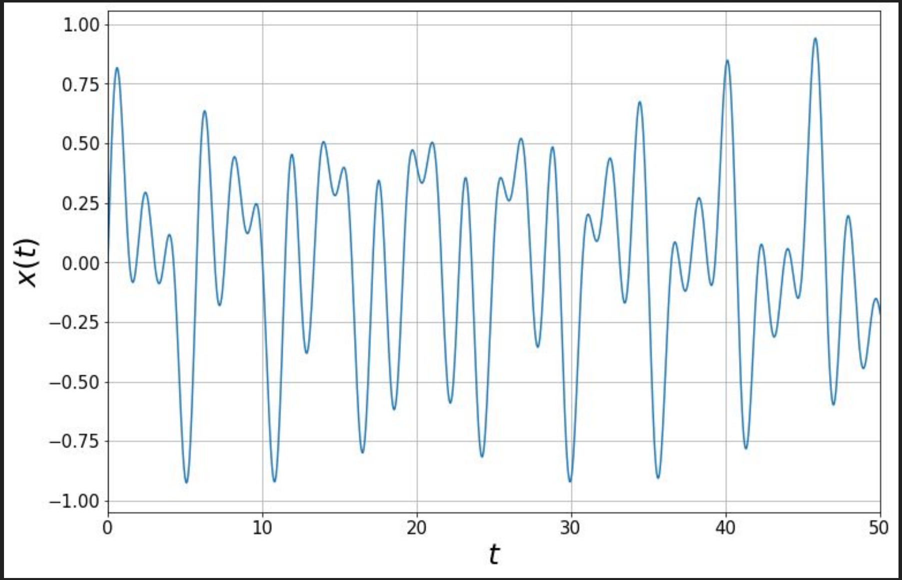
Time series prediction with Neural Networks

Luan de Souza Silva

The dataset

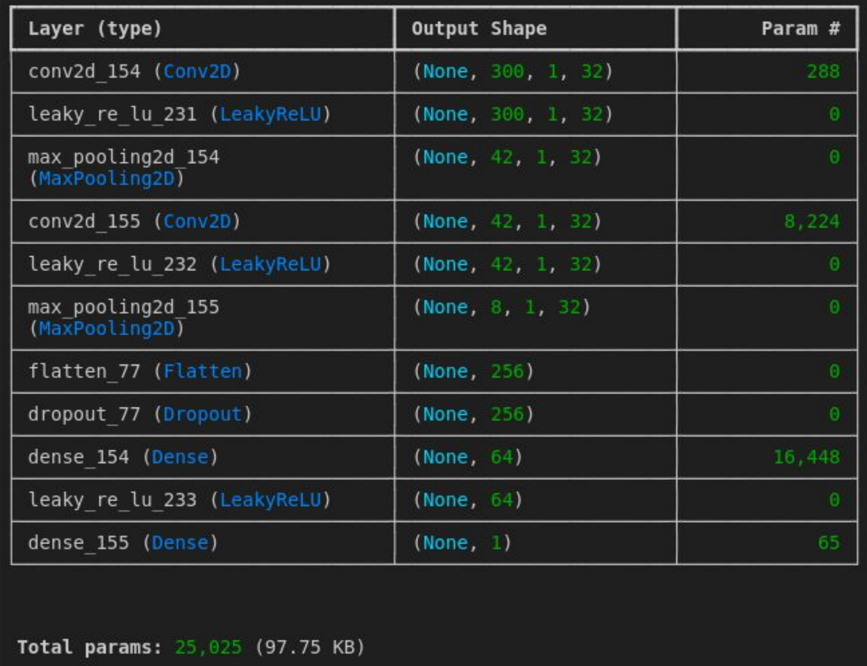
- Sets of sums of periodic functions (sines) with incommensurable frequencies.
- For instance, let the dataset be generated by the following function:

$$x(t) = \frac{1}{3}(\sin(t) + \sin(\sqrt{5}t) + \sin(\sqrt{11}t))$$



- The Model

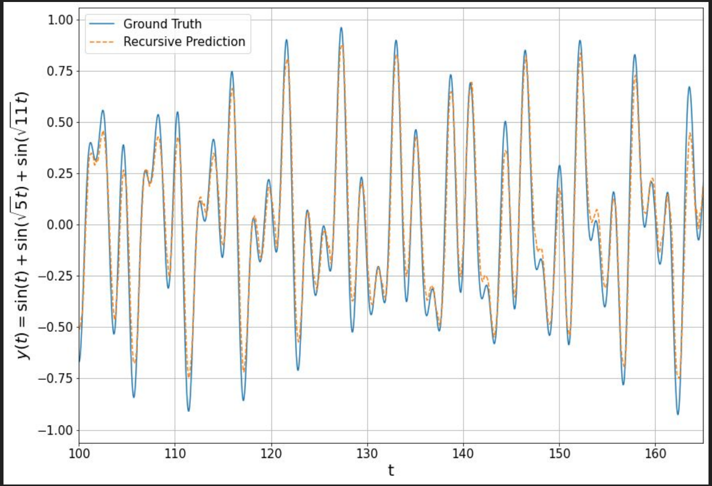
- Convolutional Neural Network (CNN)
- Two convolutional layers with N_neurons each
- One Dropout (alpha = 0.3) after the Flatten Layer
- Onde Dense Layer with 2*N_neurons
- All layers uses the Leaky ReLU as activation function
- For N_neurons = 32 there are 25,025 trainable parameters
- The Mean Squared Error (MSE) were used as Loss Function



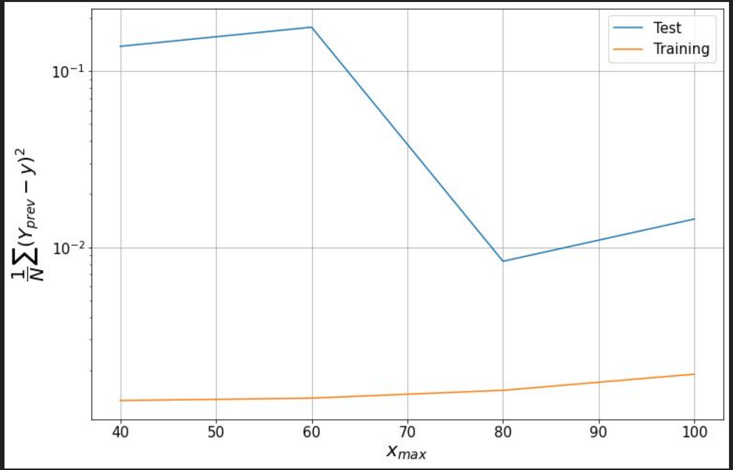
```
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
│ conv2d_154 (Conv2D)             │ (None, 300, 1, 32)     │           288 │
│ leaky_re_lu_231 (LeakyReLU)     │ (None, 300, 1, 32)     │             0 │
│ max_pooling2d_154               │ (None, 42, 1, 32)      │             0 │
│ (MaxPooling2D)                  │                        │               │
│ conv2d_155 (Conv2D)             │ (None, 42, 1, 32)      │         8,224 │
│ leaky_re_lu_232 (LeakyReLU)     │ (None, 42, 1, 32)      │             0 │
│ max_pooling2d_155               │ (None, 8, 1, 32)       │             0 │
│ (MaxPooling2D)                  │                        │               │
│ flatten_77 (Flatten)            │ (None, 256)            │             0 │
│ dropout_77 (Dropout)            │ (None, 256)            │             0 │
│ dense_154 (Dense)               │ (None, 64)             │        16,448 │
│ leaky_re_lu_233 (LeakyReLU)     │ (None, 64)             │             0 │
│ dense_155 (Dense)               │ (None, 1)              │            65 │

Total params: 25,025 (97.75 KB)
```

- Training and testing

- A range of values for the look_back parameter and t_max were used.
- The best case was t_max = 80 with look_back = 300.
- In order to test the capability of the model to extrapolate the training dataset, a recursive method was implemented to generate 1700 points beyond the end of the training dataset



- MSE was calculated for both training and test datasets.



- Conclusions and further analysis

- The training time was about 1 minute (using the GPU of my personal laptop) and the prediction time was around 1 minute. Probably the time complexity of the prediction is O(n), where n is the number of points being predicted
- It is clear that the model needed a relatively large period of time (many of the longer characteristic period) to “learn” the patterns, due to the non-trivial behavior of the function
- Maybe with more complex data, the training dataset needs to be larger.

- Trying to predict beyond...

- Same model, same dataset, but predicting 19.700 points instead of 1.700;
- The models training with low x_max fail to predict beyond the previous range:

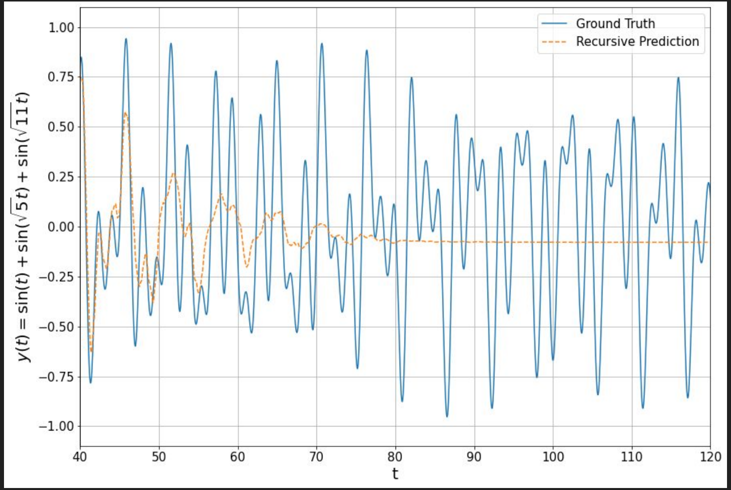


Fig 2.1: Simulation for t_max = 40.

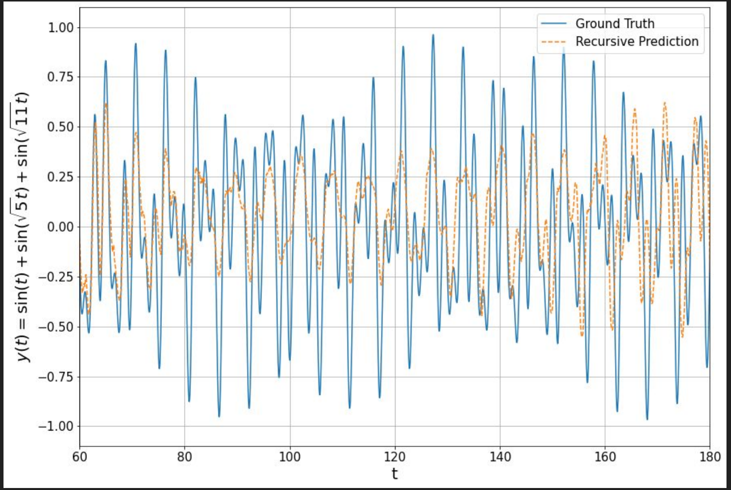


Fig 2.2: Simulation for t_max = 60.

- For t_max = 40 the prediction falls into a constant value of y(t) = 0;
- For t_max = 60 the predictions doesn’t fall to a constant value, but it doesn’t capture the frequencies and amplitudes.

- Trying to predict beyond...

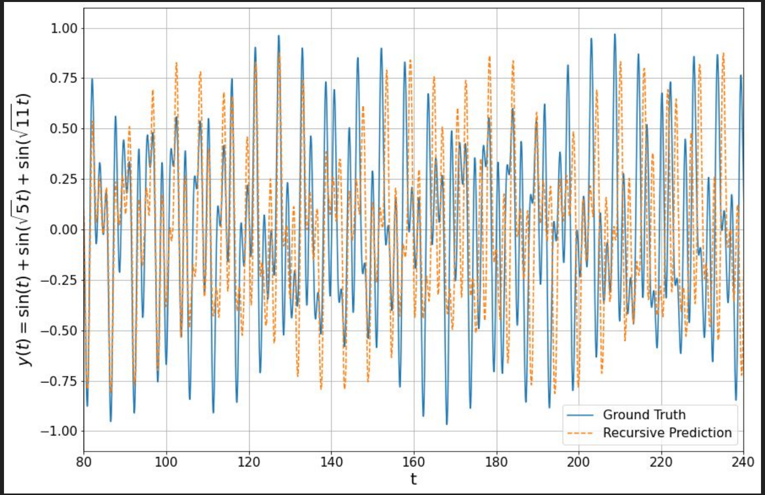


Fig 2.3: Simulation for t_max = 80.

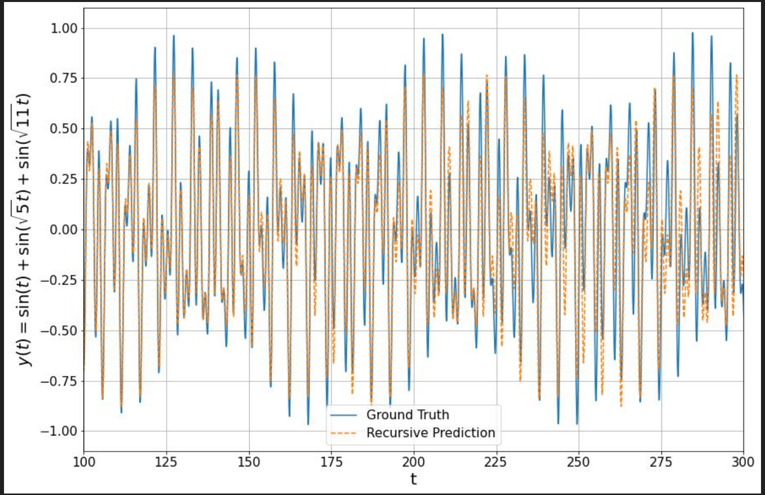


Fig 2.4: Simulation for t_max = 100.

- t_max = 100 going beyond…

- Notice that the model trained with t_max = 100 predicts very well to times up to 300, as you can see here with a larger figure

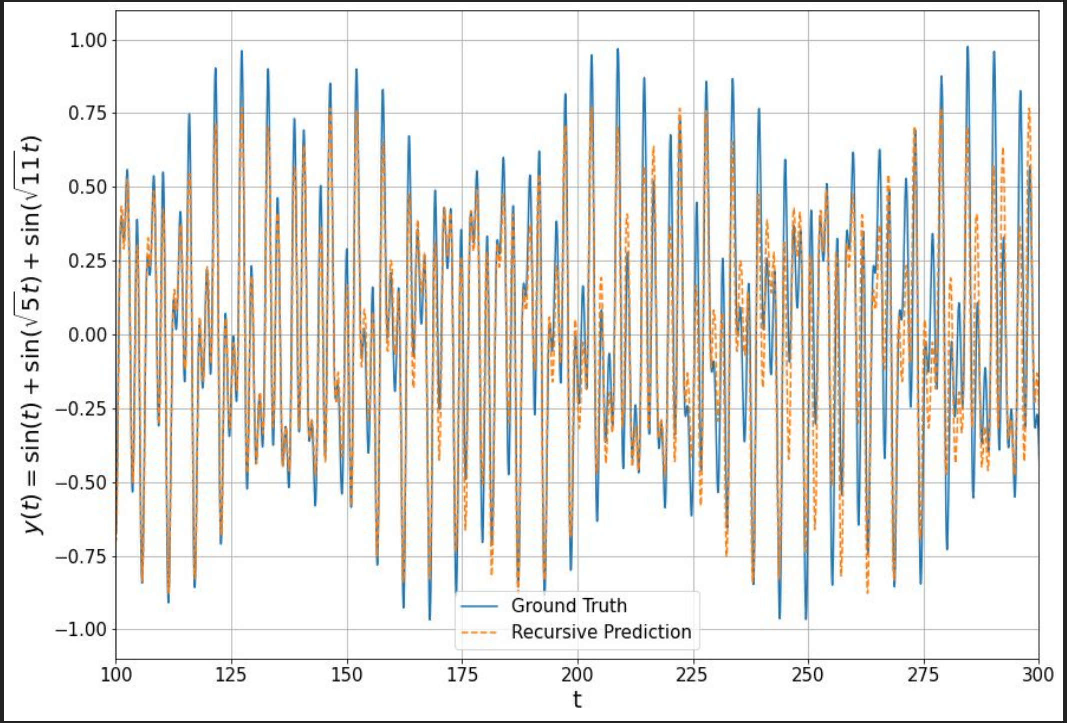


Fig 2.5: Simulation for t_max = 100. Now a little larger.

- Going further with t_max = 100

- We can go a little further, beyond to t = 300 up to t = 500, which corresponds to 7.700 predicted points.
- We see that despite the prediction is going very well yet, it is starting to “diverges” from the ground truth.

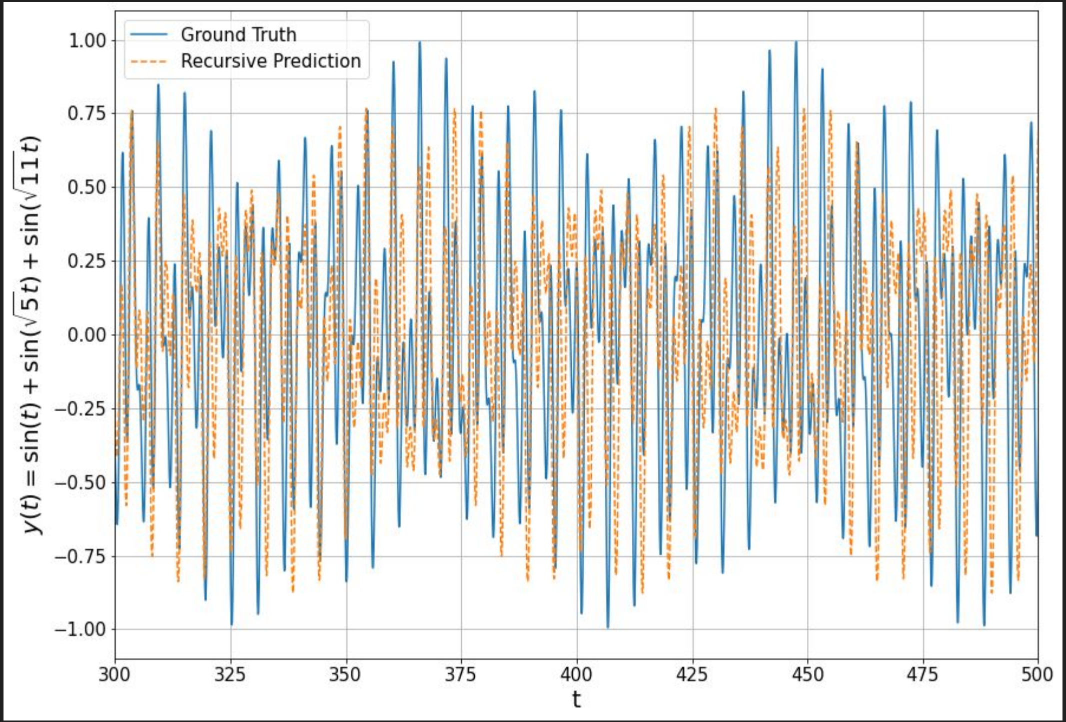


Fig 2.6: Simulation for t_max = 100. Now showing the prediction from t = 300 to t = 500.

- Going even further with t_max = 100

- Going even further, from t = 900 up to t = 1.000, which corresponds now to 19.700 predicted points.
- Now the predictions starts to fail hard. Apparently capturing the frequencies and amplitudes, but it seems like an "out of phase" prediction

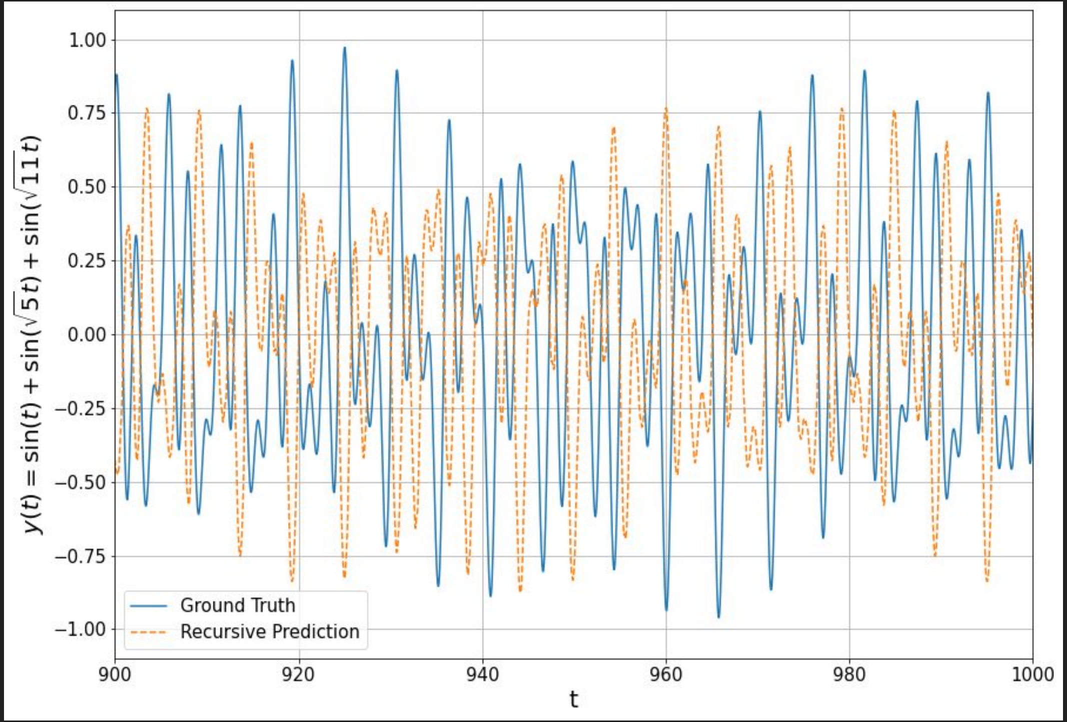


Fig 2.7: Simulation for t_max = 100. Now showing the prediction from t = 900 to t = 1000.

- Mean Squared Error Analysis

- From the data points, we can now see the quantitative performance of our models.
- Given the recursive nature of our predictions, I found out it was better to calculate the MSE of parts/“windows” of the predicted set. So I used a Roll Mean with a Window of 420 ≈ (2π √11)/0.05 points

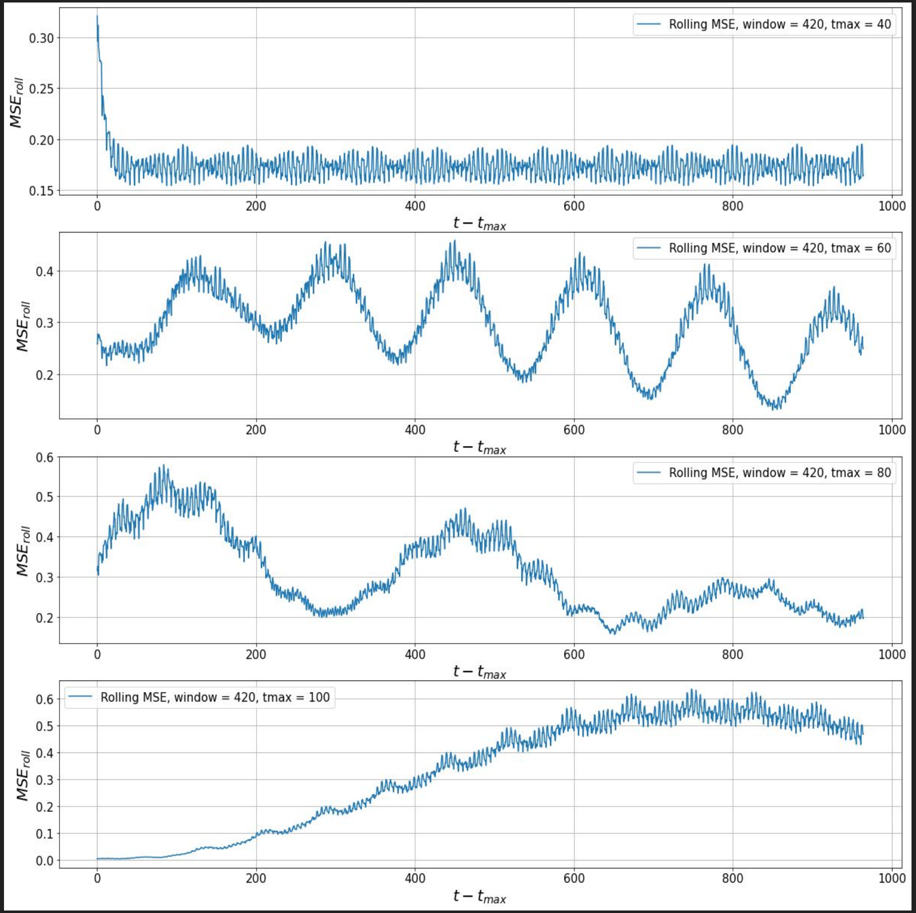
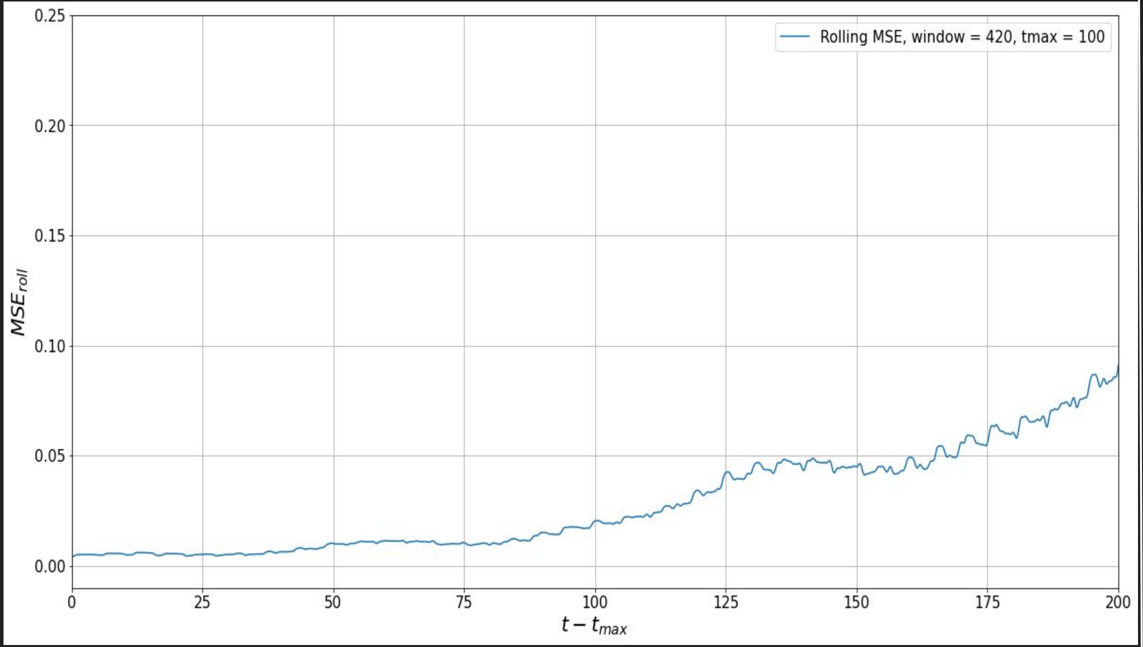


Fig 2.8: Roll Mean with the Squared Error of the predicted points in all models. The windows used in this calculations is 420, which is a number based in the higher frequency of the original data.

MSE Analysis: looking closer to the t_max = 100

- We can zoom in the first 2000 predicted points for the t_max = 100 case, where we can see that the error is very very low compared to the other cases



- Of course it is expected the error to increase in time if the prediction doesn't falls into a constant value, for example
- Notice however that there are clear tendencies in the MSE of all cases.
- Are these tendencies periodic? What are the frequencies?

Fig 2.9: Zoom in the Roll Mean of the first predicted points of the t_max = 100 model. Notice here a very low Mean Squared Error compared to the other models (previous slide).

- A little deep in the tendencies of MSE

- The MSE reminds on trigonometric functions;
- Performing a FFT on the data to find out the frequencies;
- Notice there are evident frequencies in the problem, especially for t_max = 40, 80 and 100.
- In all cases there is a peak in frequency close to ω = √11

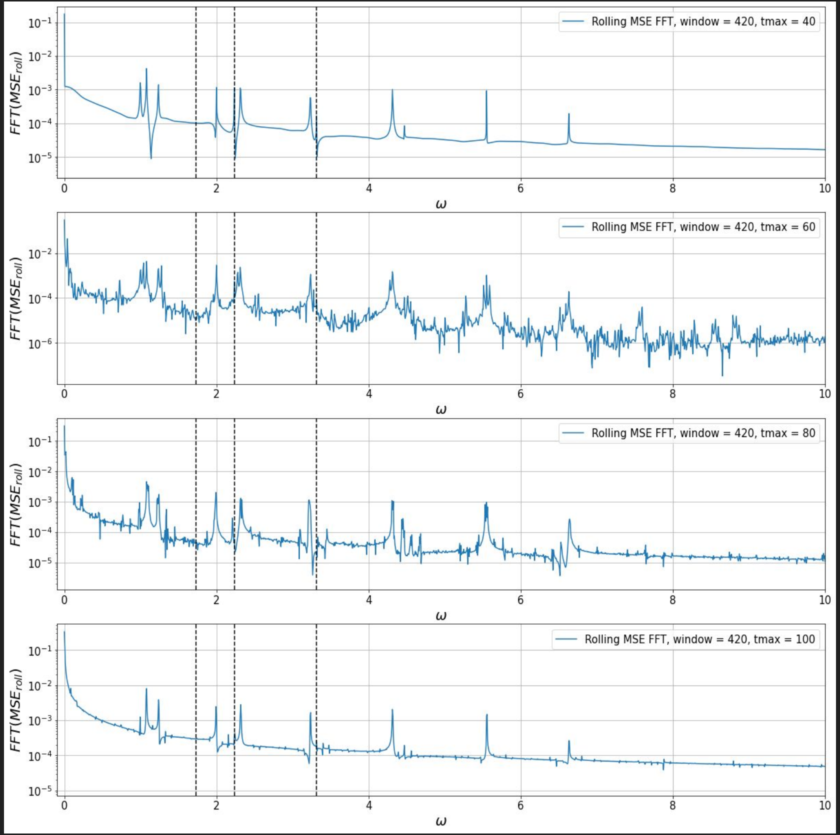


Fig 2.10: FFT spectrum of the MSE for t_max = 40, 60, 80 and 100. The dashed line corresponds to (from left to right) ω = √3, ω = √5, ω = √11, which are the frequencies of the original dataset (Ground Truth).

- Conclusions and comments

- For longer predictions, the t_max = 100 reached the better results;
- Even with the prediction diverging of the Ground Truth, it’s not exploding to infinity;
- In all cases the prediction time was about 11 minutes, in agreement with the expected previously;
- There are clear tendencies in the MSE… What does it tells us?

- Next steps:

- Maybe use Linear Prediction? I sincerely found out it is not so “trivial” to implement.
- Use some attention mechanism? (I don’t know much of this, but I know it is designed to deal with long time series).

- First try with Linear Prediction

- Future terms of a time series is given by a linear combination of the last p terms:

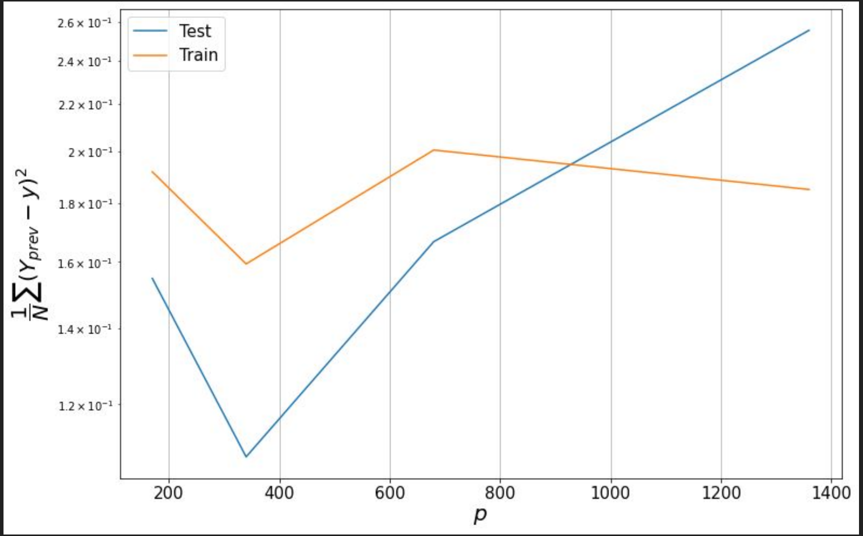
$$y_{n+1} = -\sum_{k=0}^{p-1} a_k y_{n-k}$$

- Again, the function used is

$$y(t) = \frac{1}{3}(\sin(\sqrt{3}t) + \sin(\sqrt{5}t) + \sin(\sqrt{11}t))$$

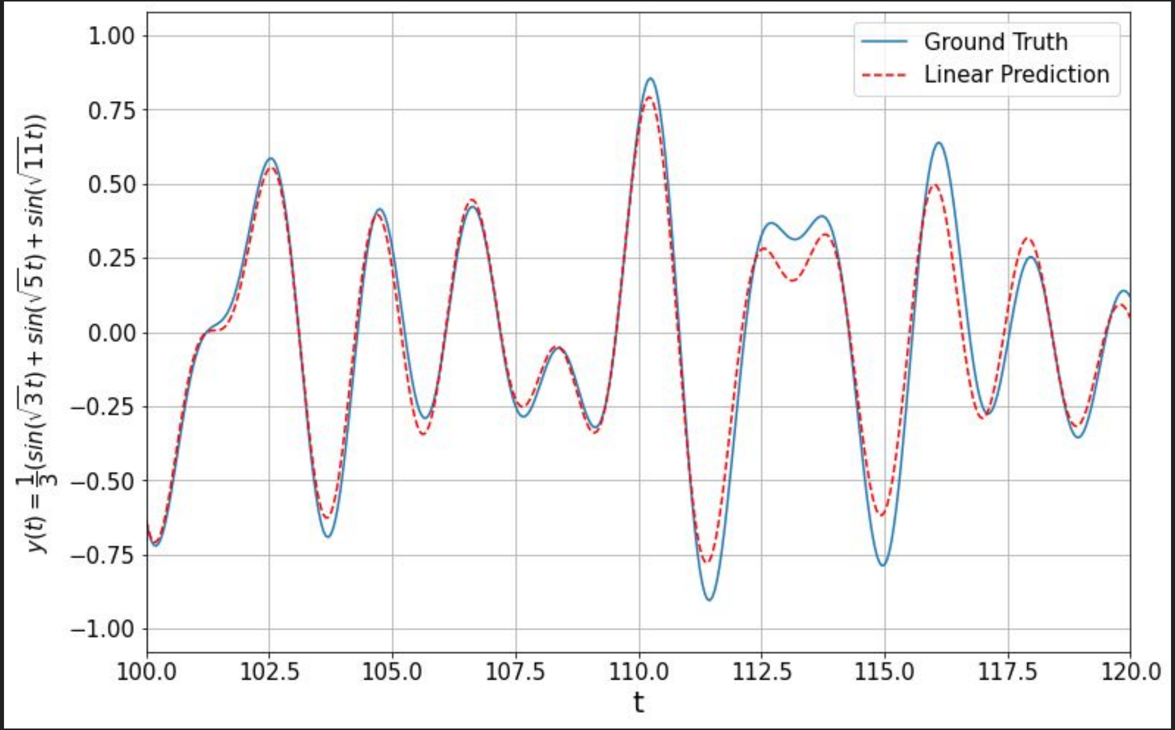
- For this simulations I used the following values for p: 170, 340, 680 and 1360.
  Notice that the minimum MSE is at p = 340 for both training and test;

- The MSE for test case was calculated for 4.000 prediction points.



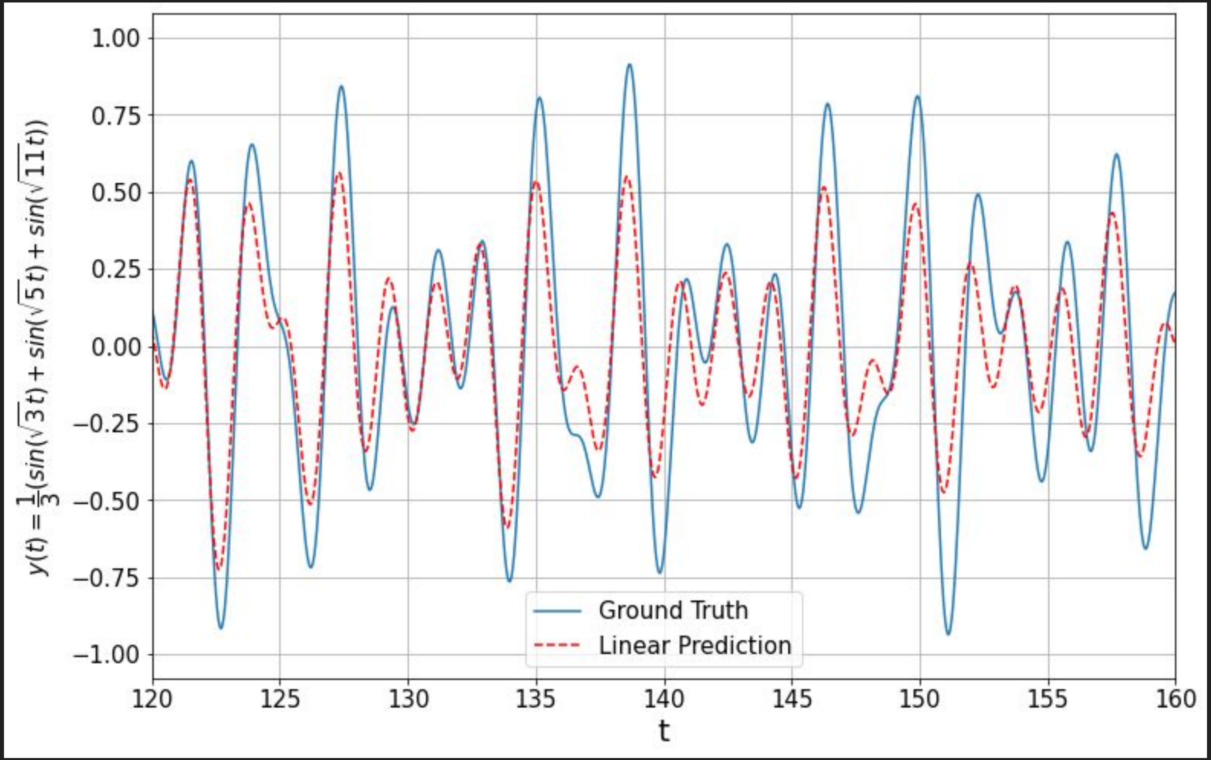
- First try with Linear Prediction

- For the first 400 prediction points (20 units of time) at p = 340:



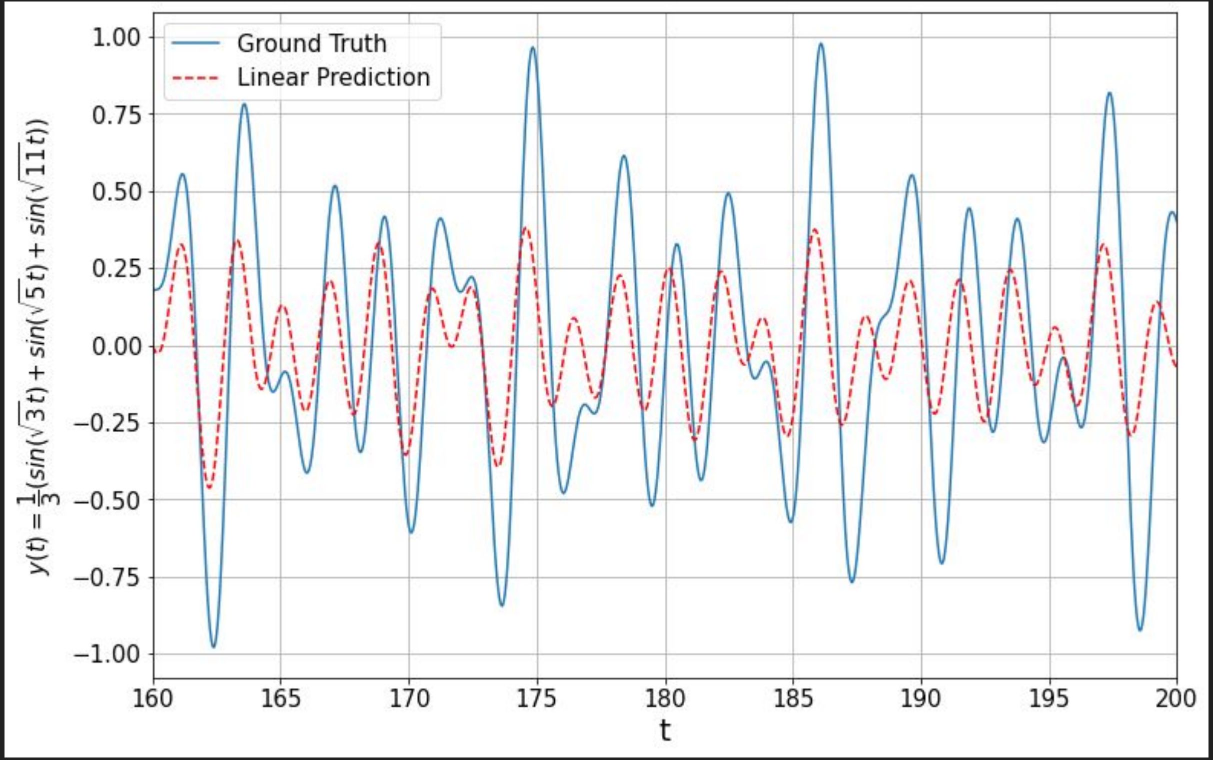
- ## First try with Linear Prediction

- From 400 to 1200 prediction points (40 units of time) at p = 340:



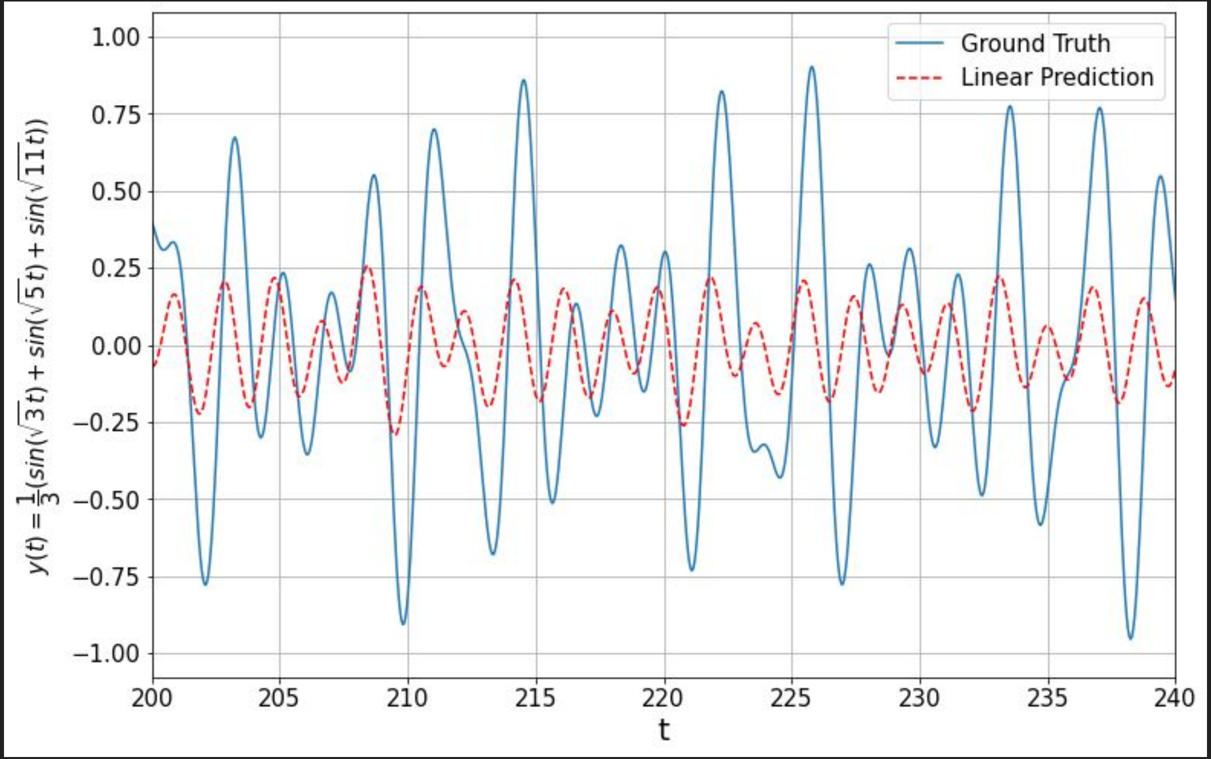
- ## First try with Linear Prediction

- From 1200 to 2000 prediction points (40 units of time) at p = 340:



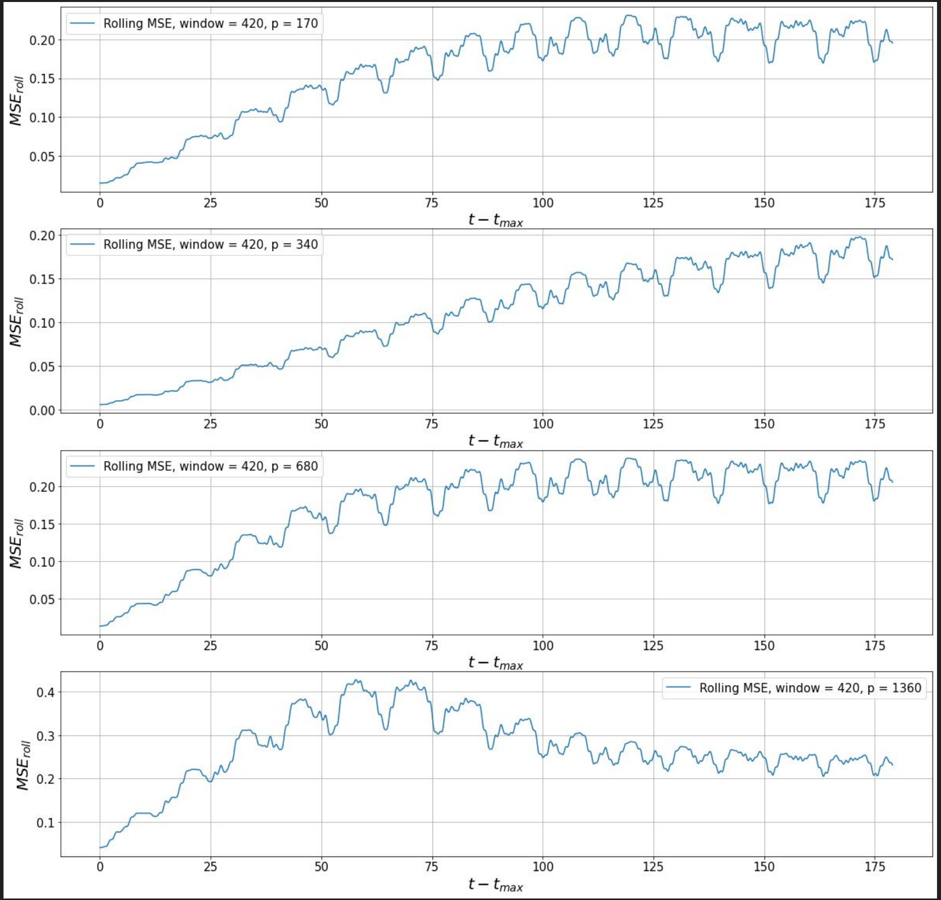
- First try with Linear Prediction

- From 2000 to 2400 prediction points (20 units of time) at p = 340:



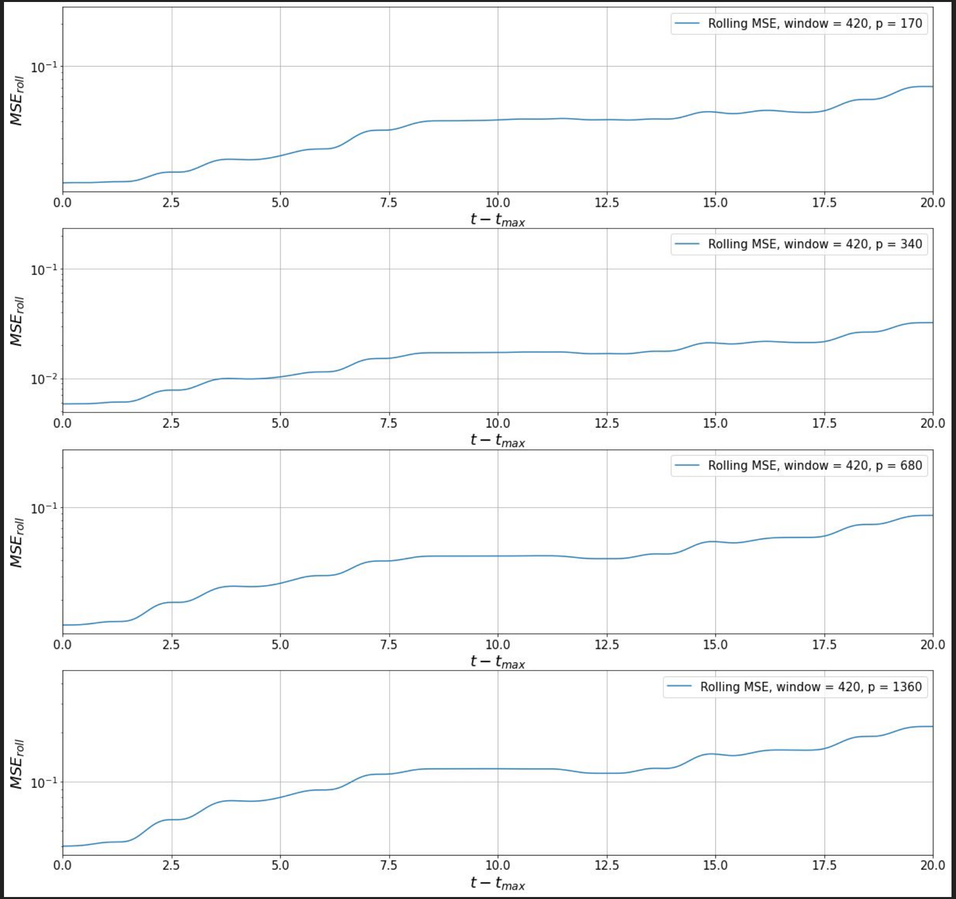
- First try with Linear Prediction

- Notice that the Linear Prediction solution begins very well but starts to decay.
- Here you can see some plots of roll mean for the predictions for all values of p used.
- The window for the rolling mean is 420.



- First try with Linear Prediction

- Looking closer to the first points of predicted data;
- The y axis is in log scale for better visualization;
- Notice that for the best case (p = 340), the mean squared error for the first 400 predicted points is around 1e-2, and for the first 100 points the MSE is even lower!



- Conclusion and comments

- The Linear Prediction is very good in predicting the first hundreds of points, but around one thousand points the solution starts to decaying and the error increases;
- The solutions for CNN is stable for longer times, but is way slower (the training and prediction for LP was around 1 second);
- Maybe explore variations of the Linear Prediction solution;