Programming Project PART 2
CS 323 - Numerical Analysis

1 Introduction

For this second part of the project you will write matlab code for the following algorithms. Instructions are the same as in the first part of the project.

2 Algorithms

The following is the list of all the algorithms that you must implement for this second part:

1. **Cubic Splines:** Given a set of points find the coefficients of all the 3rd degree polynomials that go through all the given points using the cubic splines method.

   Input:

   Sample input representing the points $(1,5),(2,6),(3,6.5),(4,5.5),(5,5.5),(6,7)$

   ```
   6
   1
   5
   2
   6
   3
   6.5
   4
   5.5
   5
   5.5
   6
   7
   ```

   where the first number corresponds to the number of points.

   Output:

   (a) Coefficients of $n-1$ polynomials. The coefficients of each polynomial in a row separated by spaces (see lecture notes for lecture 16):

   ```
   5 1.014354 0 -0.014354
   6 0.971291667 -0.043062 -0.42822967
   6.5 -0.39952167 -1.327751 0.727272667
   5.5 -0.87320567 0.854067 0.019138667
   5.5 0.89234467 0.911483 -0.30382767
   ```

   (b) A plot like the one shown on lecture notes (16).

2. **Least Squares:** Given a set of points you will find a polynomial of degree n that approximates the points minimizing the error squared.

   Input:

   Sample input representing the points $(0, 45.13)$, $(1, 51.71)$, $(2, 60.17)$, $(3, 64.83)$, $(4, 65.24)$, $(5, 65.17)$, $(6, 67.65)$, $(7, 79.8)$, $(8, 96.13)$, $(9, 115.19)$,

```
10
3
0
45.13
1
51.71
2
60.17
3
64.83
4
65.24
5
65.17
6
67.65
7
79.8
8
96.13
9
115.19
```

   where the first number corresponds to the number of points and the second number is the degree of the desired polynomial.

   Output:

   (a) The coefficients of the polynomial, separated by spaces:

   ```
   43.9719 13.4363 -3.3048 0.3005
   ```

   (b) A plot like the one shown on lecture notes (18_matlab_examples).

3. **Simpson's Rule:** Compute the definite integral of a given function using the *Composite Simpson's rule*.

   Input: Function (of x) as a string (use matlab's `inline` and `feval`), the limits of integration (a and b), and the number of subintervals. Sample input:

   ```
   x^3
   1
   5
   20
   ```

   Output: The approximate value of

   $$\int_a^b f(x)dx$$

   using Simpson's rule with n subintervals