भारतीय प्रौद्योगिकी संस्थान गुवाहाटी
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

MA 322: Scientific Computing Lab

Lab 02

AB Satyaprakash (180123062)

8th Feb 2021

Question 1.

Given $f(x) = e^x$, for $0 \leq x \leq 2$.

The Lagrange Basis for P_n is $l_0(x), \ldots, l_n(x)$, where

$l_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i}$, $j = 0 : n$

The p(x) is then obtained by,

$p(x) = f_0 l_0(x) + \ldots + f_n l_n(x)$

Using the above relations,

1. Setting n = 1, the approximation of **f(0.25)** using linear Lagrange interpolation with x0 = 0 and x1 = 0.5 is **1.324360635350064**
2. Setting n=1, the approximation of **f(0.75)** using linear Lagrange interpolation with x0 = 0.5 and x1 = 1 is **2.183501549579587**
3. Setting n=2,
    a. The approximation of **f(0.25)** using second Lagrange interpolating polynomial with x0 = 0, x1 = 1, and x2 = 2 is **1.1527742906760838**
    b. The approximation of **f(0.75)** using second Lagrange interpolating polynomial with x0 = 0, x1 = 1, and x2 = 2 is **2.0119152049056064**
4. To find which approximation is better, it's good to find the absolute error using both linear and second Lagrange polynomials and see which

has the least error as compared to the actual values at the calculated points.

`Error for linear approximation of f(0.25) is` **`0.0403352186623227`**

`Error for linear approximation of f(0.75) is` **`0.06650153296691208`**

`Error for the second polynomial approximation of f(0.25) is` **`0.13125112601165756`**

`Error for the second polynomial approximation of f(0.75) is` **`0.1050848117070684`**

Clearly, linear Lagrange interpolation gives a better approximation to obtain f(0.25) and f(0.75)

Note - For the same value of h, estimates made using a higher degree polynomial are supposed to be better, however, in this case, the value of h increases in the latter case. This results in the estimates being further from the actual value for the second Lagrange polynomial.

Question 2.

Using the same method of Lagrange interpolation as in question 1, i.e,

The Lagrange Basis for P_n is $l_0(x), \ldots, l_n(x)$, where

$l_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i}, \; j = 0 : n$

The p(x) is then obtained by,

$p(x) = f_0 l_0(x) + \ldots + f_n l_n(x)$

We in this case set n = 1, 2, and 3 for each of the 2 cases and approximate the asked values.

a. Approximation of **f(8.4)** using the Lagrange interpolating polynomials of degrees one, two, and three are:

Degree 1 = 17.87833

Degree 2 = 17.877129999999998

Degree 3 = 17.8771425

b. Approximation of f(-1/3) using the Lagrange interpolating polynomials of degrees one, two, and three are:

Degree 1 = 0.2150416666666667

Degree 2 = 0.1803055555555556

Degree 3 = 0.17451851851851857

Question 3.

Again using the same Lagrange interpolating polynomial formulae and setting degree as 2 with the function f(x) = e^{-x^2} at the nodes $x_0 = -1,\ x_1 = 0\, and\, x_2 = 1$, we get

$P_2(x) = -0.6321x^2 + 1$

The value of P2(0.9) is 0.48798234734886825

The value of P2(0.9) rounded to 6 decimal places is 0.487982
The true value of f(0.9) is 0.4448580662229411.
The max error in this calculation is 0.111250691215829(using the method on the last page of lecture 5. Also, note that the error in our case is smaller!)

Question 4.

Using the Lagrange interpolating polynomial of degree 3 with the function f(x) = $\log_{10}(\tan x)$ at the nodes
$x_0 = -1,\ x_1 = 1.05,\ x_2 = 1.10,\ and\ x_3 = 1.15$, we get

`The third Lagrange polynomial approximation for f(1.09) is 0.2826352`

`The third Lagrange polynomial approximation for f(1.09) rounded to 4 decimal places is 0.2826`

`The bound for the error in this approximation is 0.00000733574633048291` `(using the method on the last page of lecture 5)`

Question 5.

In this question, we need to compute an interpolating polynomial $P_n([f_0(x), \ldots, f_n(x)])$, for the function *erf*, where
$erf = \frac{2}{\sqrt{\pi}} \int_0^x e^{-s^2}(s)ds$

Since I have written code using python, I have used the erf function available inside the math package. The nodes are given as {1, 1.2, 1.4, ..., 3}, and we need to use 3 different bases for this question, namely:

a. Monomial
b. Lagrange
c. Newton

a. **Monomial Basis**

Consider the nodes [x0, . . . , xn] and the values [f0, . . . , fn].

Consider the monomial basis $1, x, x^2, \ldots, x^n$, which corresponds to the choice $\phi_j(x) = x_j \, for \, j = 0 : n$. Then $p(x) = a_0 + a_1 x + \ldots + a_n x^n$ is obtained by solving the Vandermonde system

$$\begin{bmatrix} 1 & x_0 & . & . & x_0^n \\ 1 & x_1 & . & . & x_1^n \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & x_n & . & . & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ . \\ . \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ . \\ . \\ f_n \end{bmatrix}$$

Using the given data, we obtain the polynomial in this case as:

$$-0.0002411x^{10} + 0.004953x^9 - 0.04417x^8 + 0.2216x^7 - 0.672x^6$$
$$+1.216x^5 - 1.161x^4 + 0.3977x^3 - 0.3009x^2 + 1.182x - 0.001038$$

b. **Lagrange Basis**

The Lagrange Basis for P_n is $l_0(x), \ldots, l_n(x)$, where

$l_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i}, \; j = 0 : n$

The p(x) is then obtained by,

$p(x) = f_0 l_0(x) + \ldots + f_n l_n(x)$

Using the given data, we obtain the polynomial in this case as:

$$-0.0002411x^{10} + 0.004953x^9 - 0.04417x^8 + 0.2216x^7 - 0.672x^6$$
$$+1.216x^5 - 1.161x^4 + 0.3977x^3 - 0.3009x^2 + 1.182x - 0.001038$$

c. **Newton Basis**

Consider the nodes [x0, . . . , xn] and the values [f0, . . . , fn].

Define $N_0(x) = 1 \, and \, N_j(x) = (x - x_0) \ldots (x - x_j) \, for \, j = 1 : n$.

Let $p(x) = a_0 N_0(x) + \ldots + a_n N_n(x)$. The interpolation conditions $p(x_j) = f_j \, for \, j = 0 : n$ yield the lower triangular system

$$\begin{bmatrix} N_0(x_0) & 0 & . & . & 0 \\ N_0(x_1) & N_1(x_1) & 0 & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ N_0(x_n) & N_1(x_n) & N_2(x_n) & . & N_n(x_n) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ . \\ . \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ . \\ . \\ f_n \end{bmatrix}$$

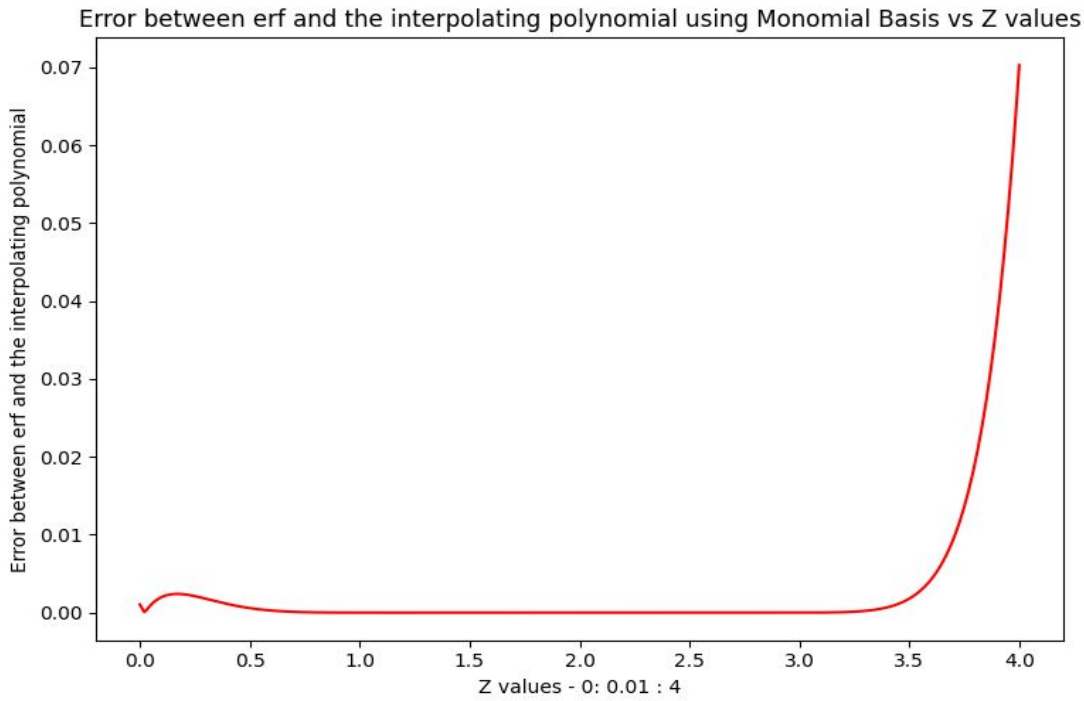
Using the given data, we obtain the polynomial in this case as:

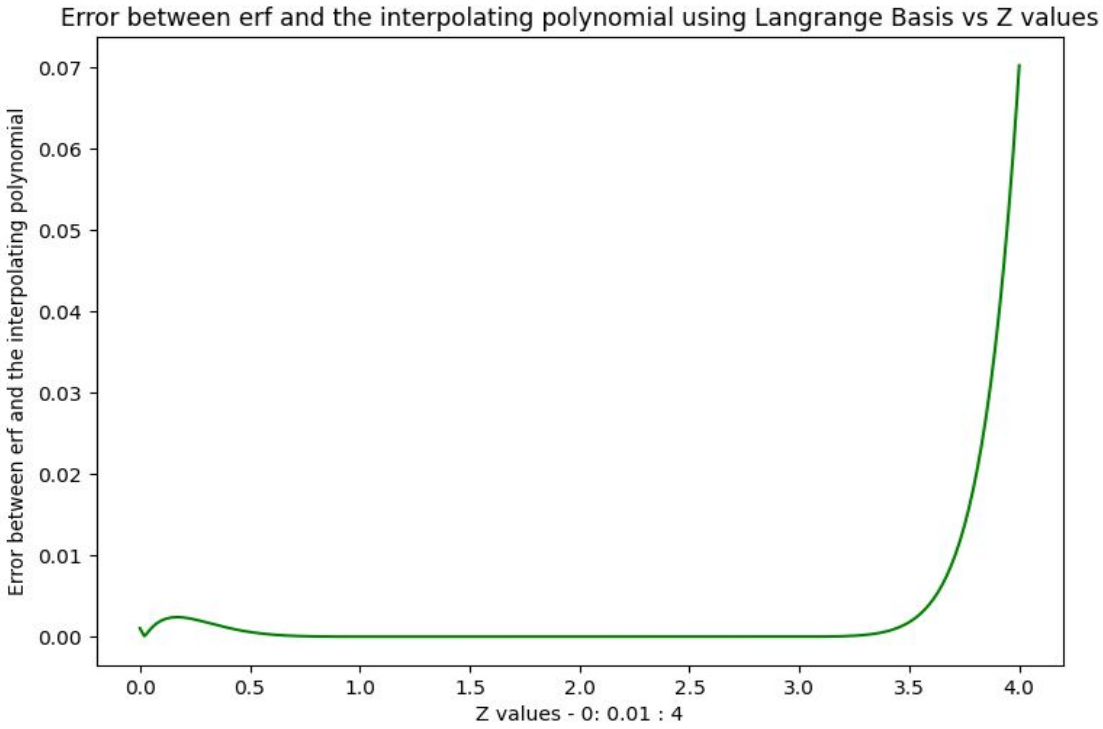
$-0.0002411x^{10} + 0.004953x^9 - 0.04417x^8 + 0.2216x^7 - 0.672x^6$
$+1.216x^5 - 1.161x^4 + 0.3977x^3 - 0.3009x^2 + 1.182x - 0.001038$

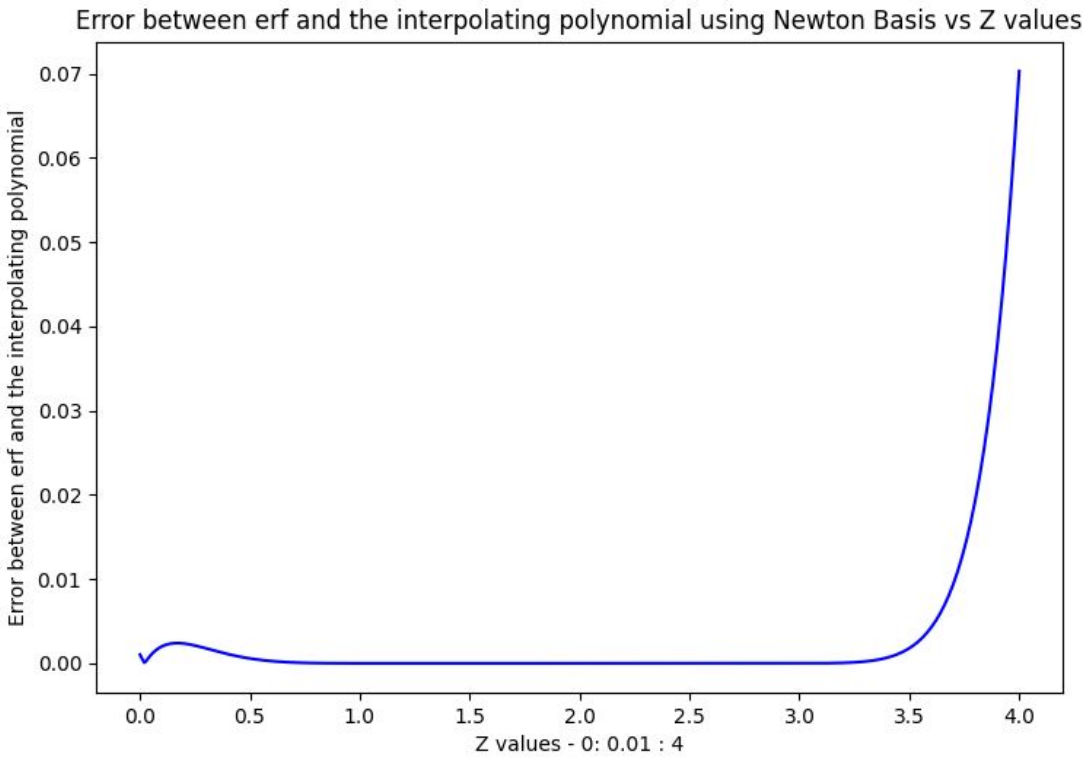
After this, we will use the above polynomials to plot the error between actual and approximate values for each of the 3 cases. The plots are attached below.

We observe that the errors (absolute) increase, as we move out of the range 1 to 3, which is obvious because we are approximating with a polynomial in this range only.

So it is not recommended to use polynomial interpolation to approximate erf at points outside [1 3]

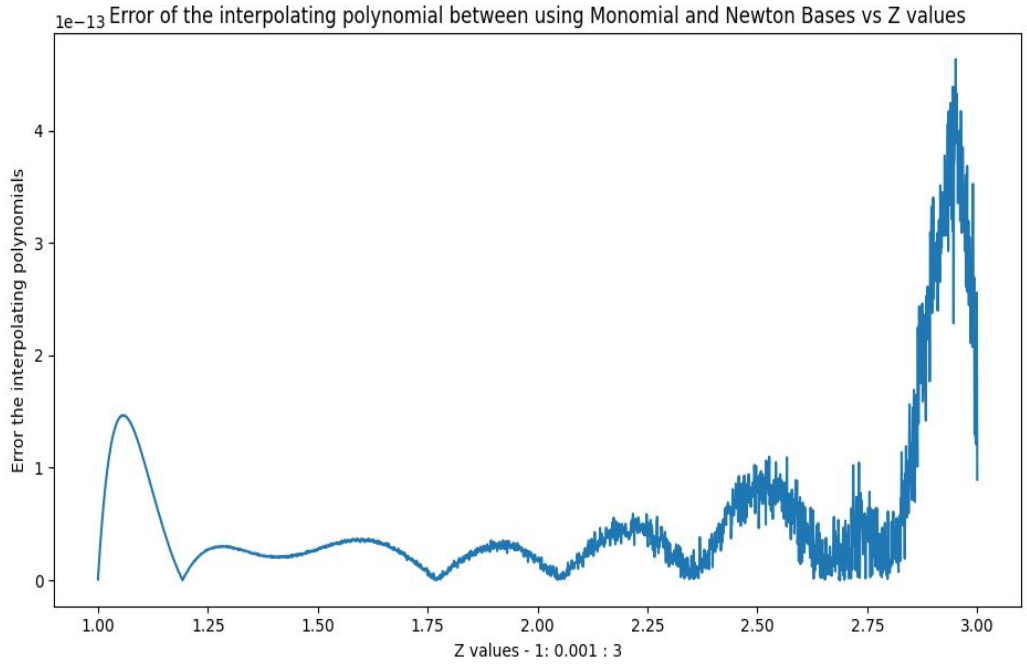

Error between erf and the interpolating polynomial using Monomial Basis vs Z values
Error between erf and the interpolating polynomial
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Z values - 0: 0.01 : 4

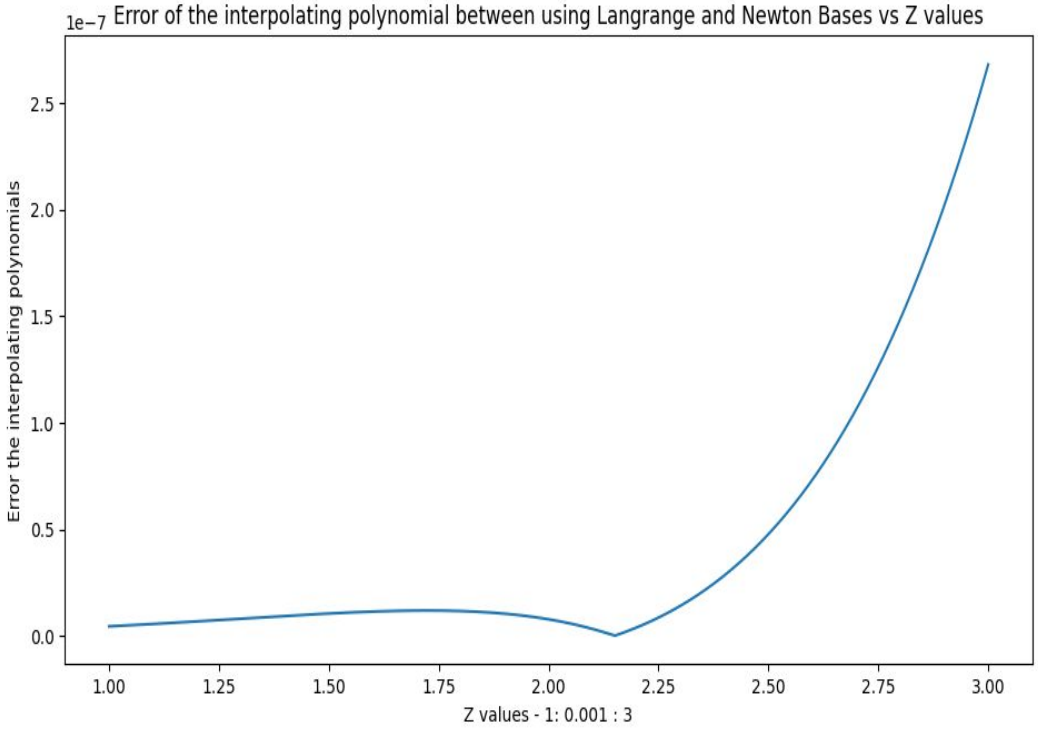

Error between erf and the interpolating polynomial using Langrange Basis vs Z values
Error between erf and the interpolating polynomial
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Z values - 0: 0.01 : 4



Next, we plot the error between the interpolation polynomial computed using the monomial basis and the Newton basis and plot the error between the interpolation polynomial computed using the Lagrange basis and the Newton basis.

The values obtained by using the Lagrange and the Newton basis are very close to each other, with the error between them being in the range of 1e-10. However, the values obtained using the Monomial Basis, are further apart than these two. Thus the monomial basis introduces the largest error in the polynomial interpolation.


1e−13
Error of the interpolating polynomial between using Monomial and Newton Bases vs Z values
Error the interpolating polynomials
0
1
2
3
4
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Z values - 1: 0.001 : 3


1e−7
Error of the interpolating polynomial between using Langrange and Newton Bases vs Z values
Error the interpolating polynomials
0.0
0.5
1.0
1.5
2.0
2.5
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Z values - 1: 0.001 : 3