Numerical Methods Lab 3

Lagrange Interpolation [C02]

- i. Open the colab file shared in BUX.
- ii. Create a copy of that shared file.
- iii. Rename the colab filename using the format Name-ID-Lab Section

Lab Introduction

We know that, general form of an n degree Lagrange polynomial:

$$p_n(x) = \sum_{k=0}^n f(x_k) l_k(x) = \sum_{k=0}^n y_k l_k(x)$$

where

$$l_k(x) = \prod_{j=0, j
eq k}^n rac{x-x_j}{x_k-x_j}$$

Now, check out the Lagrange_Polynomial class in the given code.

- 1. The constructor __init__(self, data_x, data_y) is written for you. (No task here)
- 2. The repr (self) function has been written for you. (No task here)

3. [Task 1] - 4 marks

You have to implement the l(self, k, x) function.

This method implements the Lagrange Basis to be used for interpolation using Lagrange Polynomials. This function would take k and x as inputs and calculate the Lagrange basis using the second Equation given above.

You will have to remove the "raise NotImplementedError()"

Hint: Set up a Loop to traverse through. Or you can use vectorized method.

4. [Task 2] - 4 marks

You have to implement the __call__(self, x_arr) function.

The function calculates the lagrange polynomial from a set of given nodes using the first equation given above.

You will have to remove the "raise NotImplementedError()"

Hint: The method to make the object callable. 'x_arr' is a set of given points (a numpy array). You have to use self.data_x and self.data_y to find the interpolated output of the polynomial for all elements of 'x_arr'. Implement as you wish but your 'total' numpy array where the i'th element p_x_arr[i] represents the interpolated value of p(x_arr[i]). You can use nested for loop to complete this task.

5. Plotting the polynomial (No task here)

6. [Task 3: Problem related Lagrange interpolation] – 2 marks

You will have to solve the given problem using Lagrange Polynomial class.