DL in practice
Rössler attractor

Saulnier Lucile
lucile.saulnier@gmail.com

Maazoun Imen
imen.maazoun@telecom-paris.fr

March 15, 2021

1 Chosen modelling

We followed the first approach in which we considered the full state $w = (x, y, z) \in \mathbb{R}^3$ as being known at each time step. Therefore, the goal is to train a model that predicts the following 3D coordinates as a function of the current ones $f(w_t)$. To this end, we considered :

- the discrete-time setting : the state is described by the 3D coordinates, sampled along the trajectory with a rate $1/\delta t$.
- a feed forward network with four linear layers with ReLU activations. Since we are predicting a moving 3D-coordinates point, we assumed that two consecutive states along a full trajectory are likely to be close to each other, that is to say, at each time step $w_{t+\delta t}$ is rather w_t with a slight displacement δw_t. Therefore, we adopted a ResNet-like architecture, by adding a skip connection between the input and the output layers, to efficiently track the evolution of the state along time.

2 Model training

Data

We considered the Rossler attractor defined by $a = b = 0.2$, $c = 5.7$ and generated our groundtruth training samples $(w_t, w_{t+\delta t})$ with the true simulator, taking into account the discrete setting $\delta t = 1e^{-3}$.

$$\begin{cases} \delta x_t/\delta t = -y_t - z_t \\ \delta y_t/\delta t = x_t + a y_t \\ \delta z_t/\delta t = b + z_t(x_t - c) \end{cases}$$

For our training dataset we have chosen to take a trajectory of 20,000,000 points sampled every $1e^{-3}$ second and starting from point $(-5.75, -1.6, 0.02)$. We also created a validation dataset composed of the positions of a trajectory of 200,000 points sampled every $1e^{-3}$ second and starting from point $(0.9, -0.6, 0.028)$ and a test dataset composed of the positions of a trajectory of 20,000,000 points sampled every $1e^{-3}$ second and starting from point $(0.01, 2.5, 3.07)$.
Finally, we calculated the mean and variance over the training set in order to normalise the data for the training phase.

Training protocol

We adopted a two phases training protocol. In both phases, the loss function that our training seeks to minimise is the addition of 2 terms:

- The first term seeks to minimise the difference between the position at t + 1, $\hat{w}_{t+1}$, predicted from the position at t, w_t, and the true position at t + 1, w_{t+1}.
- The second term is added to help the training by providing a minimisation objective on a local trajectory. This second term seeks to minimise the difference between the next T positions, $\hat{w}_{t+1:t+T}$, predicted from position t, w_t, with the true $w_{t+1:t+T}$ sequence. The motive behind using this latter is to control the alignment between the predictions and the ground truth trajectory locally. This is a way of emphasising the temporality of our model, which must sample every $1e^{-3}$ seconds.

First, the network is trained to minimize the objective L_1 combining the L1-loss over the immediate predictions and the normalized Soft Dynamic Time Warping loss [1] - SDTW$_{norm}$ - over the induced trajectory in the near future, along a fixed time window.

$$L_1 = \|\hat{w}_{t+1} - w_{t+1}\|_1 + \text{SDTW}_{norm}(\hat{w}_{t+1:t+T}, w_{t+1:t+T})$$
$$\text{where} \quad \text{SDTW}_{norm} : (x,y) \mapsto \text{SDTW}(x,y) - 1/2(\text{SDTW}(x,x) + \text{SDTW}(y,y))$$

Then, the network is retrained to minimize a second objective L_2 combining the L1-loss over the immediate predictions as well as the induced trajectory in the near future, along a fixed time window.

$$L_2 = \|\hat{w}_{t+1} - w_{t+1}\|_1 + \|\hat{w}_{t+1:t+T} - w_{t+1:t+T}\|_1$$

We choose a training in two phases because it is easier to get the trajectories, $\hat{w}_{t+1:t+T}$ and $w_{t+1:t+T}$, to come together at the beginning with a soft-DTW loss than with an L1-loss. However, to encourage good alignment, we found that it was more effective to do a second training session with an L1-loss rather than staying with the soft dtw.

3 Results

3.1 Dynamics

We estimated the lyaponuv exponents on 20,000,000 points trajectories from the train and test set initialization point. Given that we had to consider a finite discrete trajectory, we prefered to fairly compare our model results with the ground truth values that it would have predicted from the true trajectory taken in the same setting (starting from the same location and sampled every $1e^{-3}$ seconds).
Our best model achieved roughly the ground truth value of the largest lyapunov exponent $\sim 7.10^{-2}$, up to a 0.002 error on the train set. The results on the test set are also satisfying with an error up to 0.001. Given that the test set was generated from a different location than the train set, we can see that the model has a quite good performance.
For the equilibrum point, which we predicted with respect to the training initial location $(-5.75, -1.6, 0.02)$, there is a gap between the ground truth and the prediction, but regarding the wide range over which the 3d coordinates vary, results are also satisfying.

lyapunov exponent	true	predicted	error
train	$\begin{bmatrix} 0.070910 \\ -0.0003735 \\ -5.383140 \end{bmatrix}$	$\begin{bmatrix} 0.068845 \\ 0.000494 \\ -2.642010 \end{bmatrix}$	$\begin{bmatrix} 0.002064 \\ 0.000867 \\ 2.741129 \end{bmatrix}$
Test	$\begin{bmatrix} 0.063285 \\ 0.003576 \\ -5.393511 \end{bmatrix}$	$\begin{bmatrix} 0.061553 \\ 0.000402 \\ -2.652050 \end{bmatrix}$	$\begin{bmatrix} 0.001731 \\ 0.003173 \\ 2.741461 \end{bmatrix}$

	true	predicted	error
equilibrum point	$\begin{bmatrix} 0.007026 \\ -0.035131 \\ 0.035131 \end{bmatrix}$	$\begin{bmatrix} 0.001938 \\ 0.008005 \\ -0.002220 \end{bmatrix}$	$\begin{bmatrix} 0.005087 \\ 0.043136 \\ 0.037351 \end{bmatrix}$

3.2 Statistics

To better evaluate the results of our model we compared the statistics of the predicted trajectory from a test point (0.01, 2.5, 3.07) with the true trajectory from this same point.

Histograms

To better evaluate the performance of the model on the long term, we visualize a test 3D trajectory starting at a new location (0.01, 2.5, 3.07) as well as the distribution of the predictions coordinates as a function of x, y and z as shown in Figure 1. Results are compared with the ground truth. We notice that the model covers roughly the same locations as the true dynamic system with the same probabiltiy. For clarity, the percentage of intersection between the histograms is specified at the top of each plot in Figure 1. While this might be promising, there is no garantee that the predicted trajectory is well aligned with the ground truth on the long term. To this end, we plot the 1D trajectories at the beginning and the end of the simulated trajectory, Figure 2. Results show a perfect alignment at the beginning but a slight shift at the end of the trajectory. The interesting point is that the predicted trajectory still preserve the overall shape of the ground truth and didn't explode in one direction or another. But, with a longer trajectory or a one starting from a further location, we cannot anticipate the cumulating errors made by the model over the time.

Auto-correlations

We also visualize the time auto-correlation over a fixed time window of 100 second, at the beginning of the trajectory. We find the same periodicities between the true trajectory and the predicted trajectory, i.e. pseudo circles around the equilibrum point in the xy plane (at a high rate) and a slower periodicity in the z axis. However, this does not guarantee that there is not a time lag between the true and the predicted trajectory that sets in later.

FFT

As the system under study is not exactly periodic, it is interesting to look at the Fourier transform of the latter. As for the auto-correlations, the fast fourrier transforms are computed on the first 100 seconds of the trajectories and reported in Figure 3. We notice a good correspondence between the FFTs of the predicted trajectory and the real one.

References

[1] Marco Cuturi and Mathieu Blondel. *Soft-DTW: a Differentiable Loss Function for Time-Series*. 2018. arXiv: `1703.01541 [stat.ML]`.

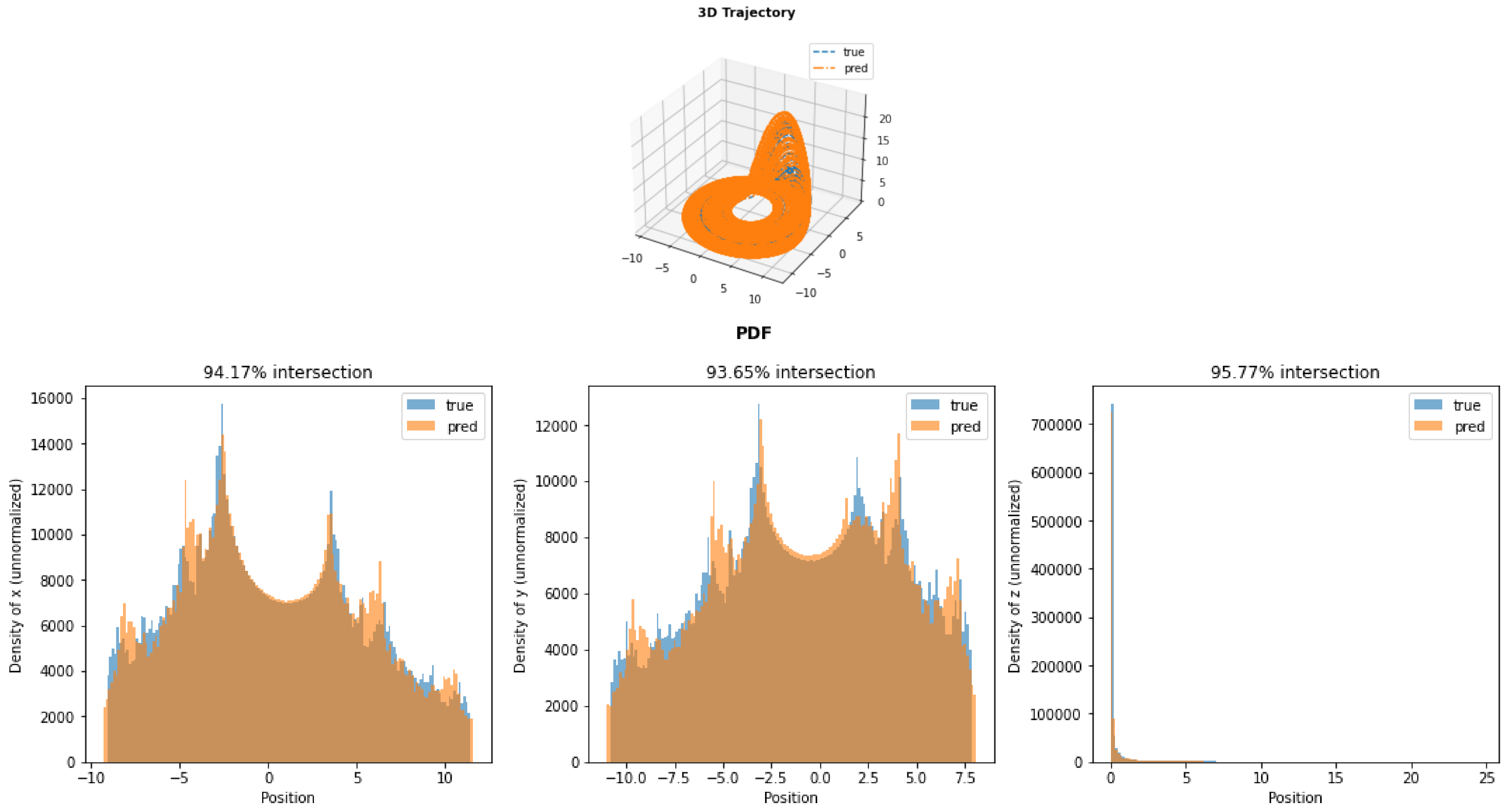

Figure 1: Test trajectory starting at location (0.01, 2.5, 3.07). Top: 3D representation of the entire trajectories (predicted and true) of 20,000,000 points. Bottom: distribution of positions along the 3 axes for the predicted and true trajectories.

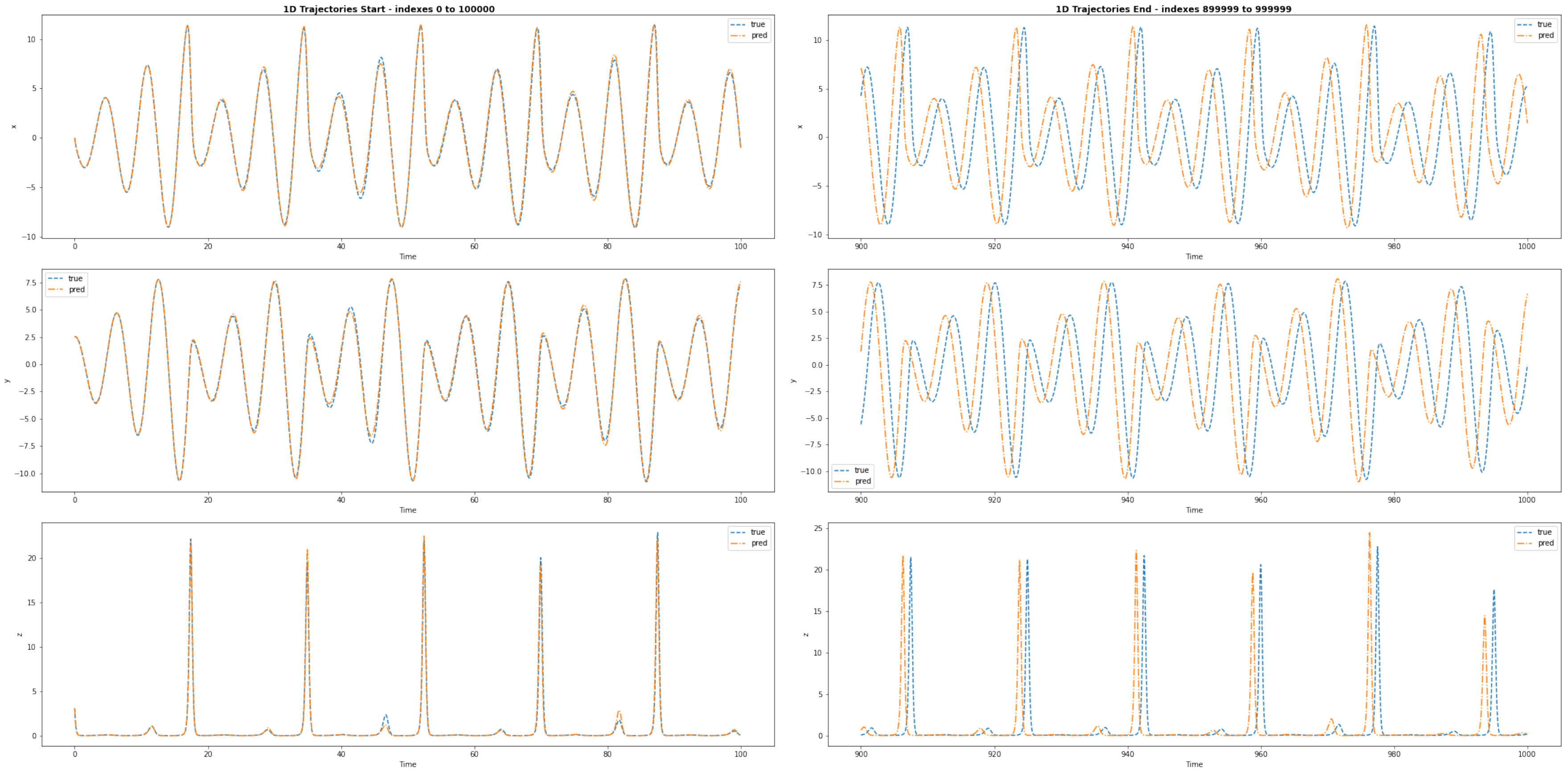

Figure 2: Test trajectory as a function of x, y and z. Left : at the beginning of the trajectory. Right: at the end of the trajectory

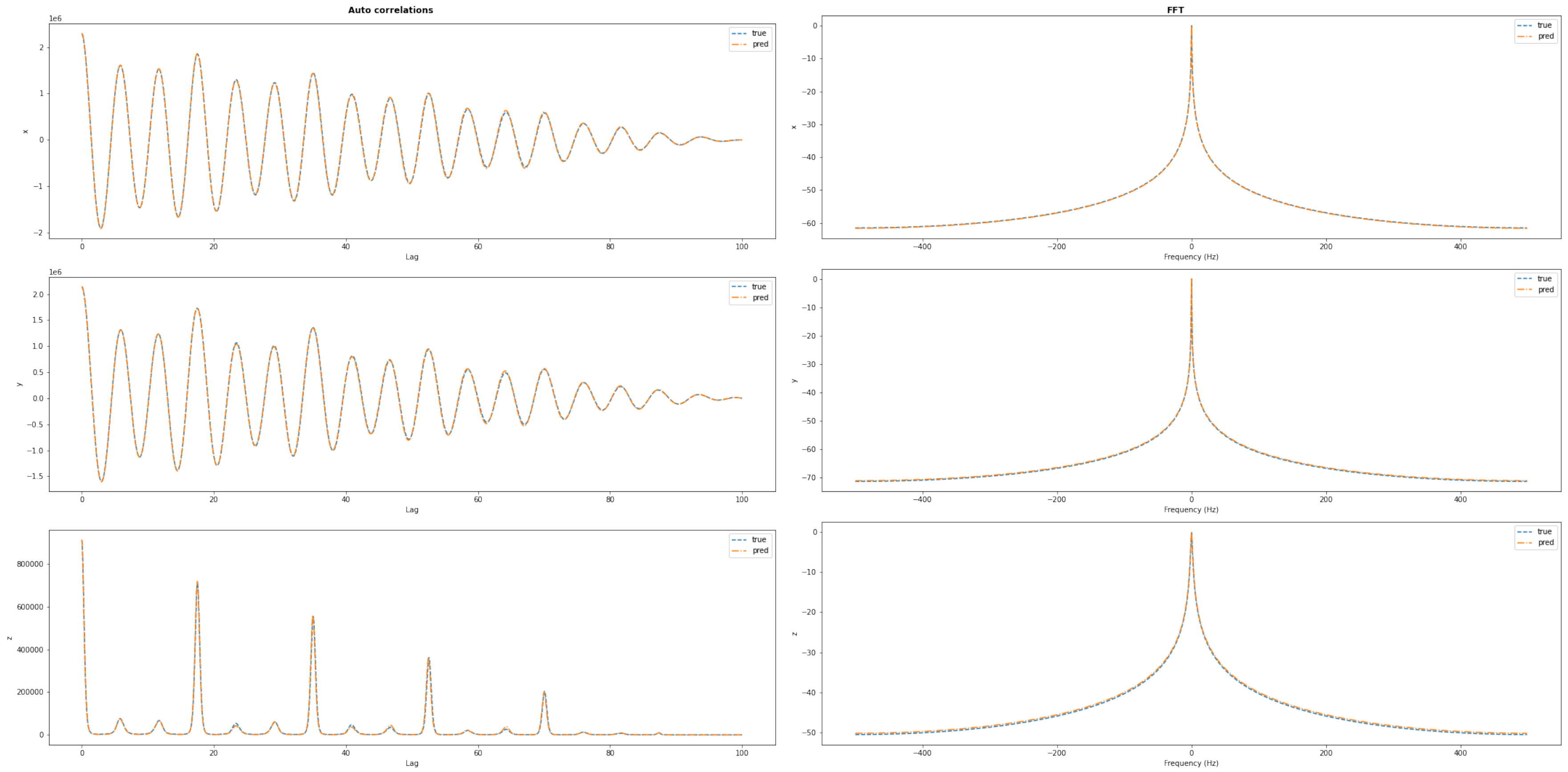

Figure 3: Time correlations (Left) and FFT (Right) at the beginning of the Test trajectory