# Assignment 1 – Linear models for regression

Darko Zibar

August 1, 2024

**Exercise 1** (70%)
We consider a system where the input is time, $t$, and the output is described by the following function: $y(t) = \sin(2\pi t)$. The time $t$ is limited to the following interval $t \in [0 : 1]$. We perform a measurement of the system's output at specific times $t_k$, where $k = 0, 1, ...$ is a discrete index. The recorded signal, will be corrupted by the noise, $n_k$, is now expressed as:

$$y_k = \sin(2\pi t_k) + n_k, \quad k = 1, ..., L \tag{1}$$

where $n_k$ is the measurement noise, drawn from the Gaussian distribution with zero mean and variance $\sigma^2$, i.e. $n_k \sim \mathcal{N}(0, \sigma^2)$.

1. Plot $y_k$ given the following time-interval $t_k = 0 : 0.01 : 1$. Assume that the noise is zero. Since the time interval (sampling time) is small $\Delta t = 0.01$s, this will emulate a continues signal $y(t)$.
2. In the same figure, plot $y_k$ (when plotting do not connect the dots) when $t_k$ is drawn from the uniform distribution in the range from 0 to 1, i.e. $t_k \sim \mathcal{UI}[0; 1]$. Choose the length of the vector $t_k$ to be 100. Set the variance of the noise, $\sigma^2$ to 0.09. Comment on the figure by describing the impact of the measurement noise, and the fact that $t_k$ is sampled from the uniform distribution and not regular gird.
3. Plot $y_k$ for the noise variance of $\sigma^2 = 0.09$ and $\sigma^2 = 0.4$ on $y_k$. Comment on the results.

We are now going to assume that we do not know the analytical how $t_k$ is related to $y_k$. The objective is to then learn it from the recorded training data–set, $\mathcal{D} = [t_k, y_k]$, where $k = 1, ..., L$, and $L$ is the length of the data–set.

The training data–set consists of the inputs $\mathbf{X} = [t_1, t_2, ..., t_L]^T$ and the outputs $\mathbf{Y} = [y_1, y_2, ..., y_L]^T$, data, where $T$ denotes the transpose. The data–set is denoted $\mathcal{D}^{L\times 2} = [\mathbf{X}, \mathbf{Y}]$. Since we assume that we do not know the exact analytical expression for the function, $y(t_k)$, we will assume the following model (polynomial):

$$\hat{y}(t_k, \mathbf{W}) = w_0 + w_1 t_k + w_2 t_k^2 + ... + w_M t_k^M = \sum_{j=0}^{M} w_j t_k^j \tag{2}$$

where $M$ and $\mathbf{W} = [w_0, w_1, ...w_M]$ are the unknown polynomial order and the unknown weight vector (adjustable model parameters), respectively. First, we assume a certain value of $M$ and determine the weight vector, and later we determine the optimum $M$.

1. Use the lecture slides and implement the gradient descent based iterative algorithm for determining the weights $\mathbf{W} = [w_0, w_1, ...w_M]$. Assume $L = 100$, $\sigma^2 = 0.2$, $M = 3$, $\eta = 0.3$. Use $L_{iter} = 3000$ iterations for the gradient descent.

2. Plot the mean square error between the learned model, $\hat{y}(t_k, \mathbf{W})$ and the training data $\mathbf{Y} = [y_1, y_2, ..., y_L]^T$ as a function of the number of iterations, $L_{iter}$. What can be concluded by observing the evolution of the mean square error? Comment on if we are able to learn the model between the input, $t_k$, and the output data, $y_k$.

3. Evaluate the learned model, $\hat{y}(t_k, \mathbf{W})$, by testing it for the unseen input data given by the following vector $t'_k = 0 : 0.01 : 1$. Plot $\hat{y}(t_k, \mathbf{W})$ together with $y_k = \sin(2\pi t'_k)$. Plot also the training data. Comment on the results!

4. Repeat task 3 by varying the length of $L$ from 10 to 100 in steps of 10. Adjust $\eta$ and the number of iterations to obtain the best fit. Assume $\sigma^2 = 0.3$ and $M = 3$. What is the impact of decreasing the training data set $L$?

Instead of using the gradient descent to determine the weights $\mathbf{W}$, it is more convenient to use Moore-Penrose Pseudo–Inverse (MPPI). The weight vector $\mathbf{W}$ can then be obtained in a single step without the need to adjust the step–size $\eta$.

1. Use the lecture slides and implement the MPPI method for determining the weights $\mathbf{W} = [w_0, w_1, ...w_M]$. Assume $L = 100$, $\sigma^2 = 0.3$, $M = 3$.

2. Evaluate the learned model, $\hat{y}(t_k, \mathbf{W})$, by testing it for $t' = 0 : 0.01 : 1$ and plotting it against $y_k = \sin(2\pi t'_k)$. Compare the results with the ones obtained using the gradient descent by plotting the results in the same figure.

3. Set the length of the data–set $L$ to 100 $\sigma^2 = 0.9$. Employ 10–fold cross-validation for learning and testing the model. This implies that you use 90 points for training and 10 for testing. By shuffling the test–set through the data–set you will obtain 10 test–sets and therefore you will obtain 10 different values for the weight vector $\mathbf{W}$. (Please consult the slides on how to perform cross–validation.

4. Evaluate and plot the root mean square error on the training and test–sets for each of the folds. The root mean square error is defined as:

$$E_{RMS} = \sqrt{\left(\frac{1}{L_{train/test}} \sum_{k=0}^{L_{train/test}} [y_{k,train/test} - \hat{y}(x_{k,train/test}, \mathbf{W})]^2\right)} \tag{3}$$

where $L_{train/test}$ is the length of the training and the test–set, respectively. $x_{k,train/test}$ and $y_{k,train/test}$ are the input/output values belonging to the training and test–set, respectively.

5. Plot the training error and test root mean square error as a function of the polynomial order $M$ ranging from 1 to 11. How do we find the right polynomial model order?

**Exercise 2** (30%)
The goal of the following exercise is to implement Rosenblatt perceptron and use it for learning logical gates functions.

1. Implement AND,OR and XOR table.

2. Generate a training and test–set by performing uniform sampling from AND table. Set the length of the training and test–set to 2000, respectively. HINT: to perform uniform sampling first generate a vector `X = randi(4,1,L)`, where `L` is the length of the vector.

3. Use the slides (book) and implement a Rosenblatt perceptron. Plot the mean square error computed on the training and the test–set. Use $\eta = 10^{-6}$. Comment on the results!

4. Repeat item 2–3 for OR and XOR gate. Can you learn OR and XOR gate? Comment on the results