**Taylor and Maclaurin Polynomials**

**Shubha Swarnim Singh**

**September 6th, 2024**

**Dr. Brandy Wiegers**

**CSC – 455 Numerical Computation**

1. **Introduction**

This report aims to analyze and approximate the function f(x) = ex cos(x) using a 3rd-degree Maclaurin polynomial centered at x0 = 0 and a 3rd-degree Taylor polynomial centered at x0 = 2. The approximation techniques are crucial for understanding how well these polynomials can estimate the function's behavior over a given interval. This process will allow us to compare the accuracy of both approximations at various points and assess the errors associated with the approximations in the interval [0, 2].

This analysis includes constructing the Maclaurin polynomial M3(x), and the Taylor polynomial T3(x), and computing the actual function values at specific points to determine the degree of error.

1. **Maclaurin Polynomial Approximation**

We first construct the 3rd-degree Maclaurin polynomial M3(x) for the function f(x) = ex cos(x), centered at x0 = 0. The general form of the Maclaurin series is:

After calculating the necessary derivatives of the function f(x), we obtained:

* f (0) = e0 cos (0) = 1
* f' (0) = ex (cos (x) – sin (x)) = 1
* f'' (0) = ex (-2 sin (x)) = 0
* f''' (0) = ex (-3 sin(x) – cos(x)) = -2

Thus, the 3rd-degree Maclaurin polynomial becomes:

This polynomial will be tested over a range of values for x and compared to the actual values of the function f(x).

Using M3(x) at x = 0.5:

The true value of f (0.5) is approximately f (0.5) = e0.5 cos (0) ≈ 1

The absolute error is:

**Error Calculation:**

To find an error bound for the 3rd-degree Maclaurin polynomial M3(x) in its approximation of f(x) = ex cos(x) over the interval [0, 2], we will use the remainder term in the Taylor series approximation, also known as the Lagrange remainder. For a 3rd-degree Maclaurin polynomial M3(x), the error R3(x) is given by:

where:

- f(4) (ξ) is the 4th derivative of f(x) = ex cos(x),

- ξ is some point between 0 and x (this value is typically unknown but can be bounded).

For the 3rd-degree polynomial, the 4th derivative of f (x):

For the error bound, we need to estimate f(4) (x) for x in [0, 2]. Since ex cos(x) and ex sin(x) are bounded on this interval, we can use the maximum possible value of | f(4) (x) | in the interval to get an error bound. To estimate the maximum of | f(4) (x) |, we evaluate f(4) (x) at key points x = 0, x = 2, and possibly use numerical methods or graphing tools like Desmos or a Python script to find the maximum value of | f(4) (x) |in the interval.

At x = 0:

At x = 2:

Given that ex grows rapidly, and cos(x) and sin(x) oscillate, f(4) (x) will reach larger values as x increases. We can approximate the maximum value of | f(4) (x) | on [0, 2] using a calculator or Python to find that it's around 60.

Using m = 60 as the maximum value of | f(4) (x) |, we can now compute the error bound for x in [0, 2]:

Let's test x = 1 and compare the actual error with the bound.

* Error bound at x = 1:
* Actual error at x = 1:

We can compute the actual error by finding the difference between the true value of f(1) and the approximation M3(1):

* Actual error:

| f (1) – M3(1) | = | 1.4687 - 2.6667 | 1.198

The actual error of 1.198 is smaller than the error bound of 2.5, as expected. This shows that the Lagrange remainder provides a useful upper bound for the error in the approximation.

1. **Taylor Polynomial Approximation**

Next, we construct the 3rd-degree Taylor polynomial centered at x0 = 2. The general form of the Taylor polynomial is:

To obtain the Taylor polynomial, we calculated the function and its derivatives at x0 = 2:

* f (2) = e2 cos (2) ≈ −3.0751
* f' (2) = e2 (cos (2) – sin (2)) ≈ − 9.791
* f'' (2) = e2 (-2 sin (2)) ≈ −16.507
* f''' (2) = e2 (-3 sin(2) – cos(2)) ≈ −17.085

By substituting these values into the Taylor series, we constructed T3(x), which will also be evaluated at the same points as M3(x).

The 3rd-degree Taylor polynomial is:

Using T3(x) at x = 1:

The true value of f (1) is approximately f (1) = e1 cos (1) ≈ 1.460.

The absolute error is:

**Error Calculation:**

To find an error bound for the 3rd-degree Taylor polynomial T3(x) in its approximation of f(x) = ex cos(x) over the interval [0,2], we will use the remainder term in the Taylor series approximation, also known as the Lagrange remainder. For a 3rd-degree Taylor polynomial T3(x), the error R3(x) is given by:

where:

* f4 (ξ) is the 4th derivative of f(x) = ex cos (x),
* ξ is some point between 0 and x (this value is typically unknown but can be bounded).

We already have:

At x = 0:

At x = 2:

Given that ex grows rapidly, and cos(x) oscillate, f4 (x) will reach larger values as x increases. We can approximate the maximum value of | f(4) (x) | on [0, 2] using a calculator or Python to find that it's around 30.

Using m = 30 as the maximum value of | f(4) (x) |, we can now compute the error bound for x in [0, 2]:

Let's test x = 1 and compare the actual error with the bound.

* Error bound at x = 1:
* Actual error at x = 1:

We can compute the actual error by finding the difference between the true value of f (1) and the approximation T3(1):

* Actual error:

| f (1) – T3(1) | = | 1.4687 – 0.33 | 1.135

The actual error of 1.135 is smaller than the error bound of 1.25, as expected. This shows that the Lagrange remainder provides a useful upper bound for the error in the approximation.

1. **Comparison of Function Values and Polynomial Approximations for f (x)=ex cos (x) at Specified Points**

Here is the table summarizing, the values for f (x), M3(x), and T3(x) at the provided x-values

|  |  |  |  |
| --- | --- | --- | --- |
| x | f (x)=ex cos (x) | M3(x) | T3(x) |
| -1 | 0.198 | 0.333 | -1.3679 |
| 0 | 1 | 1 | -0.6456 |
| 0.5 | 1.447 | 1.485 | 0.5977 |
| 0.75 | 1.549 | 1.609 | 1.0414 |
| 1 | 1.469 | 1.667 | 1.2146 |
| 1.5 | 0.317 | 1.375 | 0.2940 |
| 2 | -3.075 | 0.333 | -3.0749 |
| 15 | -2483432.984 | -1109 | -3934.4418 |

These values match what we received from the computer analysis and mathematical problem-solving.

1. **Predictions about the M3 and T3 approximations**

Based on the analysis, T3(x), the 3rd-degree Taylor polynomial, is centered at some other point (say x = 2). It should give a good approximation near that point. The further away from x = 2, the less accurate the approximation will be. However, M3(x), the 3rd-degree Maclaurin polynomial, is designed to approximate f(x) = cos(x) near x = 0 because it’s centered at x = 0. It should give the best approximation there with low error, but its accuracy decreases as x moves away from 0.

1. **Computer Analysis**

**The python code that I used to approximate the functions is as follows:**

import math

import numpy as np

import matplotlib.pyplot as plt

# define the function f(x) = e^x \* cos(x)

def f(x):

return np.exp(x) \* np.cos(x)

# define the Taylor polynomial T(x) with default Maclaurin expansion

def T(x, x0=0):

if x0 == 0: # Maclaurin polynomial

return 1 + x - (x\*\*3) / 3

else: # Taylor polynomial centered at x0

return -3.07493232064 - 9.79378201807 \* (x - x0) \

- 6.71884969745 \* (x - x0)\*\*2 - 1.2146391256 \* (x - x0)\*\*3

# define the remainder (error estimate) for the Taylor approximation

def R(x, x0=0):

if x0 == 0: # Maclaurin series remainder

return (-4 \* (x\*\*4)) / math.factorial(4)

else: # Taylor series remainder centered at x0

return (12.2997292826 \* (x - x0)\*\*4) / math.factorial(4)

# calculate the actual error between the function and the Taylor approximation

def calc\_error(x, x0=0):

return f(x) - T(x, x0)

# example calculations

x\_values = [1, 2]

# nicely formatted output function

def display\_output(x):

print(f"\n{'='\*40}")

print(f"Results for x = {x}:")

print(f"{'-'\*40}")

print(f"f(x): {f(x):.6f}")

print(f"Maclaurin T(x): {T(x):.6f}")

print(f"Taylor T(x) at x0=2: {T(x, 2):.6f}")

print(f"Maclaurin remainder R(x): {R(x):.6f}")

print(f"Taylor remainder R(x) at x0=2: {R(x, 2):.6f}")

print(f"Maclaurin error: {calc\_error(x):.6f}")

print(f"Taylor error at x0=2: {calc\_error(x, 2):.6f}")

print(f"{'='\*40}\n")

# printing

for x in x\_values:

display\_output(x)

**Output:**

Results for x = 1:

----------------------------------------

f(x): 1.468694

Maclaurin T(x): 1.666667

Taylor T(x) at x0=2: 1.214639

Maclaurin remainder R(x): -0.166667

Taylor remainder R(x) at x0=2: 0.512489

Maclaurin error: -0.197973

Taylor error at x0=2: 0.254055

========================================

Results for x = 2:

----------------------------------------

f(x): -3.074932

Maclaurin T(x): 0.333333

Taylor T(x) at x0=2: -3.074932

Maclaurin remainder R(x): -2.666667

Taylor remainder R(x) at x0=2: 0.000000

Maclaurin error: -3.408266

Taylor error at x0=2: 0.000000

========================================

The output from the computer analysis has matched the report. Hence proving that the computer analysis is right.

**The Three functions (f (x), M3(x), T3(x)):**

# set up x-values from 0 to 2 with steps of 0.1

xs = [x / 10 for x in range(0, 21)]

# compute the values for f(x), Maclaurin T(x), and Taylor T(x) at x0=2

fs = [f(x) for x in xs]

m3s = [T(x) for x in xs] # Maclaurin polynomial values

t3s = [T(x, 2) for x in xs] # Taylor polynomial values at x0 = 2

# calculate the approximate and actual errors for Maclaurin and Taylor series

m3\_approx\_error = [R(x) for x in xs]

m3\_calc\_error = [calc\_error(x) for x in xs]

t3\_approx\_error = [R(x, 2) for x in xs]

t3\_calc\_error = [calc\_error(x, 2) for x in xs]

# plot f(x), Maclaurin M\_3(x), and Taylor T\_3(x) curves

curves = {"f(x)": fs, "Maclaurin M3(x)": m3s, "Taylor T3(x) at x0=2": t3s}

plt.figure(figsize=(8, 6))

for curve\_name, curve\_data in curves.items():

plt.plot(xs, curve\_data, label=curve\_name)

# label the axes and add title and grid

plt.xlabel("X-axis")

plt.ylabel("Y-axis")

plt.title("f(x), Maclaurin M3(x), and Taylor T3(x) curves")

plt.legend()

plt.grid(True)

# display the plot

plt.show()

**A graph of a function

Description automatically generated**

This graph compares f (x), the Maclaurin polynomial M3(x) centered at 0, and the Taylor polynomial T3(x) centered at x3 = 0. It highlights how well each polynomial approximates f (x) near their respective center points.

**Approximated and Calculated error for M3(x) and T3(x):**

# define the data for f(x), Maclaurin approximation error, and calculated error

curves = {

"f(x)": fs,

"Maclaurin M(x) Approx Error": m3\_approx\_error,

"Maclaurin M(x) Calculated Error": m3\_calc\_error

}

# create the plot

plt.figure(figsize=(8, 6))

for curve\_name, curve\_data in curves.items():

plt.plot(xs, curve\_data, label=curve\_name)

# label the axes and set up title, grid, and legend

plt.xlabel("X-axis")

plt.ylabel("Y-axis")

plt.title("f(x), M(x) Approx Error, M(x) Calculated Error")

plt.legend()

plt.grid(True)

# show the plot

plt.show()  
----------------------------------------------------------------------------------------------------------------

# define the data for f(x), Taylor approximation error, and calculated error

curves = {

"f(x)": fs,

"Taylor T(x) Approx Error": t3\_approx\_error,

"Taylor T(x) Calculated Error": t3\_calc\_error

}

# create the plot

plt.figure(figsize=(8, 6))

for curve\_name, curve\_data in curves.items():

plt.plot(xs, curve\_data, label=curve\_name)

# label the axes and set up title, grid, and legend

plt.xlabel("X-axis")

plt.ylabel("Y-axis")

plt.title("f(x), T(x) Approx Error, T(x) Calculated Error")

plt.legend()

plt.grid(True)

# show the plot

plt.show()

**For M3(x)**

**A graph with different colored lines

Description automatically generated**

This graph shows f(x) alongside the theoretical Maclaurin approximation error and the actual calculated error. It visualizes the accuracy of the Maclaurin polynomial approximation, indicating the deviation between the true function and the polynomial around x0 = 0

**For T3(x)**

**A graph with different colored lines

Description automatically generated**

This graph presents f (x) along with the theoretical and calculated errors for the 3rd-degree Taylor polynomial centered at x0 = 2. It emphasizes the approximation accuracy of the Taylor polynomial in predicting f (x) near x = 2.

1. **Conclusion**

To approximate f (x) = ex cos(x), the third-degree Maclaurin polynomial 𝑀3(𝑥) and the third-degree Taylor polynomial 𝑇3(𝑥) were compared in this analysis. When x = 0, the Maclaurin polynomial performed well; however, as x increased, its accuracy declined. Likewise, the Taylor polynomial yielded precise approximations in the vicinity of 𝑥 = 2, but its efficacy decreased in the further distance from its center. Although both polynomials were helpful in the local area, they showed notable errors at higher values of x, highlighting the significance of selecting a suitable center for Taylor series approximations.

The comparison of the 3rd-degree Maclaurin polynomial and the 3rd-degree Taylor polynomial on the interval [0,2] shows that the polynomial with the smallest maximum absolute error provides the best approximation of f(x)=cos(x). M3 is expected to perform better near x = 0, while T3 should provide a closer approximation near x = 2. If T3 demonstrates a smaller maximum absolute error over the entire interval, it indicates that T3 offers a more accurate overall approximation on [0,2], particularly for values of x away from the origin. This result would align with the prediction that T3(x), centered at a point closer to 2, should more closely follow the function as x increases.

1. **References**

* Desmos: <https://www.desmos.com/calculator/qu1ueoxxlr>
* Teatime Numerical Analysis
* LibreTexts Mathematics. <https://math.libretexts.org/Courses/Cosumnes_River_College/Math_401%3A_Calculus_II_-_Integral_Calculus/04%3A_Power_Series/4.03%3A_Taylor_and_Maclaurin_Series>
* ChatGpt