AM160 Homework 1

1 Problem 1

1.1 Subproblem A (5 points)

Consider n observations, from a data distribution, D, given as: (x_1, y_1), (x_2, y_2), $(x_2, y_2) \cdots$ (x_n, y_n). We formulate a linear regression problem with feature matrix, A, containing p features using the basis function map $\Phi : R^1 \to R^p$. So A is given by:

$$\mathrm{A}_{(n\times p)} = \begin{bmatrix} \Phi_1(x_1) & \Phi_2(x_1) & \Phi_3(x_1) & \cdots & \Phi_p(x_1) \\ \Phi_1(x_2) & \Phi_2(x_2) & \Phi_3(x_2) & \cdots & \Phi_p(x_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \Phi_1(x_n) & \Phi_2(x_n) & \Phi_3(x_n) & \cdots & \Phi_p(x_n) \end{bmatrix}$$

The unknown parameters, we want to estimate is given by $\Theta_{p\times 1}$. Prove that, if $p \gg n$, then the solution to: $A\Theta = y$, which is given by $\Theta^* = A^\dagger y$ has the minimum 2-norm amongst all the other infinite solutions.

1.2 Subproblem B (10 points)

Generate training and testing dataset from the generating function: $y_n = 2x_n + cos(25x_n) + N(0, 1)$, where N is a Gaussian distribution. To fit a linear model to this function, develop a random Fourier feature (RFF) model, where the basis function $\phi : x_n \to cos(2\pi\omega_i x_n)$. Here, ω_i is sampled from the Gaussian $N(0, 1)$ and $i \in [1, 2, 3, \cdots, p]$. Write a Pytorch code to demonstrate double (multiple) descent(s) with this model. Your submission should be a PDF that shows training and testing error as a function of the number of features, p. Even if you do not get double (multiple) descent(s), explain your procedure and what steps you took to arrive at it.

2 Problem 2 (7.5 + 7.5 points)

. Consider the SinDY problem we solved in the class. Use the same Lorenz 63 dataset and generate $10,000$ temporal samples. You are going to implement sinDY to discover the Lorenz 63 equations in the following settings:

(a) Assume you have the true derivatives, but they are noisy. Simulate that by adding a Gaussian with variance 0.01, 0.1, and 0.5. Present your results in a way that shows how well the accuracy of your discovered coefficients improve/worsen with increase in noise. Play with the maximum number of iterations in STLSQ algorithm or with the number of basis functions.

(b) Compute the derivatives using data of $x(t)$, $y(t)$, and $z(t)$. Use first order finite difference, or central difference, or any other method. Perhaps try fitting a cubic spline using every 4 consecutive temporal samples and then computing the derivative using the splines. Finally, report your discovered coefficients in a way that shows how the effect of your derivative scheme reflects in your discovered coefficients