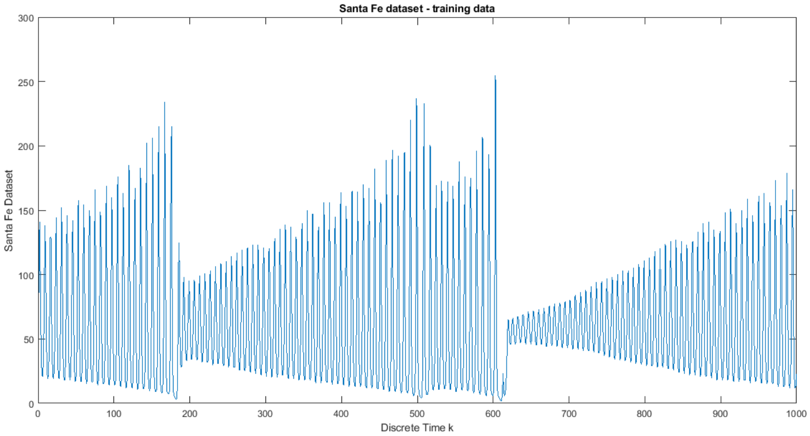
Long Short-Term Memory Network

LSTM network used here has 1-unit Input layer, 200 hidden layer neurons, and a fully connected output layer that is a regression layer used to make step-ahead on Santa Fe dataset predictions. Adam optimization algorithm is used, and threshold is set at 1.

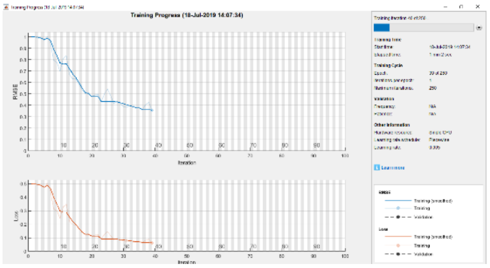
1st step is normalizing data and converting it to a Time Sequence. We split data to train the network on the first 90% of the sequence and test on last 10%.



Santa Fe Dataset

2nd step is to prepare Predictors and Responses. The output or response at each time step is the values shifted by one-time step. The predictors are the training sequences without the final time step.

3rd step is training of LSTM network for forecasting values by predicting time steps one at a time and then updating network state at each prediction.



Training iteration 40 of 250

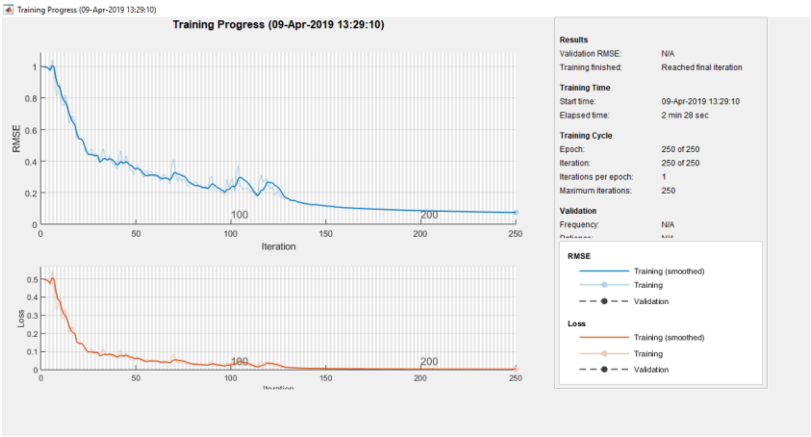
Training Time
Start time: 18-Jul-2019 14:07:34
Elapsed time: 1 min 2 sec

Training Cycle
Epoch: 39 of 250
Iterations per epoch: 1
Maximum iterations: 250

Validation
Frequency: N/A
Patience: N/A

Other Information
Hardware resource: Single CPU
Learning rate schedule: Piecewise
Learning rate: 0.005

Learn more



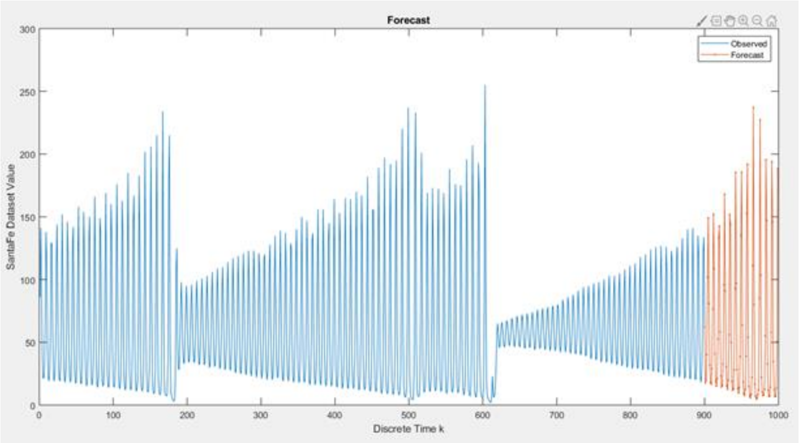
Results
Validation RMSE: N/A
Training finished: Reached final iteration

Training Time
Start time: 09-Apr-2019 13:29:10
Elapsed time: 2 min 28 sec

Training Cycle
Epoch: 250 of 250
Iteration: 250 of 250
Iterations per epoch: 1
Maximum iterations: 250

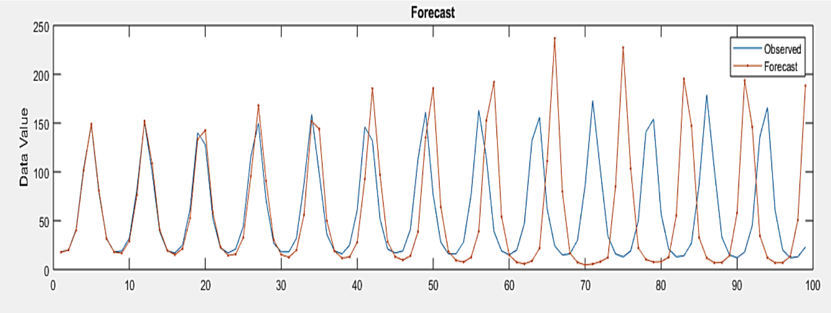
Validation
Frequency: N/A

4th step is after training preparing the network to forecast the next 100 points (test data), we *first predict on the 'Training Data'* and then use the last time step of the training response.

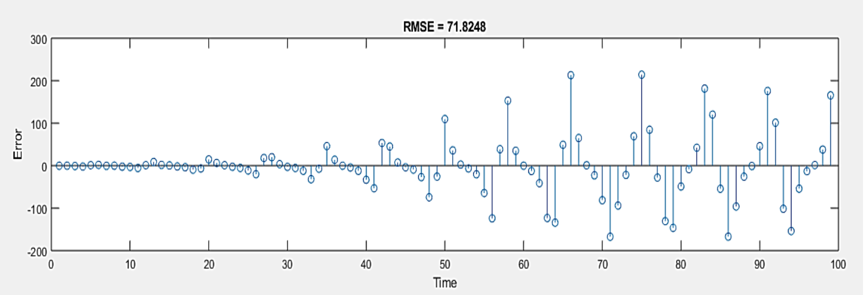


Forecasted training set datapoints 901st-1000th (red)

5th step is the comparison of the forecasted values with the test data. The first 40 forecasted values are closer to actual test data but then the accuracy of predictions drops for the next 60 values.

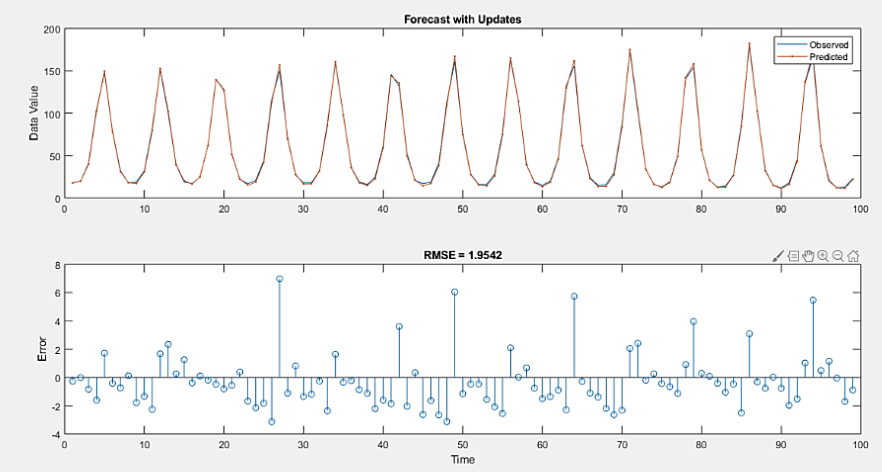


Comparison of forecasted values with test data



RMSE of predictions

Update Network State at each prediction: Due to high RMSE of the test data predictions, weights and biases of the network need to be adjusted. In LSTM we have access to the actual values of time steps between predictions to update the network state with the observed values instead of the predicted values. LSTM nodes have an internal state as a working memory so that information can be stored and retrieved over many time steps. The input value (previous output) and the internal state are all used in the node's calculations, to give output value and to update network state.



Reduced RMSE after network state updated

To prevent previous predictions from affecting the predictions on the new data, we **reset the network state**, and then initialize it by predicting on the training data. Again, the RMSE is calculated and the forecasted values are compared with the test data. We can see here that the predictions are very accurate and the RMSE has decreased from almost 72 to 2.