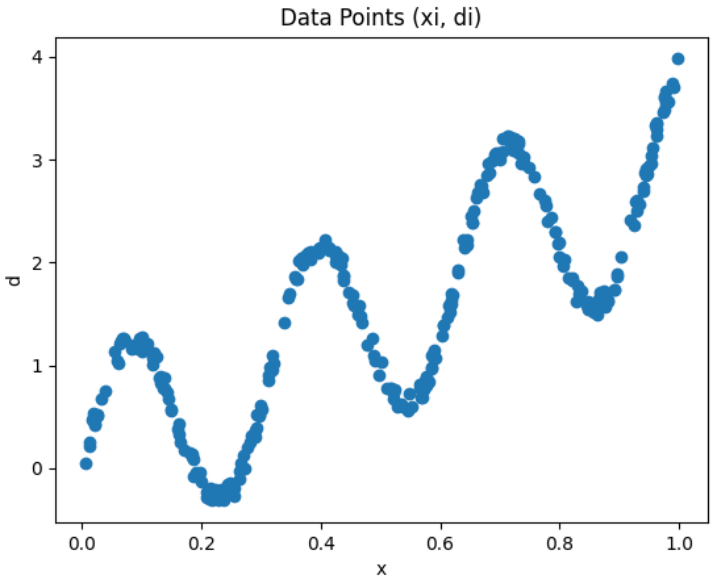
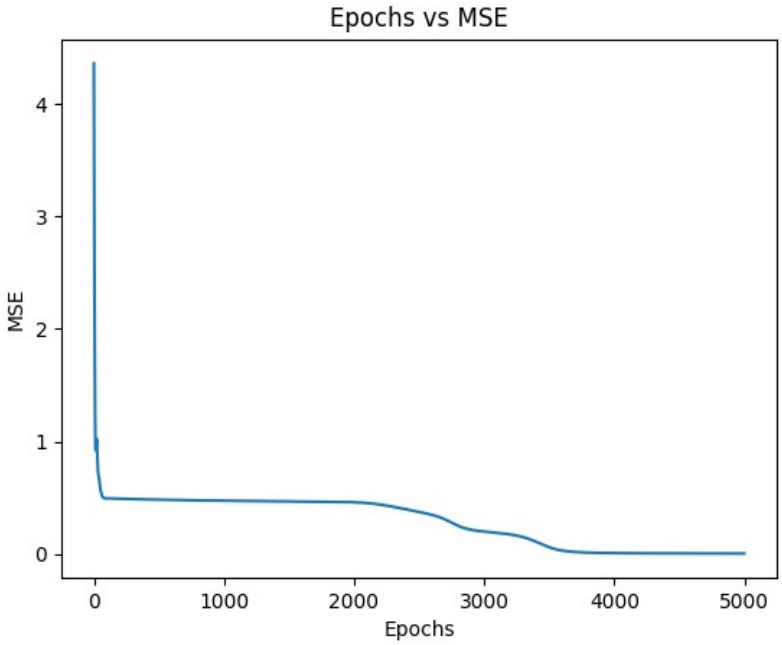
HW4 Yaroslav Popryho
ypopry2@uic.edu

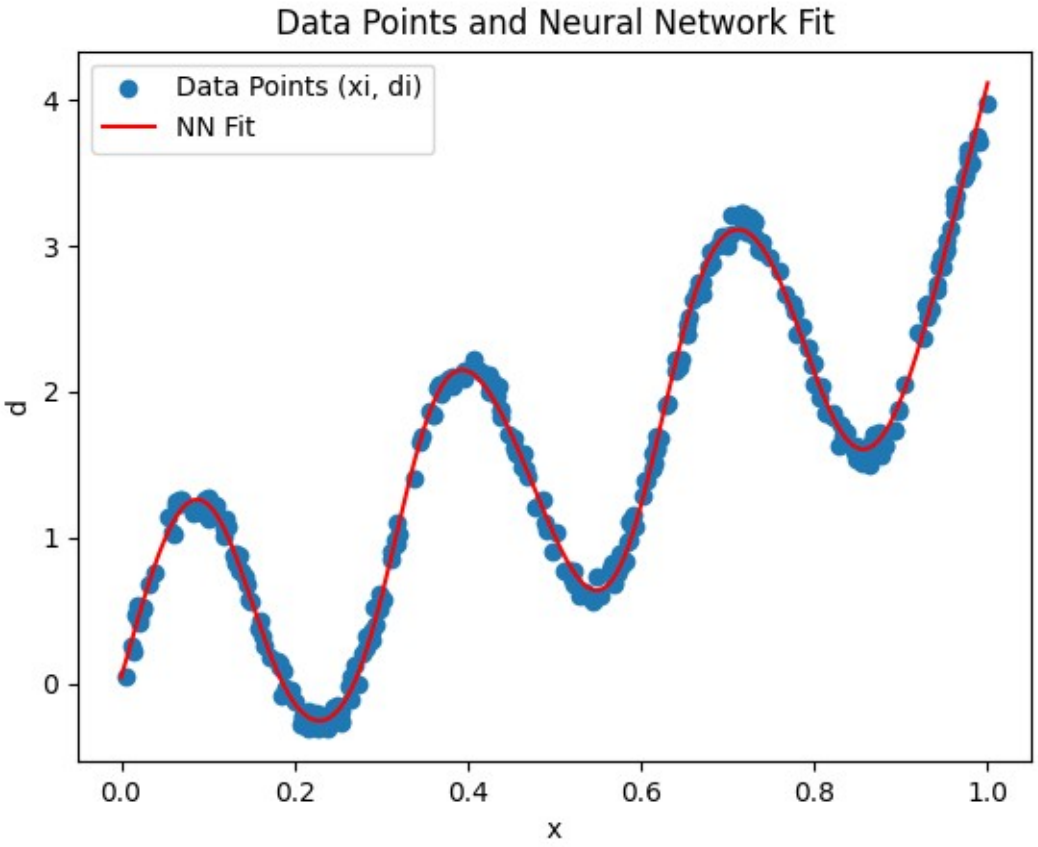
This plot visualizes the data points generated in steps 1-3 of the problem. The x-values are drawn uniformly at random from [0, 1], and the corresponding y-values (or d-values) are computed using the given formula.



This plot tracks the training progress of the neural network. It shows how the mean squared error (MSE) changes over each training epoch. Ideally, as the network learns, the MSE should decrease, indicating that the network's predictions are getting closer to the actual values.



This plot visualizes the final result of the neural network's training. It overlays the neural network's predictions (the fitted curve) on top of the original data points to show how well the network has learned to approximate the data.



6. Pseudocode

```
Initialize:
- Set random seeds for reproducibility
- Define N = 24 (number of hidden neurons)

Generate Data:
1. For i = 1 to 300:
    - Draw x[i] uniformly at random from [0, 1]
    - Draw nu[i] uniformly at random from [-0.1, 0.1]
    - Compute d[i] = sin(20 * x[i]) + 3 * x[i] + nu[i]

Initialize Neural Network:
1. Define input layer with 1 neuron
2. Define hidden layer with N neurons and tanh activation
3. Define output layer with 1 neuron (no activation)
4. Initialize all weights and biases randomly

Training Loop:
1. Set learning rate (lr) to 0.01
2. For epoch = 1 to 5000:
    - For each data point (x[i], d[i]):
        - Forward pass:
            - Compute z_hidden = weights_hidden * x[i] +
bias_hidden
            - Compute hidden_layer_output = tanh(z_hidden)
            - Compute prediction = weights_output *
hidden_layer_output + bias_output
        - Compute error = prediction - d[i]
        - Compute MSE = mean(error^2 over all data points)

        - Backward pass (using chain rule):
            - dMSE/dprediction = 2 * error
            - dMSE/dweights_output = dMSE/dprediction *
hidden_layer_output
            - dMSE/dbias_output = dMSE/dprediction
            - dMSE/dhidden_layer_output = dMSE/dprediction *
weights_output
            - dMSE/dz_hidden = dMSE/dhidden_layer_output * (1 -
tanh^2(z_hidden))
            - dMSE/dweights_hidden = dMSE/dz_hidden * x[i]
            - dMSE/dbias_hidden = dMSE/dz_hidden

        - Update weights and biases using gradient descent:
            - weights_output = weights_output - lr *
dMSE/dweights_output
            - bias_output = bias_output - lr * dMSE/dbias_output
```

*- weights_hidden = weights_hidden - lr * dMSE/dweights_hidden*
*- bias_hidden = bias_hidden - lr * dMSE/dbias_hidden*

Evaluate:
1. For x in range [0, 1]:
- Compute neural network output using trained weights and biases
- Plot (x, output) alongside the original data points (x[i], d[i])