Sequential Modeling - Ex 1

Dan Ben Ami, id: 316333079
Nir Mualem, id: 205467780

November 2022

1 Introduction and Data Generation

In this work, we consider the well-known Harmonic oscillator system. The governing linear differential equation of the Harmonic oscillator is given by

$$\frac{d^2x}{dt^2} = -\omega^2 x. \tag{1}$$

Firstly, we generated 500 starting points (positions in time $t = 0$) $x_0^i, 0 \leq i \leq 500$, each sampled randomly in uniform distribution from $[0, 1]$. Secondly, for each one of the initial points, we generated the rest of the position points in time $t = 1, 2, 3, .., 20$ by using the Verlet method which is given by:

$$x_1 = x_0 + v_0 t\Delta t - \frac{1}{2}x_0\Delta t^2 \tag{2}$$

$$x_{t+1} = 2x_t - x_{t-1} - x_t\Delta t^2 \tag{3}$$

According to the instructions, we set each initial velocity to be 0, i.e. $v_0^i = 0 \forall i \in [500]$. Subsequently, we divided the 500 generated series into a train set and test set, where the train set contains 400 series (each series is 20 points) and the test set contains the rest. The Δt we chose for the samples is 0.32.

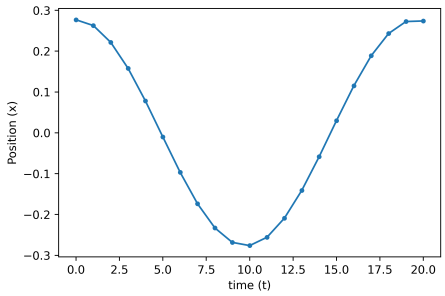

Figure 1: The graph shows position as a function of time, of one sample series from the generated data.

Next, we calculated the velocity as a function of the position of each of the series in the generated data. We performed the calculation in two different ways:

1. First-order Taylor approximation: the velocity at the starting position point of each series is given as $v_0 = 0$, then each of the following points was approximated by $v_t = \frac{x_t - x_{t-1}}{\Delta t}$. Figure 2 presented the phase portrait of the generated series. As seen, since we used an approximation of the velocity and not the actual expression, each one of the phase portraits is oval-shaped, instead of a symmetrical circle.

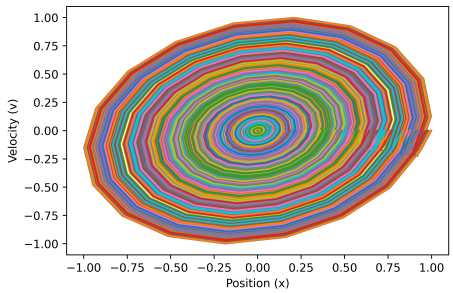

Figure 2: Phase portrait, i.e. velocity as a function of position, of 500 series in the generated data. The velocity calculated by first-order Taylor approximation.

2. Analytical analysis: we solved the equation analytically, and came to the conclusion that $v_t = x_0 \sin(\Delta t \cdot t)$. We created the speed series by putting $t = 0, 1, 2, 3.., 19$ in the above expression. In figure 3 presented the phase portrait of the generated series.

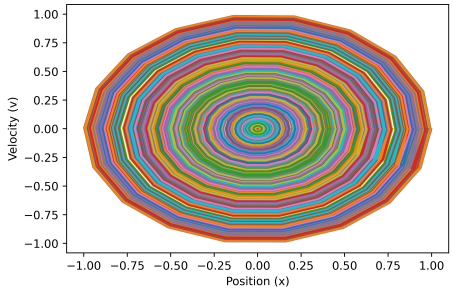

Figure 3: Phase portrait, i.e. velocity as a function of position, of 500 series in the generated data. The velocity calculated by the result of the analytical calculation.

As seen in the analytical analysis, the trajectory of each time series is determined by the starting point (and the differential equation), i.e., since we are dealing here with a harmonic oscillator without friction, and since the initial velocity of each time series is 0, the initial point actually determines what the most extreme position (that is, the largest position in absolute value) of the movement is. In

the phase portrait, this starting point is actually the radius of the circle. In addition, we will note that at the most extreme position the speed is 0, and when the position is 0, the speed is the greatest (in absolute value). In both of the phase portraits, the circles consist of straight lines, since we only took 20 positions in each time series. In order to get complete circles we had to generate time series longer than 20, and reduce Δt accordingly.

2 RNN Forecasting

Given a time series $x_{1:t} := x_1, ..., x_t$ our task is to predict the next item in the sequence using a function f that the model learned, i.e. $x_{t+1} \approx f(x_{1:t})$.
As mentioned in the last section, we generated 500 sequences of 20 items in each sequence, then, we split the generated data to train and test sets which are consisting of 80% and 20% of the generated sequences, respectively.
We designed a model that uses Recurrent Neural Network (RNN) and was trained using the standard mean squared error (MSE) loss for any index t:

$$MSE(\tilde{x}_t, x_t) = \|\tilde{x}_t - x_t\|_2^2 = \sum_j (\tilde{x}_t[j] - x_t[j])^2$$

where $x_t[j]$ is the j-th entry in the vector x_t, $\tilde{x}_t$ is the prediction.
Our model is made of two parts, an RNN part, and a linear layer.

1. The RNN part has 2 layers and a hidden state of size 100. We tried several hyper-parameters and concluded empirically that those values are achieving the best prediction of an entire sequence out of only x_0. Hidden

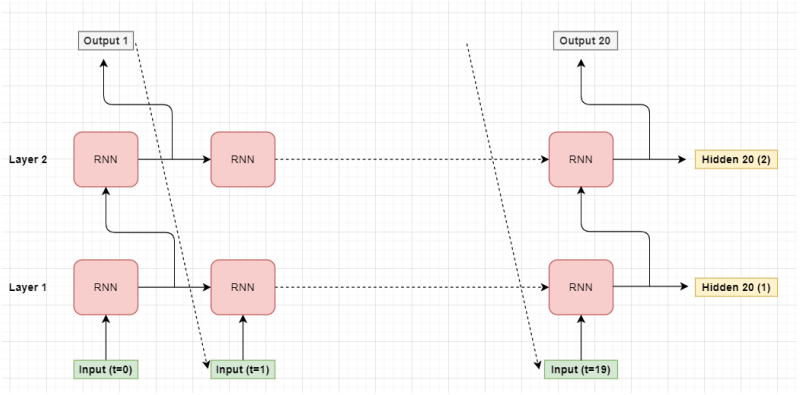

Figure 4: RNN part architecture - two layers of RNN cells, the dashed line is used only while inference the entire sequence out of x_0 (the training makes use of all $x_{0:19}$).

state h_t is computed by $h_{t+1} = \sigma(Wh_{t-1} + Ux_t)$, where W is the hidden-to-hidden matrix, U is the input-to-hidden matrix, and σ is the nonlinear function tanh.

2. A Fully-connected layer (linear layer), we need this part because the size of the RNN output (using PyTorch implementation) is the hidden state size. In regression tasks we need the output size to be 1.

Additional hyper-parameters are the learning rate (0.001) and the number of epochs (100). The batch size is one, that is one sequence per batch. The model training time is about 100 seconds on Tesla T4 GPU (google colab free GPU), while the inference time is about 0.0025 seconds. We measured the error regarding to each epoch on the train and test sets using the averaged MSE on the entire set.

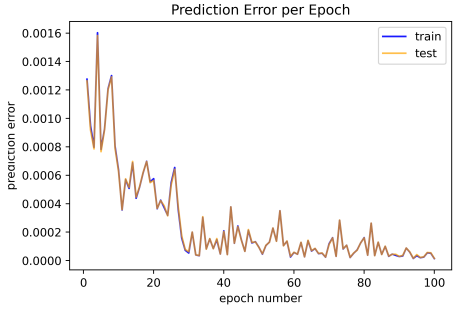

Figure 5: Train and test prediction errors (averaged MSE) as a function of epoch number, we can notice that most of the effective training happens in the first 30 epochs and train error is a bit better than the test error most of the epochs.

3 Phase Portrait of the Learned Space

In the last part of this work, we sampled another 10,000 points from the interval [0,1] using the uniform distribution. For each stating position $x_0^{(i)}, 1 \leq i \leq 10,000$, we used the trained RNN model to predict for each stating position point the next 20 position points.

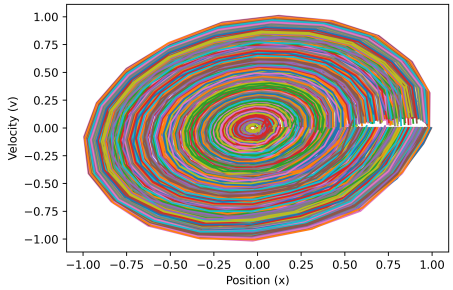

Figure 6: Phase portrait, i.e. velocity as a function of position, of 500 predicted series. The velocity calculated by first-order Taylor approximation.

As seen in figure 6, the RNN model predicts quite well the trajectory of

each starting point. We will note that there is still a difference between the model predictions and the real location which can be predicted by performing an analysis of the system and predicting the locations according to the results of the differential equation of the system. This difference can be due to several reasons such as:

- The amount of training data is insufficient (which can create a lack of accurate evaluation of the true distribution of the trajectories)
- Over training of the model, i.e. too many epochs (which can create over-fitting).
- The number of parameters and the architecture of the model (which can create under-fitting).
- Incorrect choice of hyper-parameters (which can prevent the model from converging to the values of the weights that result in the best performance).

4 Conclusions

In this assignment, we found out that the initial point determines the most extreme position and speed of the movement (the initial position is the radius of the movement circle as shown in figure 1).
Then, we built a model that forecasts the next item in a time series related to the harmonic oscillator, after the initial point x_0 is given. We showed that our model can predict reasonably, however, we believe that by use of longer sequences and some modifications in the architecture, as RNN models are known for their long memory problems, we can archive better predictions.
We summarize by showing figure 7 where we can see one successful prediction against the ground truth which is our generated data.

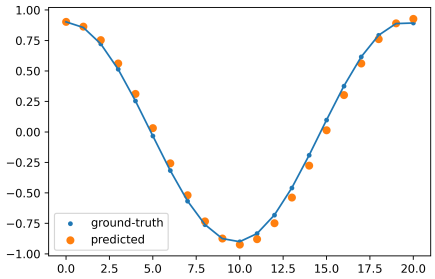

Figure 7: Prediction is reasonably against the ground truth, the prediction was made using only x_0 as an input.