Performance Learning for Linear Prediction

Mikhail Podvalkov

Due to March 1, 2022

1. **Introduction**

The data for learning was an APPLE company daily closing price of a particular stock index evenly divided into three parts training, validation, and testing. The computer assignments aim to find the linear predictor parameters from the data using performance learning. The computer assignments created an autoregressive model in MATLAB code [1] and used the stochastic gradient descent method from performance learning to find the model's coefficients. Moreover, the model parameters were compere with the result of the least-squares solution. After that, experiments were provided by changing the step size and compere the final model results. Furthermore, was created a performance test on the validation, and test data.

2. **Theory**

The linear predictor model used an autoregressive model [2], presented below (1).

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + \cdots a_N y(n-N) + e(n) \; (1)$$

For the computer assignment third-order model was used presented below (2).

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + a_3 y(n-3) \; (2)$$

To find the coefficient of the model was using a stochastic gradient descent method [3] presented below

1. Initiate the 0 parameters w(0)
2. Find gradient vector $\frac{\partial J(w(k))}{\partial w(k)} = -2x(k)e(k)$
3. Update parameters w(k+1)=w(k)+μ*x(k)*e(k)
4. Go back to step 2

To compare the final result of parameters was us Wiener-Hopf solution [4] which comes from if we imagine an autoregressive model for all N of data it was:

$$\begin{gathered} y(4) = a_1 y(3) + a_2 y(2) + a_3 y(1) \\ y(5) = a_1 y(4) + a_2 y(3) + a_3 y(2) \\ \dots \\ y(N) = a_1 y(N-1) + a_2 y(N-2) + a_3 y(N-3) \end{gathered}$$

Transform this equation in matrix form:

$$\begin{bmatrix} y(4) \\ y(5) \\ \cdots \\ y(N) \end{bmatrix} = \begin{bmatrix} y(3) & y(2) & y(1) \\ y(4) & y(3) & y(2) \\ \cdots & \cdots & \cdots \\ y(N-1) & y(N-2) & y(N-3) \end{bmatrix} * \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (3)$$

The first matrix is Y, the second X, and the third is W. To find the W matrix used the matrix formula which came from equation 3

$$W = (X^T X)^{-1} X^T Y$$

3. Results and discussion

The Matlab code provides the stochastic gradient descent method presented above. The code presented the learning curve of error and for the number of samples for $\mu=10^{-4}$ and 10^{-5} (Fig 1)

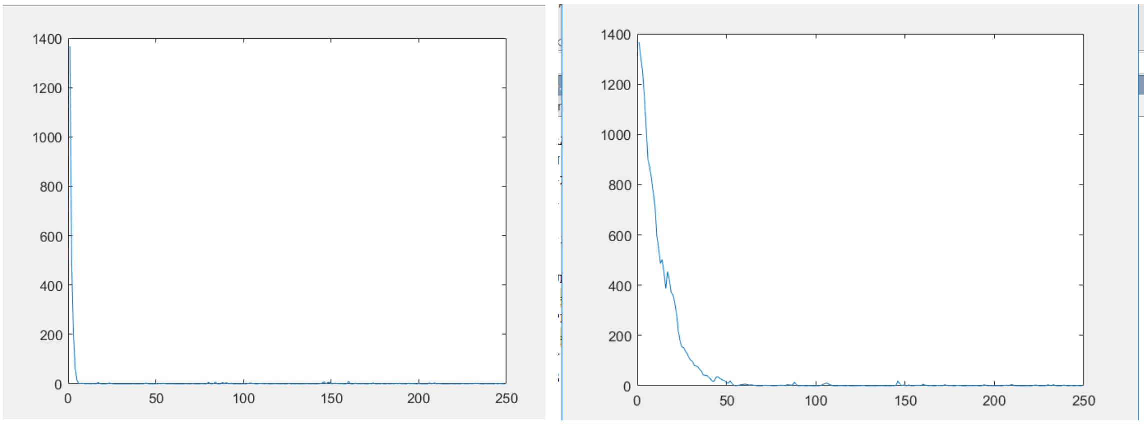

Figure 1 The learning curve of the algorithm for μ=10^-4 and 10^-5

The parameters obtained from the algorithm and the Wiener-Hopf solution are presented in Table 1.

	$\mu=10^{-4}$	$\mu=10^{-5}$	Wiener-Hopf
a_1	0.34	0.34	0.86
a_3	0.33	0.33	0.026
a_2	0.34	0.33	0.11
time	0.7	1.44	3.36e-4

The model's coefficient is comparable to the Wiener-Hopf Solution but not the same because samples are not infinite. The main difference is the speed of the algorithm comper the μ that more accuracy the algorithm then more time needs to find the solution.

The MSE on the validation dataset equals 10.13, which is big enough, and the distribution error is presented in figure 2.

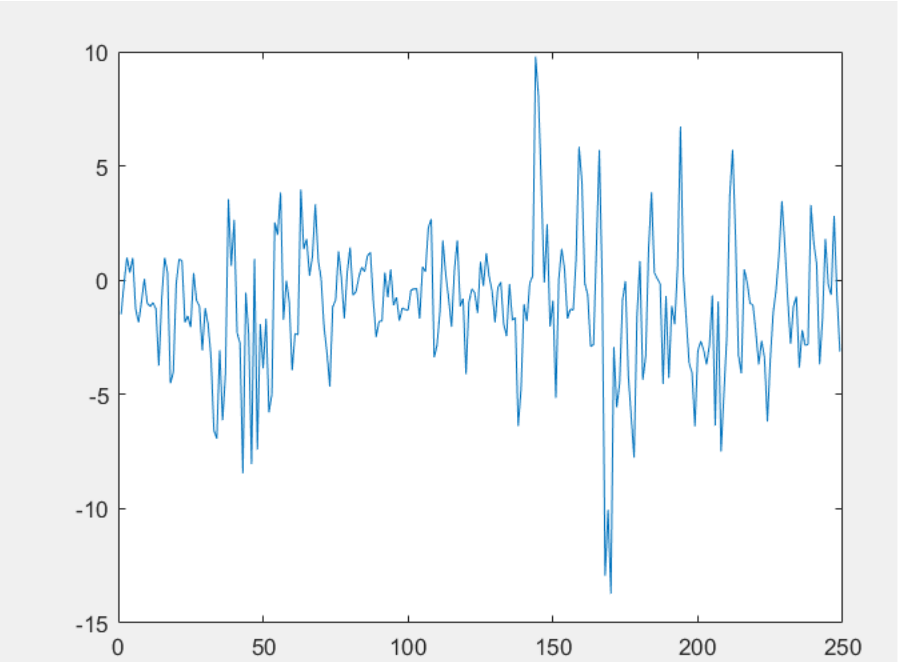

Figure 2 The distribution error on the validation dataset.

The distribution error is not the Gaussian random process, and MSE is too big that shows that the predictor is not enough to get qulity result because the 3rd oder autoregresive model does not suit all the complexity of the data of the stock market.

The MSE of the test dataset equals 12, which is worse than on the validation set. That means the predictor is not enough to create a high performance on this dataset.

4. Conclusion.

The computer assignments show how to create and train the autoregressive model on the real dataset by comparing the performance learning with the least-square solution. The way to improve the assignments is to use different methods and compare the results.

Reference

1. The official MATLAB site: https://www.mathworks.com/
2. The text computer assignments.
3. The professor M.R Azimi lecture 5-6 page 2-3
4. The professor M.R Azimi lecture 3-4 page 11