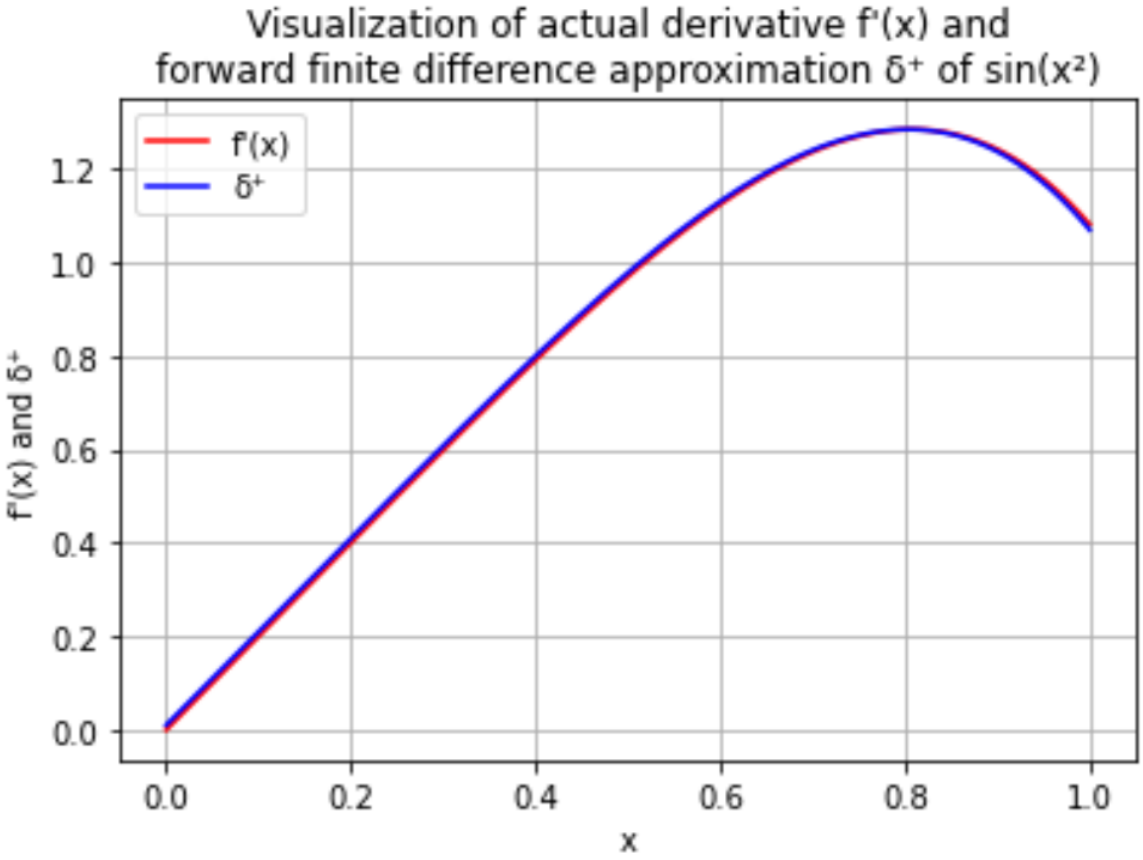
CS5016: Computational Methods and Applications

Lab Report - 4

111901030
Mayank Singla

Q1.

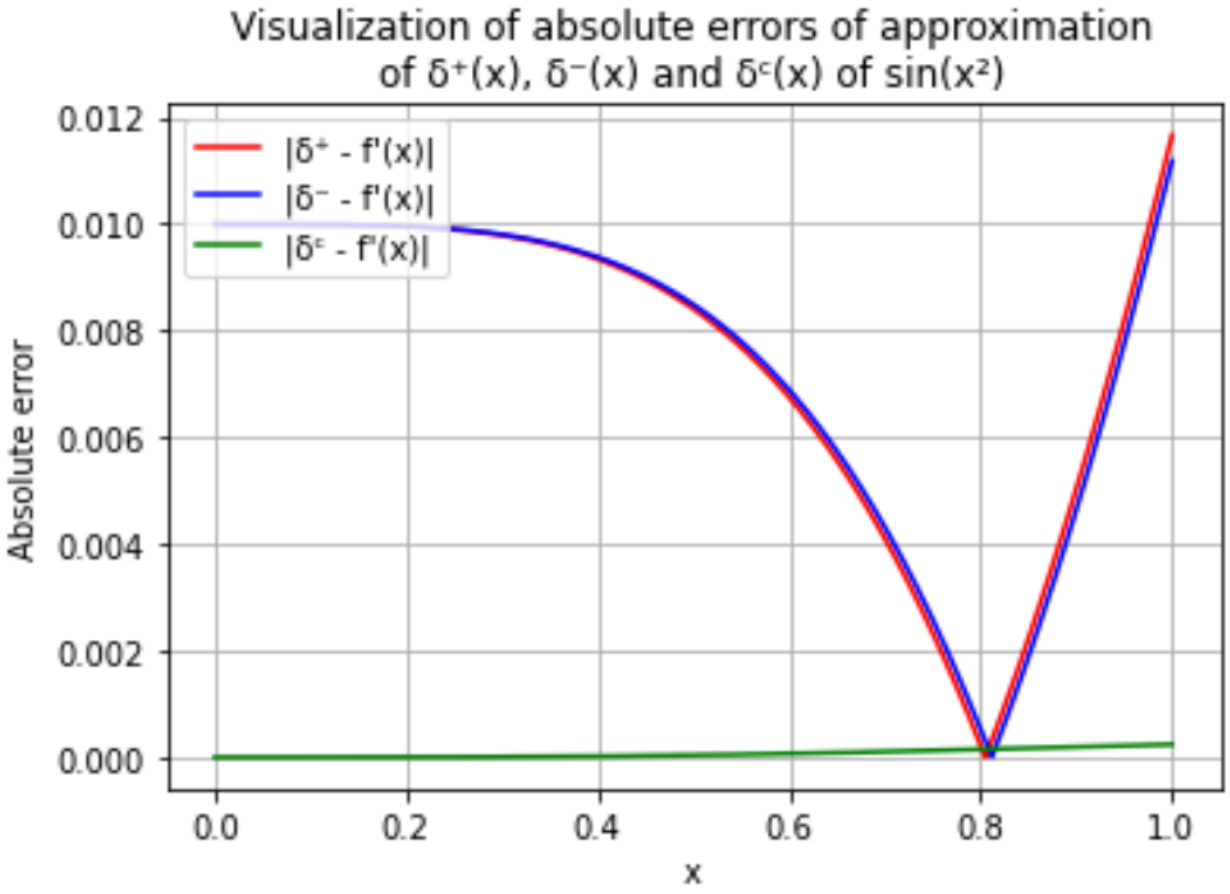
First of all, I am creating the functions `sin_x_2` and `sin_x_2_derivative` that will give me the value and value of the derivative of the given function at any input value x.
Next, I am creating a function `visualize`, that takes a function and its derivative as its argument and it helps in then visualizing the actual derivative and forward finite-difference of the input function. I am creating a nested function `forward_finite_difference` which returns the forward finite difference approximation at any input value using its formula. Then, generating some random number of points in the given interval and plotting both the actual derivative and the forward finite difference.
Here is the result I obtained. Both the curves almost overlap each other.



Q2.

In the same way, as in the previous question, using the functions `sin_x_2` and `sin_x_2_derivative` same as before and creating the nested functions `forward_finite_difference`, `backward_finite_difference`, and `centered_finite_difference` that will compute the corresponding values from the standard formula. Then for the input interval, I am creating some random points and evaluated the absolute errors of approximation for each case, and finally plotted the curve.

Here is the output I obtained.



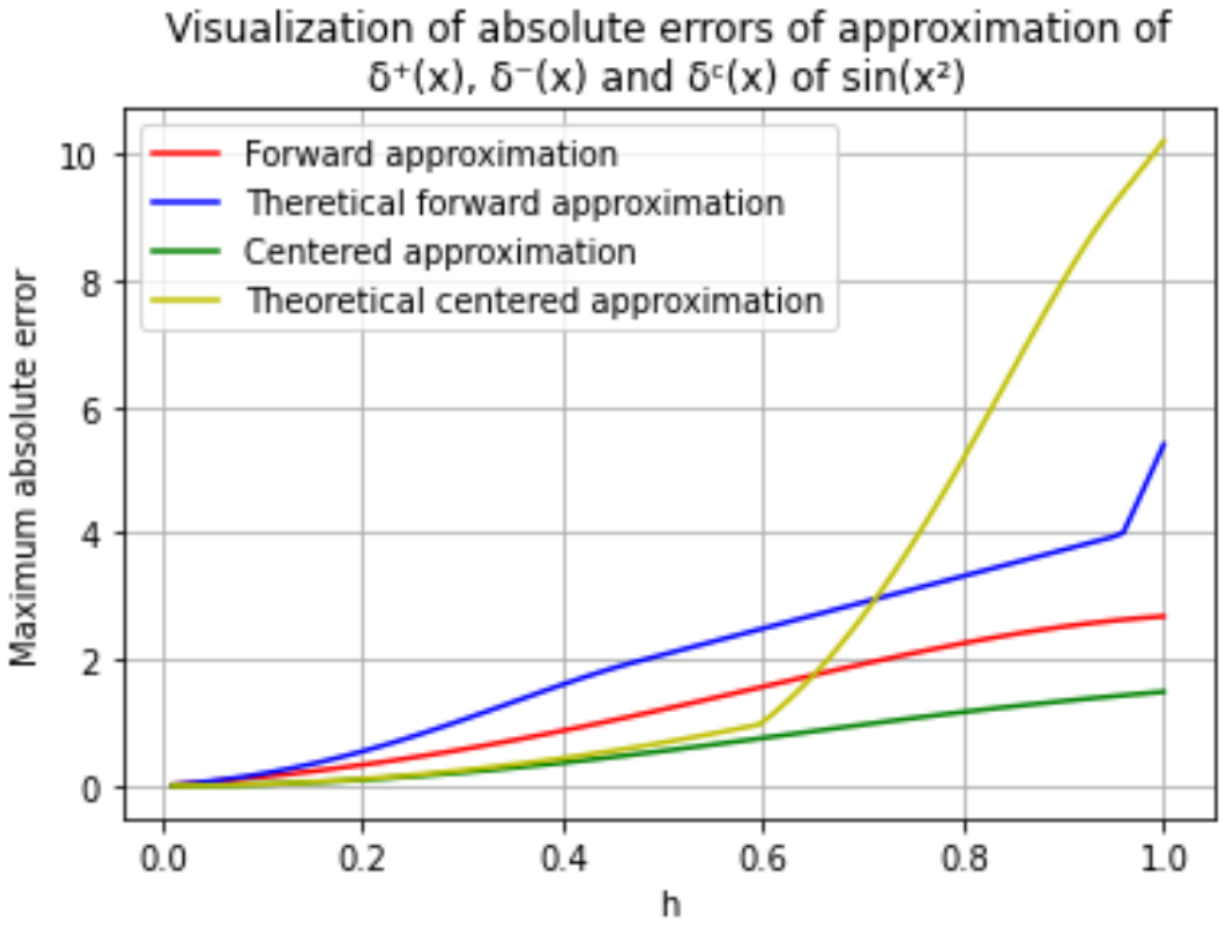
Q3.

n the same way, as in the previous question, using the functions `sin_x_2` and `sin_x_2_derivative` same as before, and also creating the function `sin_x_2_double_derivative` and `sin_x_2_triple_derivative`.

Next, I am creating the function `visualize` that takes these as arguments and helps in visualizing the maximum absolute error of approximation for forward and centered finite difference as a function of `h`. It also plots the theoretical maximum. I am creating nested functions `ffd` and `cfd` to find the forward and centered finite difference at input `x`.

Next, generating some random points for different `h`. Then for each `h`, I am generating some random value of `x` in the input interval. Then computing the maximum value of the error of forward and centered finite difference and for theoretical maximum, again generating some values between `x` and `x + h` to find the maximum value of the corresponding derivative. Storing these values in a list and then finally plotting everything.

Here is the plot I obtained.

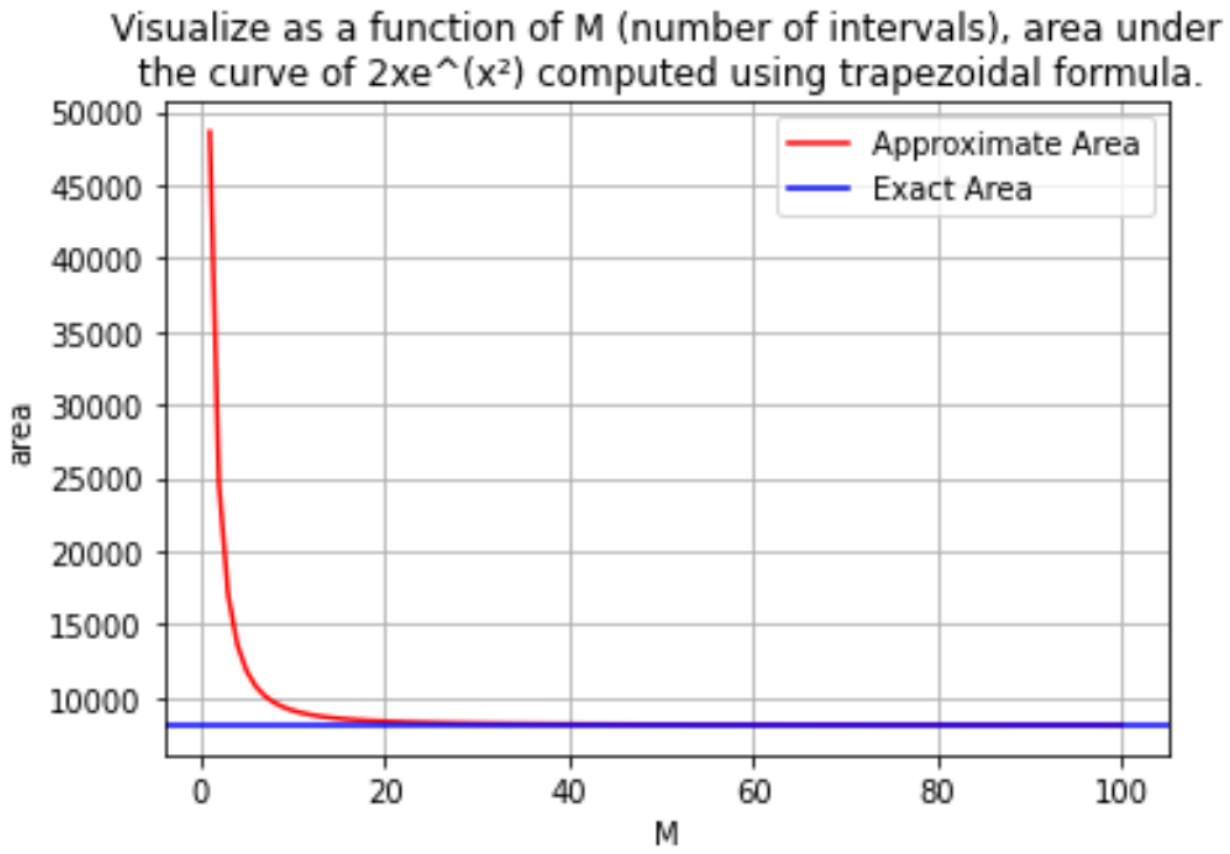


Q4.

I am creating the functions `inp_func` and `inp_func_integral` to evaluate the value and the integral value of the function at input x. Then, I am creating the function `visualize` to visualize as a function of M (number of intervals), the area under the curve of input function computed using trapezoidal formula. It also plots the exact area.

The actual area will be simply the difference in the integral values at the endpoints of the interval. For the trapezoidal area, I am generating some values of M, and for each M, calculating the area under the curve using the trapezoidal formula. Then, I am finally plotting all the values.

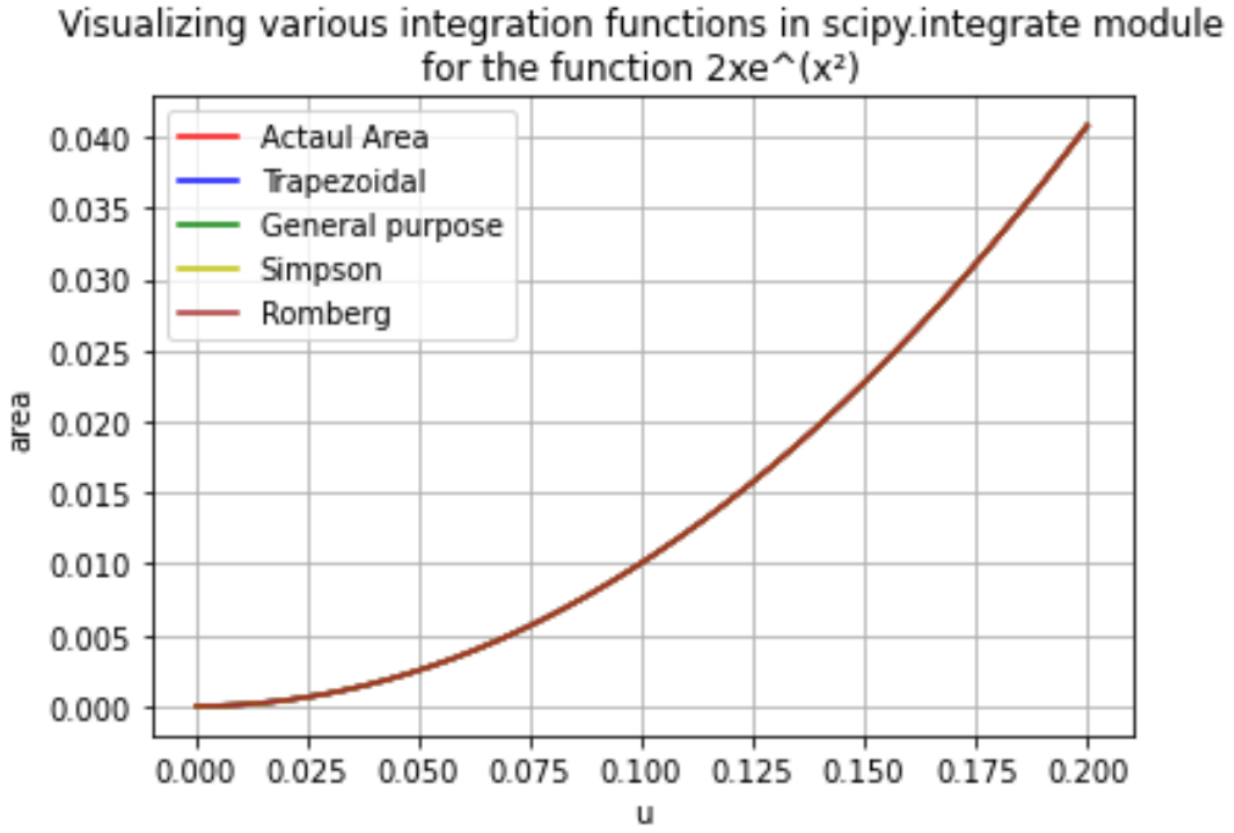
Here is the plot I obtained.



Q5.

Using the same functions `inp_func` and `inp_func_integral` as in the previous case as before. Then generating some values of `u` between a maximum and minimum value and for each `u`, I am storing the area computed using different functions in `scipy.integrate` module. Then finally simply plot all the values to obtain the plot.

Here is the plot I obtained. All the functions almost approximately give the same value and all the curves seems to overlap.



Q6.

Copying the same polynomial class from the previous assignment.

Adding the method `derivative` that computes the derivative of the polynomial and returns a new polynomial.

The derivative of any general polynomial: $a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ will be $a_1 + 2a_2x + 3a_3x^2 + \ldots + na_nx^{n-1}$

Using this general formula to evaluate the derivative of the polynomial.

I am creating a method `integral` that computes the integral of the polynomial and returns the resultant polynomial.

The derivative of any general polynomial: $a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ will be $a_0x + (a_1x^2)/2 + (a_2x^3)/3 + \ldots + (a_nx^{n+1})/(n+1)$

Using this general formula to evaluate the integral of the polynomial.

Next, I am creating the method `area` that returns the area under the polynomial in the input interval. It first checks for the validity of the input. If valid, it finds the integral of the polynomial using the method `integral` and then the area will be the difference in the values of the integral at the two endpoints of the input interval.

Q7.

I am using the Taylor's Series expansion about a point `x = a` to get an approximate polynomial for the given curve $e^x.\sin(x)$

$$f(x) = f(a) + f'(a)(x-a) + f''(a)(x-a)^2/2! + f'''(a)(x-a)^3/3! + \ldots$$

Next, I am putting `a = 0`, to get the **Maclaurin Series** expansion of it. It is a converging series.

$$f(x) = f(0) + f'(0)(x) + f''(0)(x)^2/2! + f'''(0)(x)^3/3! + \ldots$$

$$= x + x^2 + x^3/3 - x^5/30 - x^6/90 - x^7/630 + x^9/22680 + x^{10}/113400 + O(x^{11})$$

As now I have received the coefficients of the polynomial, I am building an object of the `Polynomial` class and using the `area` method to compute its area in the interval.

The absolute error I obtained was: `1.6262047264348212e-11` which is within a guaranteed range of 10^{-6}. I am also computing the actual area using its integral to find this error value.