**COMP3223 – Foundations of Machine Learning**

Coursework

Introduction

Given the loose format of this coursework I have added structure to help with referencing. When something is referred to as *given* (i.e. given by the coursework documentation) I have attached a reference in the format “(ref-[section].[subsection])” to the section in the coursework documentation as to where it is given. I have also implemented a sub-section under each section labelled *Comments* where I talk about the structure of the program and show my understanding of the mathematics and key concepts.

2 Linear Regression with Non-linear Functions

Using linear regression we aim to curve fit two basis functions to y(x) = sin(4𝜋x) + ϵ using *N* data points randomly generated, where 0 ≤ x ≤ 1 for n=1, …, N. Therefore, data points will be taken as (xn, yn) pairs. The random generation of ***x*** (vector list of *xn*) was given(ref-2.0). The function developed to determine ***y*** (vector list of *yn*)is as follows

Where noise is an array of 0 ≤ ∊ ≤ 0.1 randomly chosen for each data point.

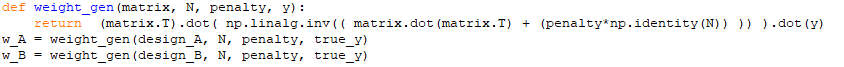
2.1 Construction of Linear Regression Functions

2.1.1 Program Definition

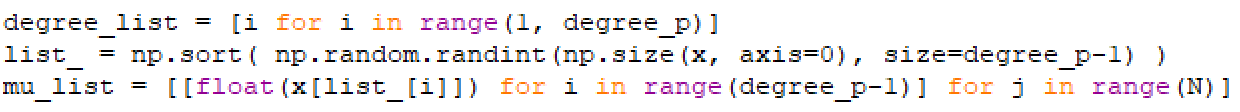
In this section two classes of functions are used, *Polynomial* basis function and *Gaussian* radial basis function, to develop two design matrixes, lets call them matrix A and matrix B, respectively. The functions used to determine matrix A and B are given(ref-2.1), however, input generation is not – this is discussed later. The basis functions have a degree of 15 and therefore the program will generate 15 values of weight. The equation used to generate the *optimal* weights using any design matrix follows



(1)

Where *N* is the number of data points (in this case 15), *λ* is the strength of regularisation coefficient (chosen as 10-7). The function that represents the weight generation for matrix A and B follows

Where ***w\_A*** and ***w\_B*** represent the vector-lists of weights for design matrix A (*design\_A*) and B (*design\_B*), respectively, *penalty* represents *λ* and *true\_y* is **y** (vector-list of *yn values*).

Though the functions for design matrix generation are given(ref-2.1), additional inputs (other than **x)** are not. For design matrix A, an ordered vector-list of degrees is necessary to fulfil *j* within Φj(xn)=xnj , where degrees is represented by0≤ j ≤ 15. For design matrix B, a vector list of centroids are necessary to fulfil ***xj*** within Φj(xn)=exp(-(xj-xn)/2σ2), where σ = 0.1(given(ref-2.1)) and 0≤ j ≤ 15. They are both characterised using the following definitions.

Where *degree­\_p* is the degree as an integer (=*N*),***x*** is a representation of the vector-list *xn*, *degree\_list* is the vector input *degree* in the function *polynomial\_basis\_fn(·)* and *mu\_list* is the input *mu* in the function *gaussian\_basis\_fn(·)*, both of which are given(ref-2.1).

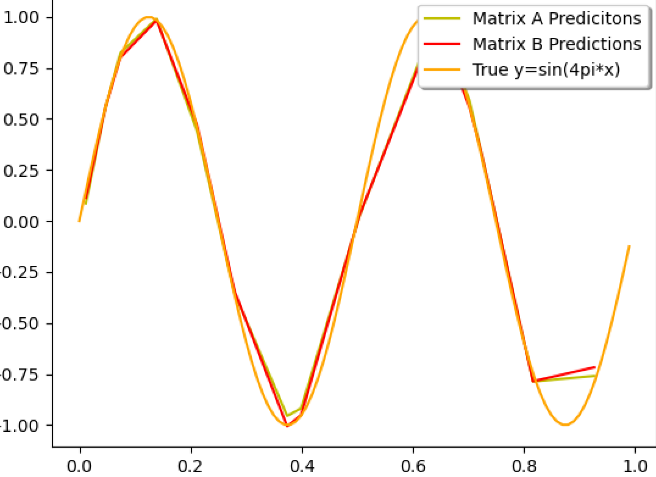
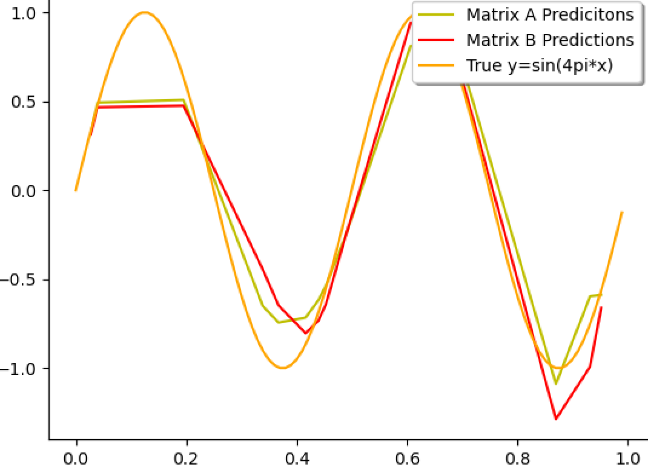
2.1.2 Comments

When developing my program, I decided to generate training and testing data before I realised it was asked of us in section 2.2 within the coursework documentation.

My understanding of what was asked was that *N*=15 data points were to be generated randomly, used and reused for training and test data. In my opinion, 15 training points is too low and will not allow us to develop reliable values for **w** (labelled ***w\_A***and ***w\_B*** in program), so while testing I felt it was more appropriate to use *N*=1 000 training points for training and *N*=15 **different** testing points.

As *λ* had not been determined, I assigned a value that I thought was appropriate (=10-7), this was subjectively determined with the help of the debugging part of the program.

While debugging my program I determined **yhat**, my y-coordinate prediction given input **x**, using **yhat** = **w**·**x**, where **x** is my test data. I plotted (x, yhat) for both matrix A and B, and compared my plot to the true y(x)=sin(4𝜋x). Below is a visual representation of the difference between 15 training and 1 000 training points. It is evident from the placement of the points in relation to the orange line that the 1000 training provides a more reliable **y**-prediction than 15 points.



N=15 training points

N=1000 training points

2.2 Generalisation

In this section I aim to answer questions from section 2.2 *Generalisation* in the coursework documentation.

2.2.1 Data Split

2.2.2 Parameter Tweaking (degree\_p and lambda)

2.2.3 Comments