MA348 Numerical Analysis, Function Interpolation

David Jefts

April 8th, 2019

Introduction

The purpose of this lab is to use Newton's Divided Difference Polynomials and Lagrange Polynomials to estimate the value of a function at any given point, using only a set of $x-$ and $y-$values. The function to be estimated is the natural logarithm function at the point $x = 2$ ($ln(2)$). The estimated line was drawn with the following points:

x	y
(1,	0.0)
(4,	1.386294)
(5,	1.609438)
(6,	1.791759)

The Lagrange Polynomial portion of this lab is based on the fact that interpolating a line through four distinct points requires a minimum polynomial degree of three.

Theory-Analysis

Newton's Divided Differences Interpolation Polynomial is a polynomial used to estimate the value of a function when the exact function and its line are not known. This lab was based on creating a polynomial of degree three using four distinct points. The Interpolation Polynomial combines a series of successive Newton Differences into a polynomial using the formula:

$$N(x) = [y_0] + [y_0, y_1](x - x_0) + \ldots + [y_0, \ldots, y_k](x - x_0)(x - x_1)\ldots(x - x_k) \quad (1)$$

The bracketed y values ($[y_0]$, $[y_0, y_1]$, etc.) are Newton Divided Differences of varying sizes based on the following formula:

$$\begin{aligned}
&\text{Degree one:} && [y_0] = f(x_0)\\
&\text{Degree two:} && [y_0, y_1] = \frac{[y_1] - [y_0]}{x_1 - x_0} = \frac{f(x_1) - f(x_0)}{x_1 - x_0}\\
&\text{Degree three:} && [y_0, y_1, y_2] = \frac{[y_1, y_2] - [y_0, y_1]}{x_2 - x_0} = \frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2 - x_0}
\end{aligned}$$

There were no assumptions made in this lab.

Numerical Solution

The entirety of the Newton Divided Differences interpolation method was coded in Python, with the report created in LaTeX.

The divided differences are based on finding derivates, which are used to draw a slope between two points. Usually they are depicted in a table, since each successive divided difference uses the divided differences from the previous iteration and the next point like so (where {DD n} stands for divided difference of degree n):

x input	DD 1	DD 2	DD 3
x_0	$f(x_0)$		
		$\frac{f(x_1)-f(x_0)}{x_1-x_0}$	
x_1	$f(x_1)$		$\frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2-x_0}$
		$\frac{f(x_2)-f(x_1)}{x_2-x_1}$	
x_2	$f(x_2)$		$\frac{\frac{f(x_3)-f(x_2)}{x_3-x_2} - \frac{f(x_2)-f(x_1)}{x_2-x_1}}{x_3-x_1}$
		$\frac{f(x_3)-f(x_2)}{x_3-x_2}$	
$\vdots$	$\vdots$		$\vdots$
		$\vdots$	
$\vdots$	$\vdots$		$\vdots$
x_n	$f(x_n)$		

The highest entry in each column is then used as the coefficient of the interpolation polynomial. Each coefficient is multiplied by $(x - x_i)$ $(k - 1)$ number of times with the following formula:

$$N(x) = \mathrm{DD1} + \mathrm{DD2}(x - x_0) + \mathrm{DD3}(x - x_0)(x - x_1) + \mathrm{DD4}(x - x_0)(x - x_1)(x - x_2)$$

Once the polynomial is created, it can be used as a function to estimate the value of any point on the interpolated function line.

Results and Discussion

The table of the divided differences values and coefficients (rounded to four decimal places) as described in the Numerical Solution Section is shown below:

x input	DD 1	DD 2	DD 3	DD 4
1.0	0.0			
		$[y_0, y_1] = 0.4621$		
4.0	1.3863		$[y_0, y_1, y_2] = -0.1969$	
		$[y_1, y_2] = -0.3254$		$[y_0, y_1, y_2, y_3] = 0.1450$
5.0	1.0609		$[y_0, y_1, y_2] = 0.5281$	
		$[y_2, y_3] = 0.7308$		
6.0	1.7918			

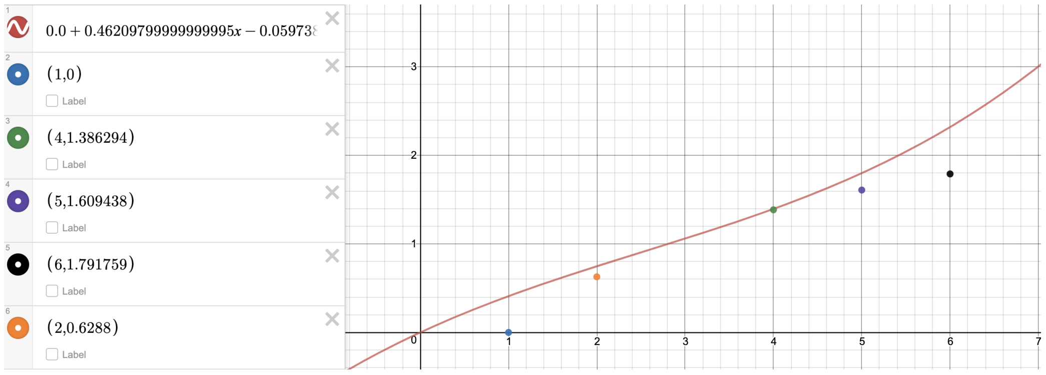
The final polynomial is therefore:

$$f(x) = 0.4102249 + 0.3285789(x-1.0) + 0.0189155(x-1.0)(x-4.0) + 0.0078654(x-1.0)(x-4.0)(x-5.0$$

Simplified:

$$f(x) = 0.0078654x^3 - 0.0597385x^2 + 0.462098x + 0$$

Below is the graph of the interpolated polynomial, the initial input coordinates, and the estimated value for ln(2):



$0.0 + 0.46209799999999995x - 0.05973$

(1,0)

(4,1.386294)

(5,1.609438)

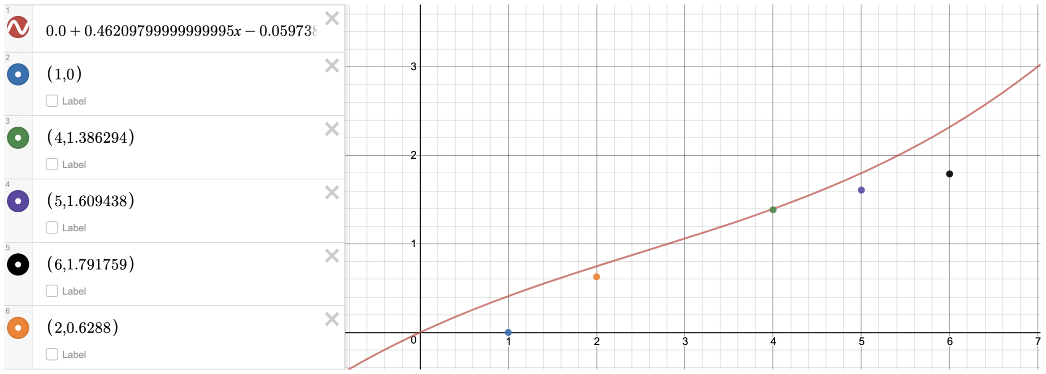
(6,1.791759)

(2,0.6288)

Conclusions

The Newton Divided Differences method is very effective at interpolating functions from a set of points, however the results from this lab may have been much more accurate with the utilization of more input coordinates. Additionally, this method seems to require a degree of precision not always possible with some modern calculators and computers. In conclusion, the Newton Divided Differences Interpolation Method is effective at estimating function. curves, but it requires a significant amount of points to increase its accuracy.

Appendix A



1 $0.0 + 0.46209799999999995x - 0.05973$

2 $(1,0)$

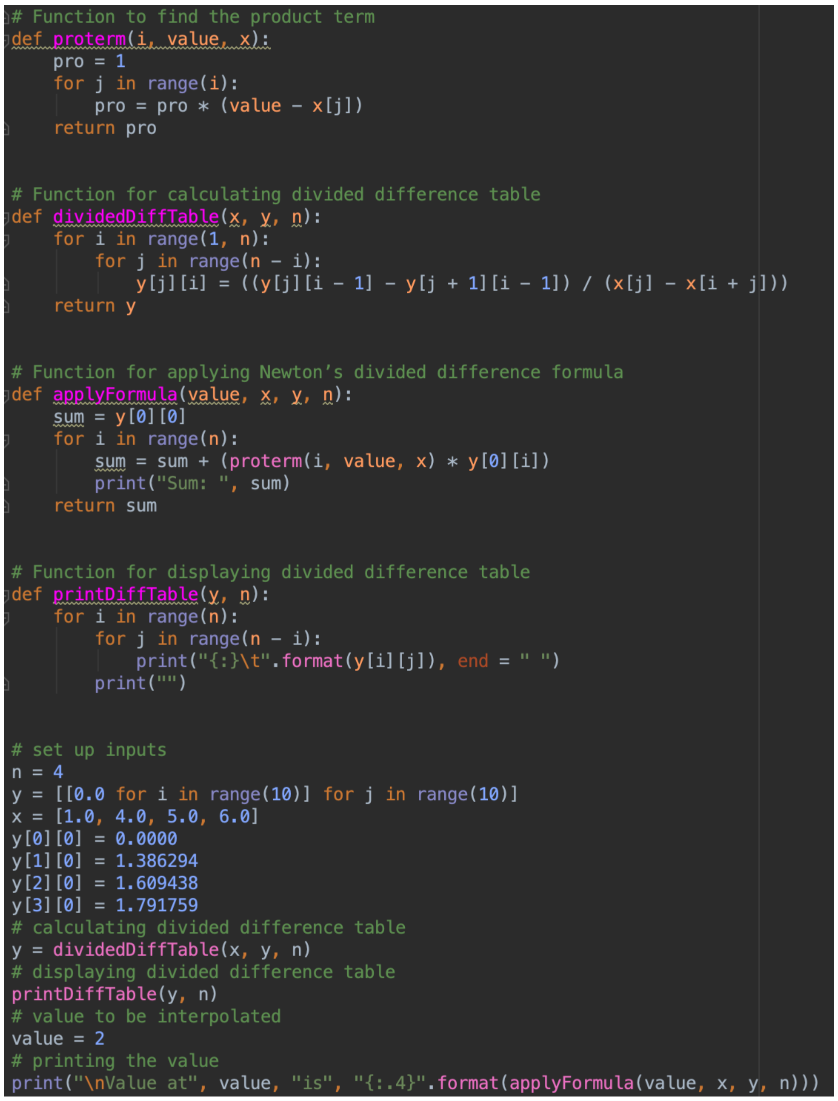
3 $(4,1.386294)$

4 $(5,1.609438)$

5 $(6,1.791759)$

6 $(2,0.6288)$

Appendix B



```python
# Function to find the product term
def proterm(i, value, x):
    pro = 1
    for j in range(i):
        pro = pro * (value - x[j])
    return pro


# Function for calculating divided difference table
def dividedDiffTable(x, y, n):
    for i in range(1, n):
        for j in range(n - i):
            y[j][i] = ((y[j][i - 1] - y[j + 1][i - 1]) / (x[j] - x[i + j]))
    return y


# Function for applying Newton's divided difference formula
def applyFormula(value, x, y, n):
    sum = y[0][0]
    for i in range(n):
        sum = sum + (proterm(i, value, x) * y[0][i])
        print("Sum: ", sum)
    return sum


# Function for displaying divided difference table
def printDiffTable(y, n):
    for i in range(n):
        for j in range(n - i):
            print("{:}\t".format(y[i][j]), end = " ")
        print("")


# set up inputs
n = 4
y = [[0.0 for i in range(10)] for j in range(10)]
x = [1.0, 4.0, 5.0, 6.0]
y[0][0] = 0.0000
y[1][0] = 1.386294
y[2][0] = 1.609438
y[3][0] = 1.791759
# calculating divided difference table
y = dividedDiffTable(x, y, n)
# displaying divided difference table
printDiffTable(y, n)
# value to be interpolated
value = 2
# printing the value
print("\nValue at", value, "is", "{:.4}".format(applyFormula(value, x, y, n)))
```