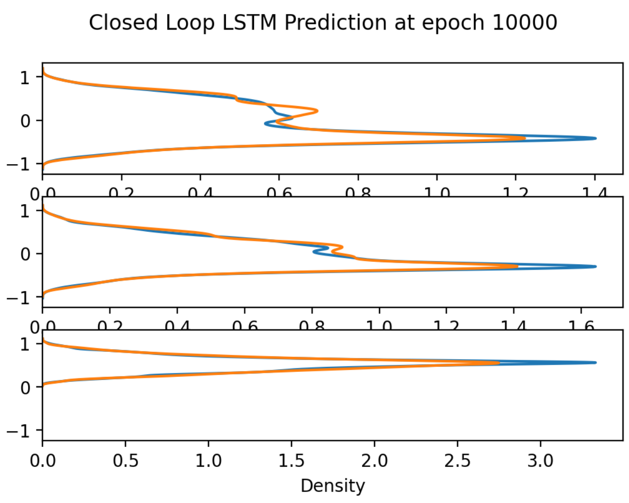
Figure 3.2: Comparison of rel $\mathcal{L}_2$ error. The blue line is the numerical solution, the dashed orange line

3.3 Visual Comparison Phase Space/ Signal

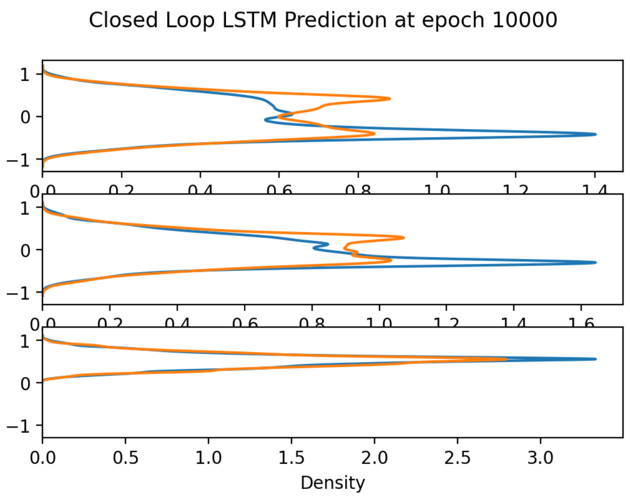
Through the visual comparison of the phase space, we check that the network prediction respect the physical chaotic behaviour, i.e. the solution lies on the butterfly shape.

3.4 Visual Comparison of Density Estimates

For a very long time interval (e.g. > 100LT), we can observe the density estimate of the network prediction and the numerical solution.



(a) Data-driven only network



(b) Additional training epochs using the physics-informed loss

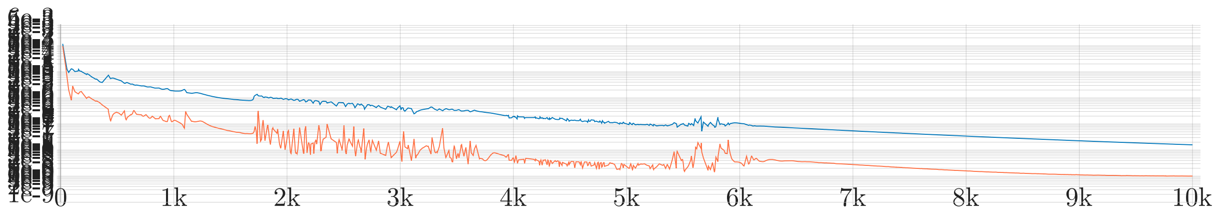
Figure 3.3: Comparison of density estimates over $\approx 100LT$.

Chapter 4: Key Observations from Comparison

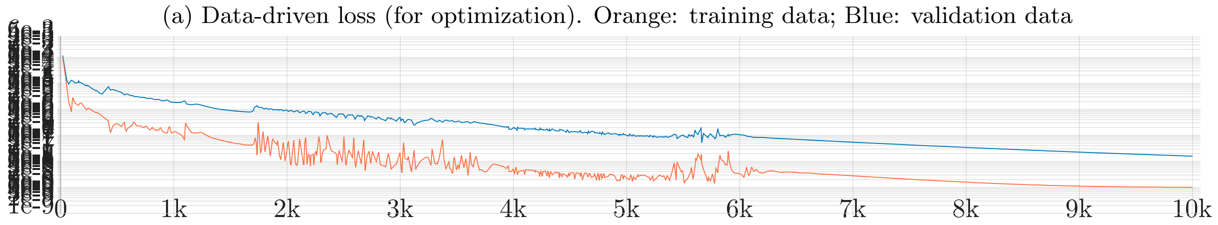
4.1 Data-driven Many-to-Many LSTM

$\mathcal{L} = \mathcal{L}_{data-driven}$, see Section 2.2

- achieves very small loss;
- also minimizes physics loss very well;
- good approximation $3-4$ LT, good looking attractor.



(a) Data-driven loss (for optimization). Orange: training data; Blue: validation data



(b) Physics loss (tracking without optimization). Orange: training data; Blue: validation data

Figure 4.1: Losses data-driven many-to-many LSTM

Despite only minimizing the data-driven loss, the network automatically also minimizes the physics loss really well. I assume this is because at every cell, the network approximates the corresponding time step really well and therefore the physics are also very accurate.

If another network architecture performs better, we expect the following:

- a lower relative $\mathcal{L}_2$ error [1];
- a higher prediction horizon for a given threshold (3.1).

4.2 Closed-Loop Prediction

Following the approach from Section 2.4, we pre-train the network using the data-driven approach and then perform the training using a physics-informed closed loop loss. To be precise, we use

$$\mathcal{L} = \mathcal{L}_{data-driven} + \mathcal{L}_{cloop}, \tag{4.1}$$

where $\mathcal{L}_{cloop}$ is dependent on the number of closed loop steps s. Observations

- larger s creates flatter prediction;
- larger s results in bad approximation;
- only 1 closed loop step might help.

number of cloop steps s	0 (data-driven only)	1	2	5	10
relative $\mathcal{L}_2$ error	$6.52e-01$	$7.56e-01$	$9.50e-01$	$1.00e+00$	$8.90e-01$
prediction horizon, k (3.1) $= 0.2$	2.33	3.53	0.41	0.18	0.14
prediction horizon, k (3.1) $= 0.5$	3.52	3.53	0.41	0.18	0.14

Table 4.1: The relative $\mathcal{L}_2$ error increases with all closed loop steps. The prediction horizon improved for $s = 1$ but otherwise significantly decreased.

From these results, I deduce that only a closed loop steep $s = 1$ may help. It improved the prediction horizon but not the relative $\mathcal{L}_2$ error.

[1] Visually, the networks predict well for $3-4$ LT. To capture this time span, we choose 5 LT.

4.3 Physics-informed LSTM

In this comparison, we tested training with the physics-informed loss from the beginning (no preinitialization). For this, we use the loss

$$\mathcal{L} = \mathcal{L}_{data-driven} + \lambda_{physics} \cdot \mathcal{L}_{physics}. \tag{4.2}$$

We iterate over $\lambda_{physics}$ and train each network for 10000 epochs. Observations:

- no consistent improvements using physics;
- improvements for $\lambda_{physics} = 1$, random?

$\lambda_{physics}$	0 (= no physics)	1	0.1	0.01	0.001
relative $\mathcal{L}_2$ error	$6.52e-01$	$4.70e-01$	$6.58e-01$	$6.52e-01$	$6.58e-01$
prediction horizon, k (3.1) $= 0.2$	2.33	3.49	2.31	2.33	2.31
prediction horizon, k (3.1) $= 0.5$	3.52	3.61	2.54	3.52	2.54

Table 4.2: Training directly with data-driven and physics loss

4.4 Preloaded Physics-informed LSTM

In this comparison, we tested training with the physics-informed loss with preinitialization. For this, we use the loss

$$\mathcal{L} = \mathcal{L}_{data-driven} + \lambda_{physics} \cdot \mathcal{L}_{physics}. \tag{4.3}$$

We iterate over $\lambda_{physics}$ and train each network for 10000 epochs.

- slight improvement for $\lambda_{physics} = 0.1$, not consistent.

$\lambda_{physics}$	0 (= no physics)	1	0.1	0.01	0.001
relative $\mathcal{L}_2$ error	$6.52e-01$	$8.84e-01$	$6.38e-01$	$6.88e-01$	$7.50e-01$
threshold k (3.1) $= 0.2$	2.33	1.81	3.35	3.41	3.50
threshold k (3.1) $= 0.5$	3.52	2.29	3.50	3.54	3.60

Table 4.3: Pre-initialized physics-informed LSTM

4.5 Noisy Data - Physics-informed LSTM

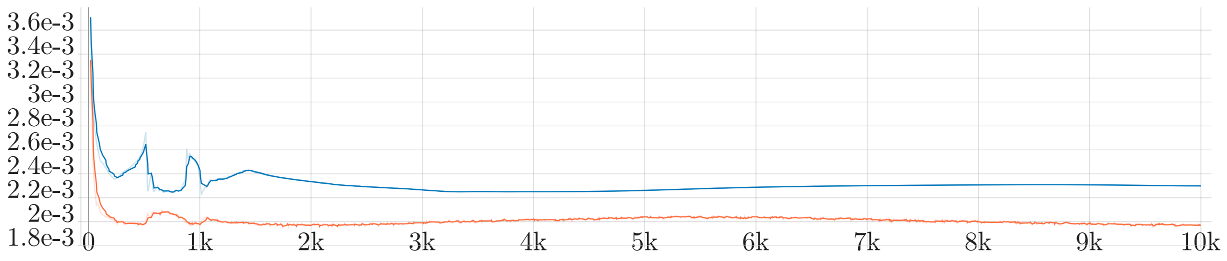
$$\mathcal{L} = \mathcal{L}_{data-driven} + \lambda_{physics} \cdot \mathcal{L}_{physics}. \tag{4.4}$$

We added noise to the data above based on a certain signal-to-noise ratio. Here, we measure the relative $\mathcal{L}_2$ error for only 2 LT since the prediction is worse.

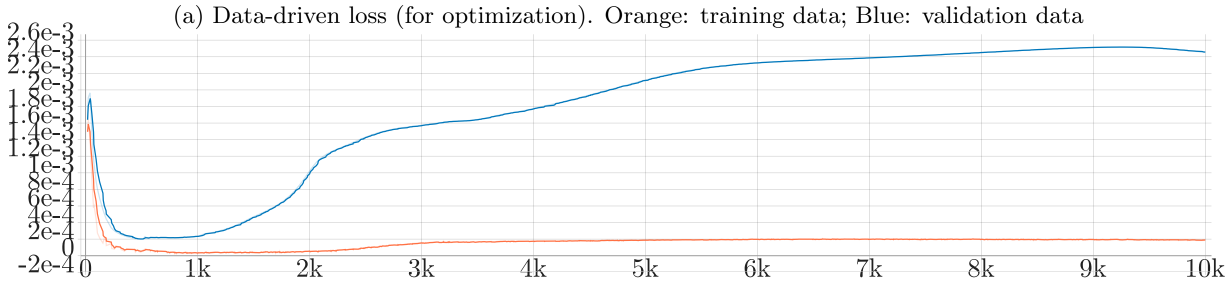
- no consistent improvements with physics,
- data-driven only approach fails with physics on validation data.

$\lambda_{physics}$	0 (= no physics)	1	0.1	0.01	0.001
relative $\mathcal{L}_2$ error	$4.52e-1$	$1.11e+0$	$1.02e+0$	$1.11e+0$	$1.03e+0$
threshold k (3.1) $= 0.2$	1.02	0.90	0.95	0.91	0.98
threshold k (3.1) $= 0.5$	0.45	0.74	0.52	0.53	0.47

Table 4.4: Signal-to-noise ratio: 30



(a) Data-driven loss (for optimization). Orange: training data; Blue: validation data



(b) Physics loss (tracking without optimization). Orange: training data; Blue: validation data

Figure 4.2: Losses of data-driven many-to-many LSTM for noisy data

$\lambda_{physics}$	0 (= no physics)	1	0.1	0.01	0.001
relative $\mathcal{L}_2$ error	$1.08e+0$	$7.67e-1$	$3.65e-1$	$1.06e+0$	$1.07e+0$
threshold k (3.1) $= 0.2$	0.56	0.35	0.42	0.59	0.56
threshold k (3.1) $= 0.5$	0.90	0.52	1.18	0.90	0.90

Table 4.5: Signal-to-noise ratio: 50

$\lambda_{physics}$	0 (= no physics)	1	0.1	0.01	0.001
relative $\mathcal{L}_2$ error	$5.72e-1$	$7.97e-1$	$6.32e-1$	$1.04e+0$	$7.14e-1$
threshold k (3.1) $= 0.2$	0.92	0.28	0.56	0.62	0.66
threshold k (3.1) $= 0.5$	1.03	0.38	1.01	0.97	1.01

Table 4.6: Signal-to-noise ratio:80