Name: Gautam Vira
Student ID: 300302217
Assignment 1
Date: 09/28/2022

Estimating coefficients for polynomial regression using gradient descent (GD)

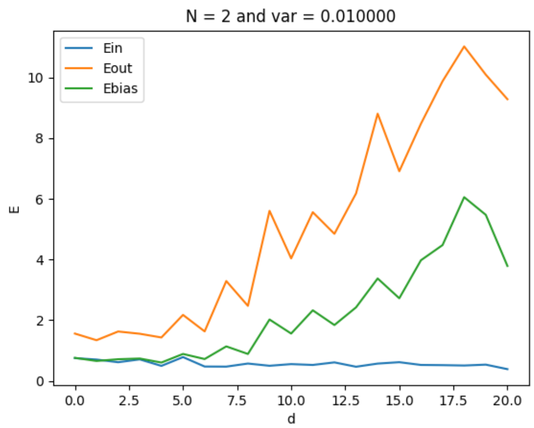
Programming Language: Python
Editor: VS Code
Libraries: NumPy, Pandas, Matplotlib, PyTorch
Execution time: Approximately 600 seconds

Procedure: A dataset is generated of N samples with two features: X and Y. A noise level is used to generate the Y, such that Y = cos(2πX) + Z, where Z is a zero mean gaussian variable with a variance, while X takes values (0, 1). The dataset is then fitted to a degree-d polynomial to estimate the coefficients of the polynomials. The coefficients are estimated using Gradient Descent. The estimated polynomials are used to find the predicted Y values for the training dataset. The Mean Square Error (MSE) is the metric used to evaluate the performance of the algorithm. The obtained coefficients are then used to predict Y values for another generated test set. The MSE of the test set serves as another metric. The gap between the $MSE_{in\ and}$ MSE_{out} can be observed to evaluate the generalization of the regression model. To experiment with the algorithm, the aforementioned process is carried out for M trials and the results (including the estimated coefficients) are averaged to test against one other test set to evaluate the bias. The outputs – MSE_{in}, MSE_{out}, and MSE_{bias} – serve as the main metrics to evaluate different configurations (sample size of the dataset, degree of the polynomial, noise level) of the model. These outputs are plotted for understanding and observing the model's performance. Furthermore, the model was evaluated with and without weight decay regularization.

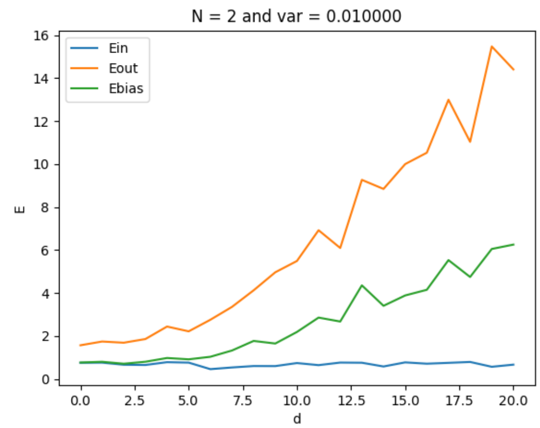
Outputs:

The graphs are plotted with the E_{in} (MSE training average), E_{out} (MSE testing avergage), and E_{bias} (MSE bias testing) along the Y-axis and the degree of the polynomial along the X-axis. Graphs are plotted for all combinations of N ∈ {2, 5, 10, 20, 50, 100, 200}, d ∈ {0, 1, 2, . . . , 20}, σ ∈ {0.01, 0.1, 1}. The Y-ranges of graphs could not be scaled to be the same for all configurations because some configurations have large errors whereas some configurations have very low errors, thus scaling them to have equal ranges would've have concealed the changes in graphs with low values or high peak errors would not have fitted in the graph.

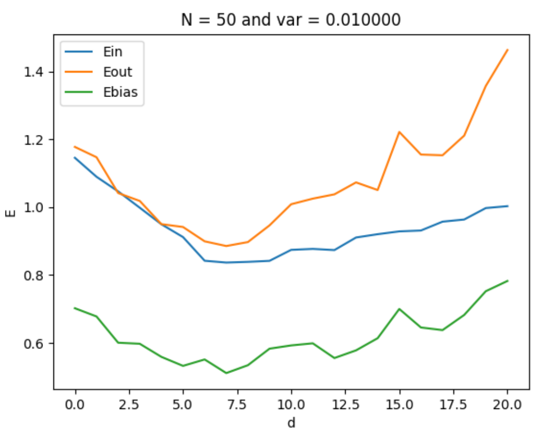
Example graphs from the output are below:



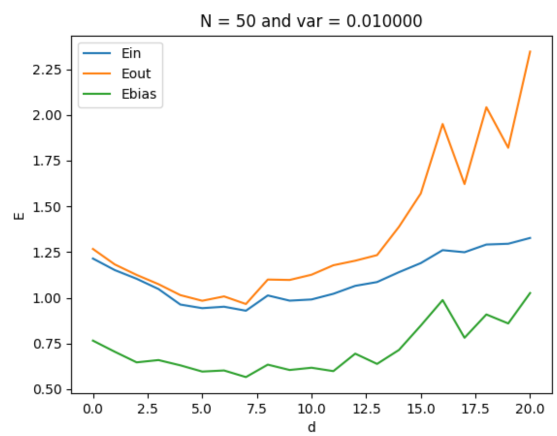
Weight Decay



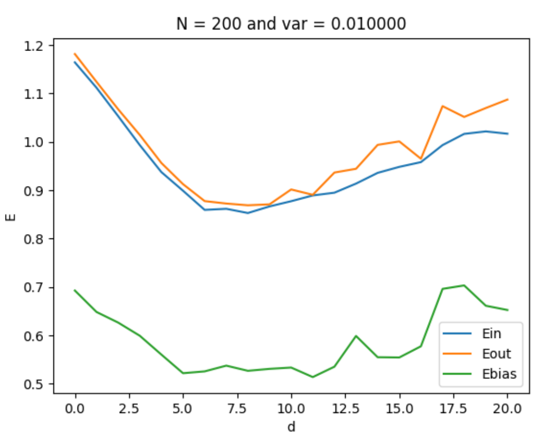
No Weight Decay



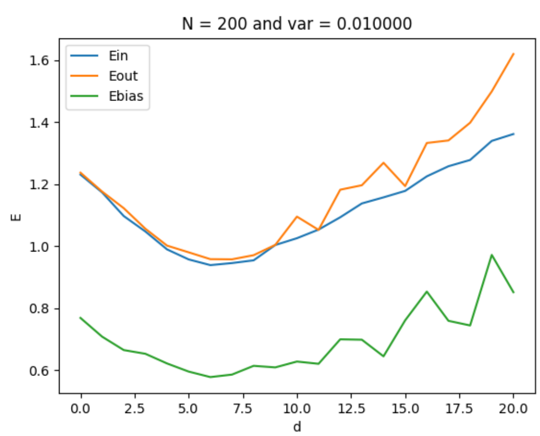
Weight Decay



No Weight Decay

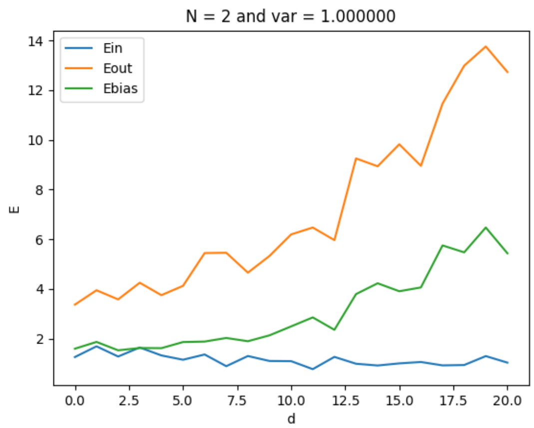


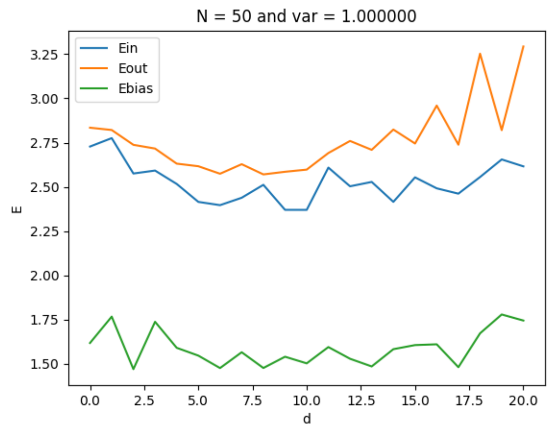
Weight Decay

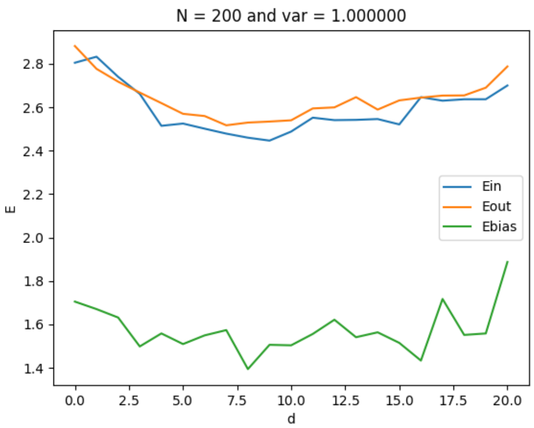


No Weight Decay

The graphs above suggest that estimating coefficients without weight decay regularization resulted in higher MSE; this would imply underfitting. Furthermore, as the sample size increased, the generalization error decreased. However, as the degree of the polynomial increased beyond the optimal solution, the bias error also increased, implying underfitting, as is visible by the rise in the MSE_{in} and MSE_{out} values; this is also visible in larger datasets when the degree is low, such as 0 or 1. For small datasets, The bias error and MSE_{out} increases much more significantly when the degree of the polynomial is increased. Overfitting is visible in small datasets especially at higher degrees as the MSE_{in} remains low while the bias error and MSE_{out} continue to rise.

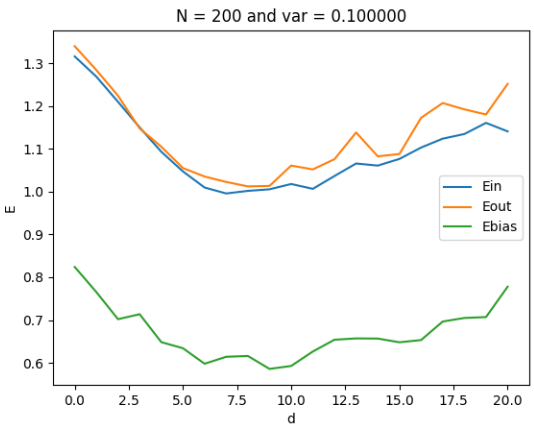


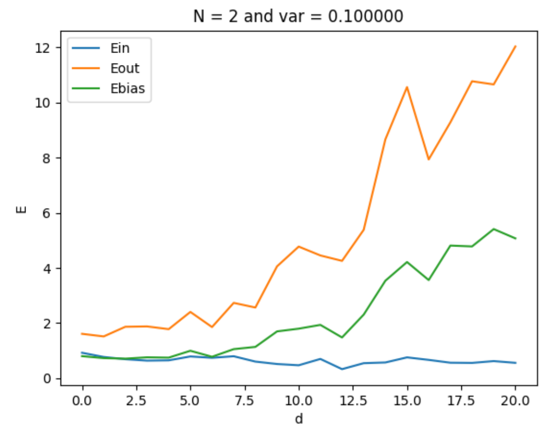


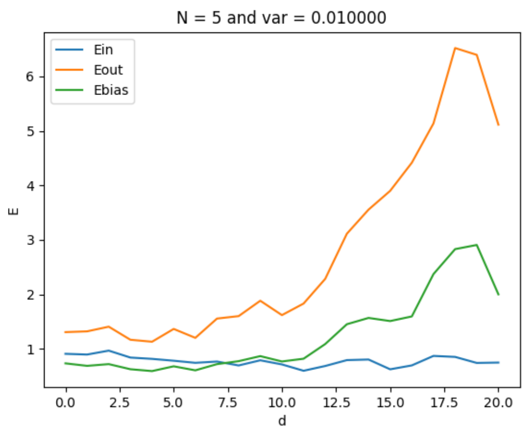


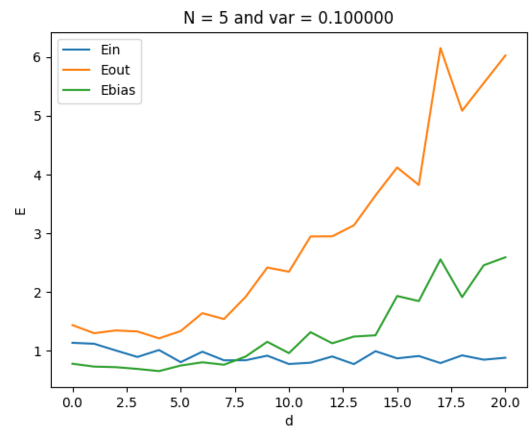
The graphs above are outputs after computing the MSE with the same sample sizes as the previous graphs but with a higher noise level. It is visible that increasing the noise level (σ) increases the MSE for all predictions.

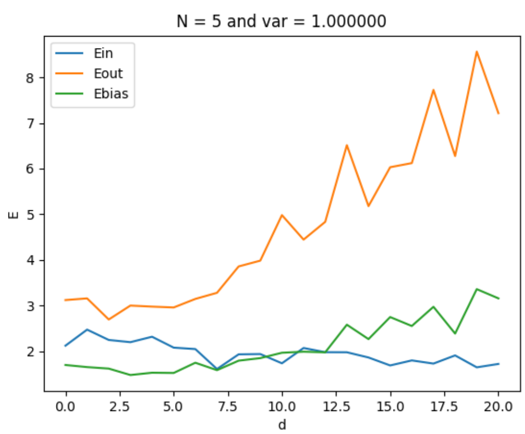
Other output graphs:

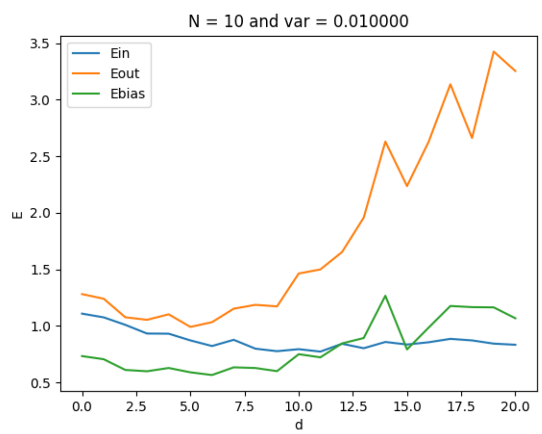


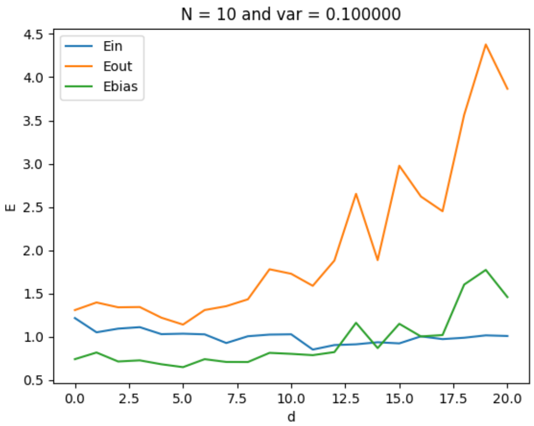


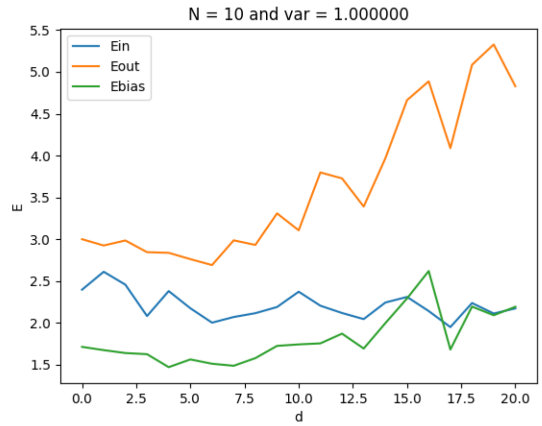


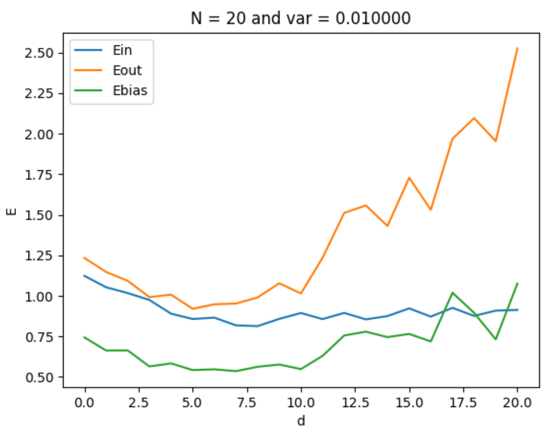


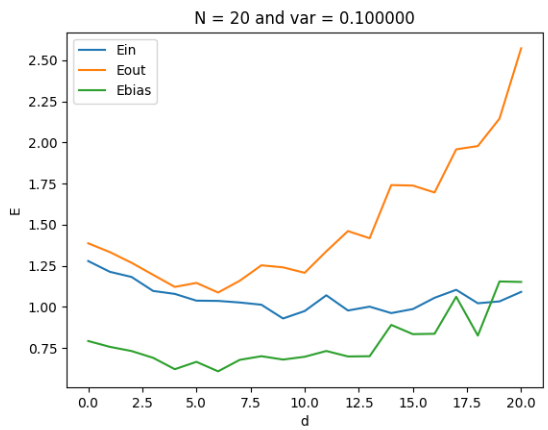


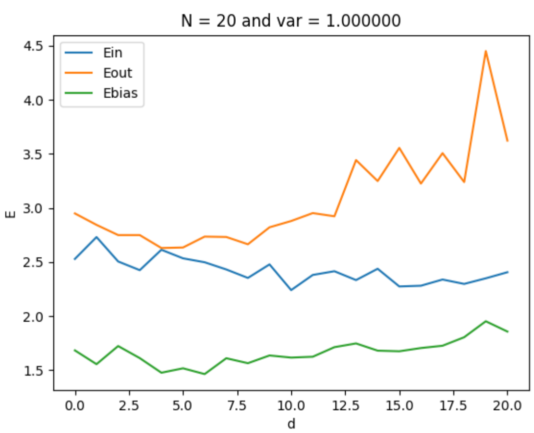


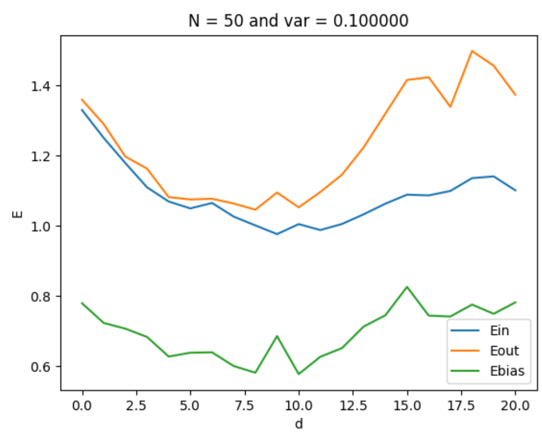

N = 10 and var = 0.100000
Ein
Eout
Ebias
E
d

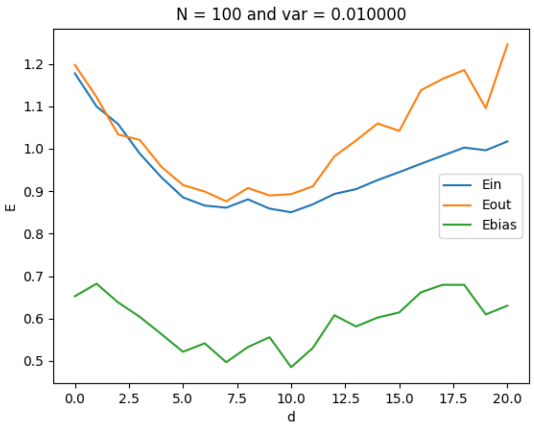

N = 10 and var = 1.000000
Ein
Eout
Ebias
E
d

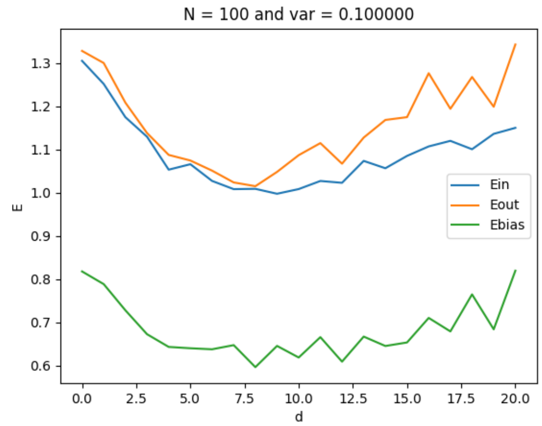

N = 20 and var = 0.010000
Ein
Eout
Ebias
E
d

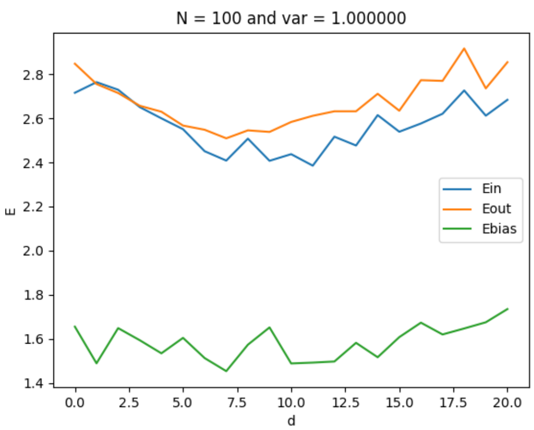

N = 20 and var = 0.100000
Ein
Eout
Ebias
E
d


N = 20 and var = 1.000000
Ein
Eout
Ebias
E
d


N = 50 and var = 0.100000
Ein
Eout
Ebias
E
d


N = 100 and var = 0.010000
E
d
Ein
Eout
Ebias


N = 100 and var = 0.100000
E
d
Ein
Eout
Ebias


N = 100 and var = 1.000000
E
d
Ein
Eout
Ebias