MECH&AE 263F Homework 3 Deliverables

Trevor K. Oshiro

***Abstract*—This document contains the deliverables as required by the third homework of the MECH&AE 263F course at UCLA. These deliverables show results from the implementation of a gradient descent algorithm to fit data.**

I. DELIVERABLE I

For this deliverable, application of gradient descent for a linear fitting of data was to be used to obtain optimized parameters. These parameters are for the true function to be fitted to through the gradient descent algorithm for a sample size of 10 within 0 to 5:

- $n_{true} = 0.06$
- $a_{true} = 0.25$
- $m_{true} = 0.57$
- $b_{true} = 0.11$

Gaussian noise was then added to the function values calculated from the generated x values. The data was then fit to a linear equation:

$$y = m \cdot x + b$$

This was then applied to the Mean Squared Error loss function shown below:

$$Loss(m,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - (m \cdot x_i + b))^2$$

The gradients of this function were then used to implement the gradient descent program.

Comparison of data with predicted values

From the gradient descent method applied for the linear fitting of the data, the following plots were generated for different values of epochs and learning rates applied during the process. The figure below was generated from a total of 100 epochs and a learning rate of 0.01. The fit from gradient descent is significantly off from the given data, and the resulting loss was around 0.02. However, with an increased learning rate, the figure 2 was generated. As opposed to a loss of 0.02 from 1, the loss from the plot of figure 2 resulted to $3e-6$.

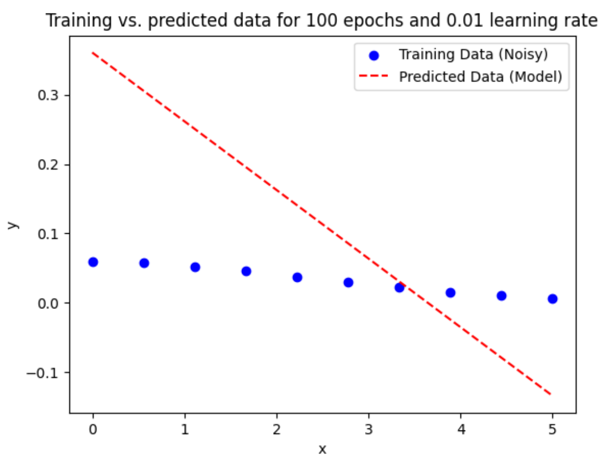

Fig. 1: Linear fit to data from 100 epochs and 0.01 learning rate

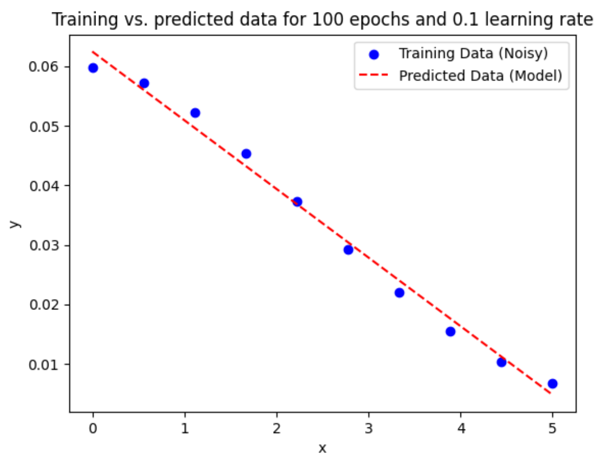

Fig. 2: Linear fit to data from 100 epochs and 0.1 learning rate

Optimizing loss with the learning rate and epochs

To further investigate the impact of the epoch number and the learning rates applied to the gradient descent method, multiple values of these values were iterated over to identify the resulting loss of the generated parameters for the linear fit. As seen in the figure below, the learning rates were applied in multiples of 10 to obtain the approximate value ranges for the optimal loss values. Similarly, the decrease in loss over epochs can be seen with the flattening of the graph with increased epochs used in gradient descent. Figure 4 below also indicates that around 0.1 for the learning rate is likely the optimal to minimize loss, as there are minimal variations in loss with further changes in the learning rate and the number of epochs used. Further limiting the range of the epochs used within the gradient descent problem then gives the figure below:

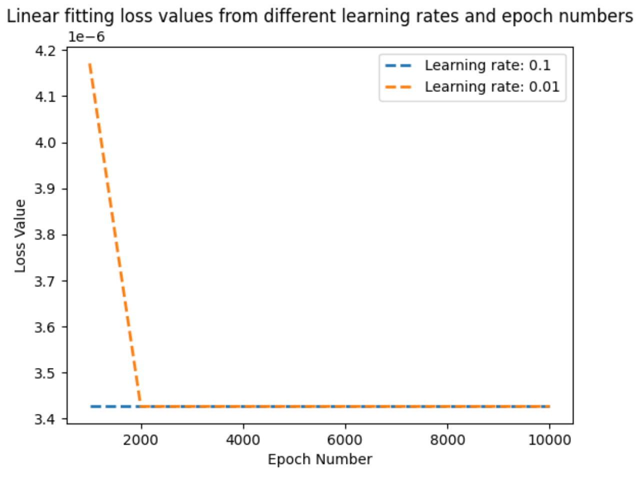

Fig. 3: Loss plotted versus different values of epochs and learning rates for a linear fit

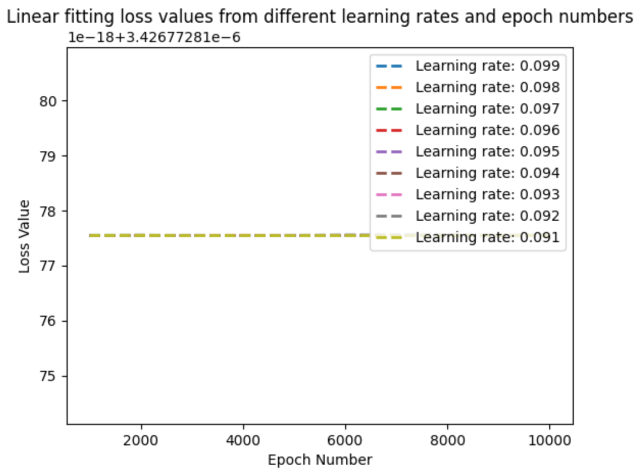

Fig. 4: Smaller increments of learning rate plotted with their associated loss for epochs

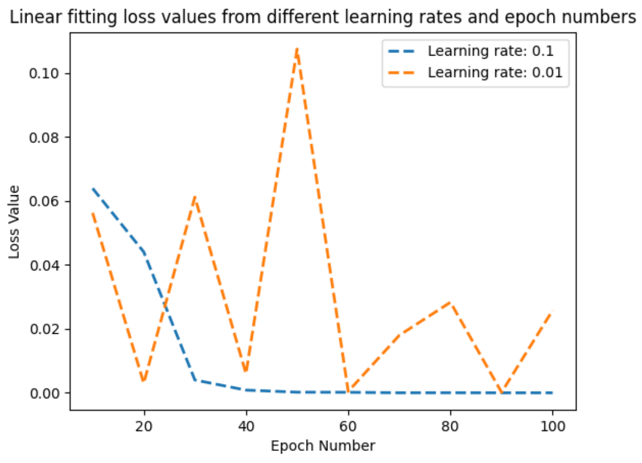

Fig. 5: Smaller increments of learning rate plotted with their associated loss for epochs

The slope down in figure 5 shows that the approximate number of epochs for optimal use of resources for the gradient descent would then fall around 100 epochs with a learning rate of 0.1. This improved fit of the parameters can be seen in figure 2 from the previous section. Overall, the impact of increasing epochs iterated over decreases loss, while increasing learning rate increases the rate the loss decreases after each epoch. However, some exceptions seen with this application of gradient descent show that there exists a number of epochs after which the loss doesn't significantly vary. Similarly, there exists an optimal range of the learning rate to utilize. Learning rates significantly higher or lower than this range make the gradient descent more prone to divergence, as can be seen with the fluctuations present in figure 2.

II. DELIVERABLE II

Similar to the first deliverable, this deliverable used the same data inputs from true fit values. These parameters are for the true function to be fitted to through the gradient descent algorithm for a sample size of 10 within 0 to 5. These data points were fit to a non-linear model. The non-linear equation used in this process is shown below:

$$y = n \cdot exp(-a \cdot (m \cdot x + b)^2)$$

This was then applied to the Mean Squared Error loss function as shown below:

$$Loss(n,a,m,b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - n \cdot exp(-a \cdot (m \cdot x_i + b)^2))^2$$

Comparison of data with predicted values

The following plots were generated for different values of epochs and learning rates applied for a non-linear fit of the data. Figure 6 was generated from a total of 500 epochs and a learning rate of 0.045. The fit from gradient descent is

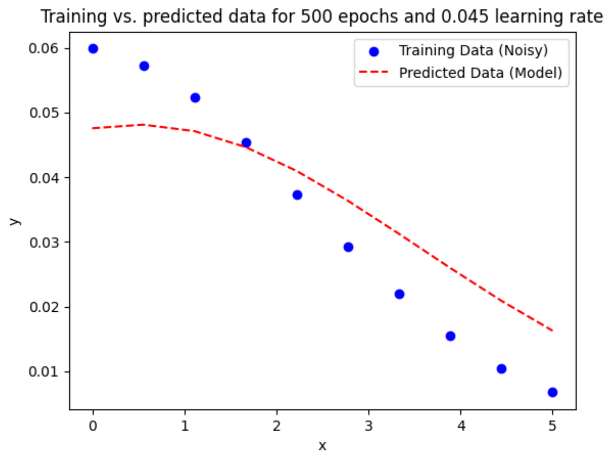

Fig. 6: Non-linear fit to data from 500 epochs and 0.045 learning rate

significantly off from the given data. The resulting loss was around $7.2e-5$. By adjusting the values of learning rate and

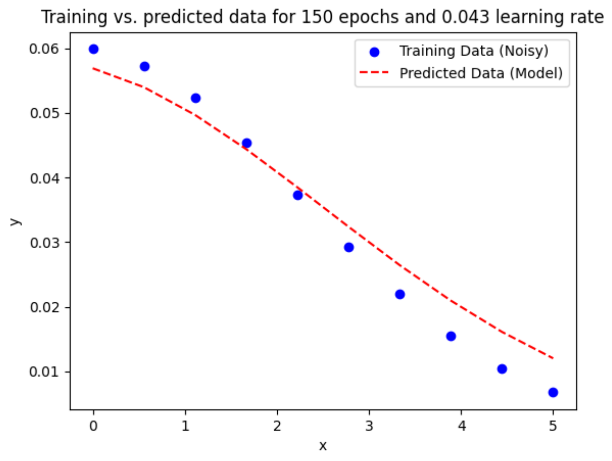

Fig. 7: Linear fit to data from 100 epochs and 0.1 learning rate

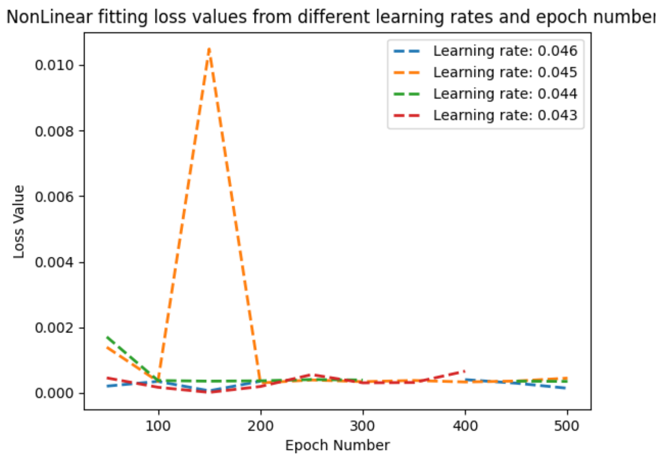

Fig. 9: Smaller increments of learning rate plotted with their associated loss for epochs

epoch number, figure 7 was generated. The loss from the plot of figure 2 resulted to $1.5e-5$. As the computed loss values were a small numbers, this possibly indicates that the parameters of the non-linear fit have a high sensitivity in relation to the resulting loss of the fit.

Optimizing loss with the learning rate and epochs

Multiple values were again iterated over to identify the resulting loss of the generated parameters for the non-linear fit. As seen in figure 8, the learning rates were applied in steps of 0.01 to obtain the approximate value ranges for the optimal loss values. Compared to the linear fit, the loss values exhibited more fluctuation between different iterations of epoch number and learning rate. The range of from figure

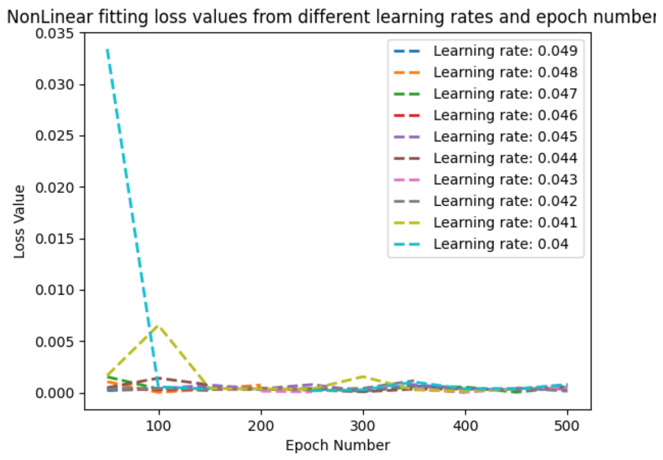

Fig. 8: Loss plotted versus different values of epochs and learning rates for a non-linear fit

8 was then plotted again for a smaller range of values to obtain the plot in figure 9. Unlike the linear fit for the data, applying a non-linear fit introduced more variations within the loss values across different numbers of epochs and learning rates. More plots can be found by running the code, or looking within the plots folder uploaded, but the approximate number of epochs found to be ideal for generating a fit resulted in around 500. Seen in figure 9, the ideal learning rate was experimentally found to be within 0.043 to 0.045. As fluctuations within lost values were seen across larger ranges of epoch numbers, the range for the implemented epoch number was kept minimal to optimize the resources used. With these considerations, the figure 7 was generated, and it had the least loss calculated from the predicted model. From tests with different epochs, a general trend can be seen in loss decreasing with increasing epoch numbers. However, as seen with figure 9, it can be seen that over a larger usage of epochs, discontinuities can result from divergence in instances of calculations of loss values. Particularly for the non-linear model, it can be seen to be sensitive to changes in learning rate values, as discontinuities or spikes in loss values were present for smaller increments of the learning rate. Thus, significant amounts of testing with selected learning rate and epoch values are likely needed to ensure convergence of the model across different datasets.

References

[1] Khalid Jawed, M, and Sangmin Lim. ”Discrete Simulation of Slender Structures”. *BruinLearn*, 2022.