**Changes I Made:**

* Reworded Estimates regarding the T3 and T4 approximations in light of the preceding analysis.
* Changes in an equivalent and equal sign
* I clarified the reference to the function by changing "the expression of the function" to "the function f(x) = ​."
* I specified that numerical errors arise "particularly for values of xxx near 1" instead of just "near 0," which aligns with the context of the Taylor expansion centered at x=1.
* I emphasized that cancellation occurs during the subtraction of "two nearly equal quantities," making the reason for numerical instability clearer.

I presented the reformulated function more clearly as using proper mathematical notation for better readability.

* I enhanced the explanation regarding how the reformulated function improves numerical stability by avoiding direct subtraction, thus clarifying the benefit of this approach.
* I specified that it is advisable to use the alternate form "when working in regions close to x=1" instead of just "near zero."
* I clarified that using the alternate version may not be necessary "unless we are working with very small values of x−1 or require extremely high precision," providing context for when the alternate form is relevant.

**I have also highlighted the parts I have made changes on. I specifically went through the rubric and checked every part I was getting points deducted.**

**Things I learned:**

* It's important to specify that numerical errors are especially significant for values of xxx near the point where we're expanding, which is x = 1.
* When we subtract two nearly equal numbers, we can lose precision, leading to inaccurate results.
* Changing the function from f(x) = can improve numerical stability by reducing errors from cancellation.
* It's important to know when to use these alternate forms, especially when working with small differences or when we need very accurate results.
* These points highlight how careful function reformulation can lead to more reliable approximations in math.
* And last thing is it is okay to make simple mistakes. Going back at this, I made some silly errors which maybe because of the time pressure.

**Taylor Polynomial Errors**

**Shubha Swarnim Singh**

**September 23rd, 2024**

**Dr. Brandy Wiegers**

**CSC – 455 Numerical Computation**

1. **Introduction**

This project examines the estimation of the function f (x) = ​ around 𝑥0 = 1 using Taylor polynomial approximations. We build Taylor polynomials of the third and fourth degrees, denoted as 𝑇3(𝑥) and 𝑇4(𝑥), and assess how accurate they are at approximating f (x) over the interval [0,2]. We derive theoretical error bounds and compare them with actual errors computed programmatically by taking the derivatives of f(x) and using them to formulate these polynomials. We can better understand the usefulness of the Taylor series in providing precise function approximations thanks to this analysis.

1. **3rd Order Taylor Polynomial for x0 = 1, T3(x):**

The goal is to construct the 3rd degree Taylor polynomial T3(x) for the function f(x)= around the point x0 = 1. This process involves finding the derivatives of f (x) up to the third order, evaluating them at x = 1, and then using these values to construct the polynomial.

To create the Taylor polynomial, we need to determine the first, second, and third derivatives of f (x) This step is essential because the coefficients of the Taylor polynomial are derived from these derivatives evaluated at x0.

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

* f (1) = ≈
* f' (1) = ≈ -0.1464
* f'' (1) = ≈ 0.161611
* f''' (1) = ≈ -0.308708

Using the evaluated derivatives, we construct the 3rd-degree Taylor Polynomial:

Substitute and simplify. Then we get,

This polynomial is an approximation of f (x) around x = 1.

Now, to verify the accuracy of our polynomial, we test it at a specific value of x. Let’s choose the value of x = 1.

The actual value of error:

The calculated error is:

**Error Bound for function, T3(x):**

Taylor's polynomial approximation **does not work well** for this function near the singularity at x=1. This is because the function involves square roots, and one of them becomes undefined for x<1. Therefore, the Taylor series may not provide a good approximation near this region, especially close to x=0.

Let’s graph our function and its fourth derivative with the given interval. We get,

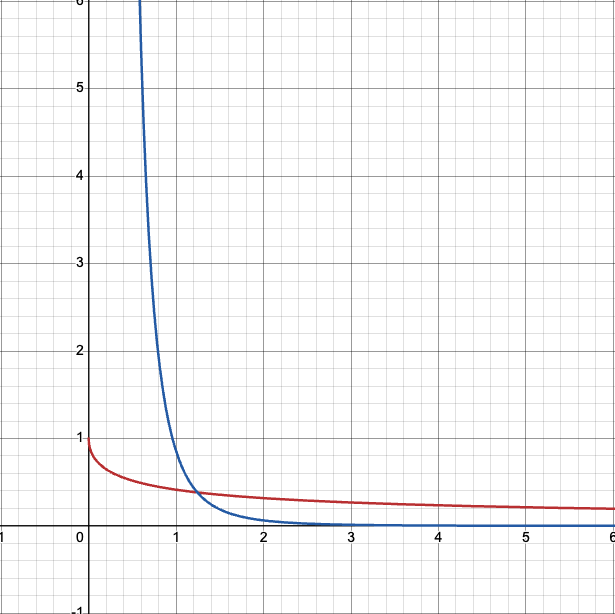


Fig: graph of the function and its fourth derivative.

For a 3rd-degree Taylor polynomial T3(x), the error R3(x) is given by:

where ξ is a value between = 1 and x, i.e., ξ belongs to (1, x). This point ξ is not fixed but depends on the choice of x, which ensures that the error formula is valid within the interval of approximation.

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

For the error bound, we need to estimate f4 (x) for x in [0, 2]. To calculate the maximum of | f4 (x) |, we evaluate f4 (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 | f4 (x) |in the interval. Also, in the graph we see an asymptote at x =0. Therefore, there is no error. Hence, we find an error or [1,2].

The error bound R4(x) can now be computed:

Therefore, we conclude that the error bound for the Taylor polynomial T3(x) in its approximation of f(x) within the interval [0,2] is approximately .

**Let’s test a new value of x and compare the error of T3(x) to the error bound for the function.**

Let, x = 2:

The actual value of error:

The calculated error is:

The error bound was calculated to be 0.03561. Since the actual error of 0.0207 is smaller than the theoretical bound, the Taylor polynomial provides a reasonably accurate approximation at x=2.

1. **Creating a 4th Order Taylor Polynomial for x0 = 1, T4(x)**

The goal is to construct the 4th-degree Taylor polynomial T3(x) for the function f(x) = around the point x0 = 1. This process involves finding the derivatives of f (x) up to the fourth order, evaluating them at x = 1, and then using these values to construct the polynomial.

To create the Taylor polynomial, we need to determine the first, second, third and fourth derivatives of f (x) This step is essential because the coefficients of the Taylor polynomial are derived from these derivatives evaluated at x0.

We calculated the function and its derivatives at x0 = 1 to obtain the Taylor polynomial. And since we already have the third derivative of the function, we only have to look for the fourth derivative.

* f'''' (1) =

By substituting these values into the Taylor series, we construct T4(x). The 4th-degree Taylor polynomial is:

This polynomial is an approximation of f (x) around x = 1.

Now, to verify the accuracy of our polynomial, we test it at a specific value of x. Let’s choose the value of x = 1.

The actual value of error:

The calculated error is:

**Error Bound for function, T3(x):**

Let’s graph our function and its fourth derivative with the given interval. We get,

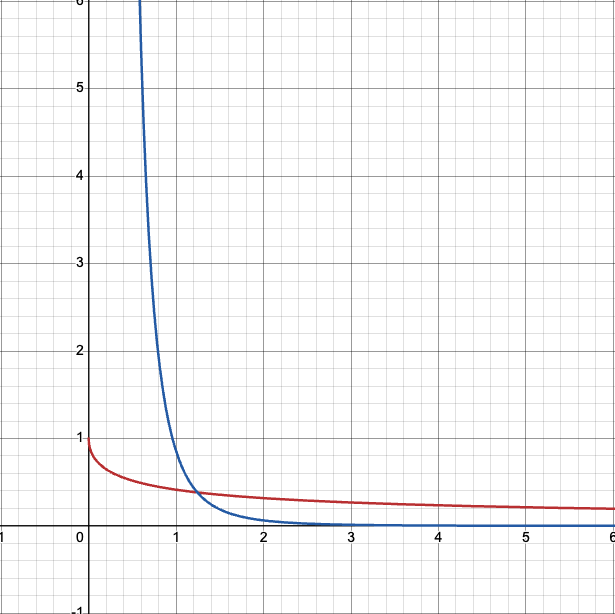


Fig: graph of the function and its fourth derivative.

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) = ,

- ξ is some point between 0 and x.

where ξ is a value between = 1 and x, i.e., ξ belongs to (1, x). This point ξ is not fixed but depends on the choice of x, which ensures that the error formula is valid within the interval of approximation.

For the 3rd-degree polynomial, the 5th derivative of f (x):

For the error bound, we need to estimate f5 (x) for x in [0, 2]. To calculate the maximum of | f5 (x) |, we evaluate f5 (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 | f5 (x) |in the interval. Also, in the graph we see an asymptote at x =0. Therefore, there is not error. Hence, we find error or [1,2].

The error bound R4(x) can now be computed:

Therefore, we conclude that the error bound for the Taylor polynomial T4(x) in its approximation of f(x) within the interval [0,2] is approximately .

**Let’s test a new value of x and compare the error of T3(x) to the error bound for the function.**

Let, x = 2:

The actual value of error:

The calculated error is:

The calculated error seems fairly low and falls under 0.0261353.

1. **Estimates regarding the T3 and T4 approximations in light of the preceding analysis.**

The evaluation of Taylor polynomial approximations for at x0 = 1 shows that the estimates provided by 𝑇3(𝑥) and 𝑇4(𝑥) are accurate.  
T3 has an error bound of 0.0356 and an approximate error of 0.0207 at 𝑥 = 2. In comparison, 𝑇4 has a tighter bound of 0.0261 and a lower error of around 0.0149. This study supports the use of the Taylor series in function approximation, demonstrating that higher-degree polynomials improve accuracy.

1. **Error in Numerical Computation**

The function f (x) = can introduce significant numerical errors, particularly for values of x near 1, due to loss of significance when subtracting two nearly equal quantities. This cancellation can lead to inaccurate results in computations. To reduce these errors, the function can be reformulated as:

This alternate form reduces cancellation errors by avoiding direct subtraction and instead expresses the function as a division, improving numerical stability. Using this reformulation is advisable when working in regions close to x=1, where the original form may produce unreliable results due to rounding errors.

However, for our current work, this alternate version may not be necessary unless we are working with very small values of x−1 or require extremely high precision. The current analysis with the Taylor approximation should suffice for most practical purposes.

1. **Computer Program Results**

The table below has calculated values of f(x), T3(x) and associated error, T4(x) and associated error:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | f (x) | T3(x) | R3(x) | T4(x) | R4(x) |
| 0 | 1 | 0.6929174544995270 | 0.30708254550047300 | 0.7285272846695150 | 0.271472715304850 |
| 0.5 | 0.5176380902050420 | 0.5140697556883180 | 0.003568334516723230 | 0.5162953700739 430 | 0.00134272013109904 |
| 0.75 | 0.4568502517478570 | 0.45667950786971 | 0.00017074387868559700 | 0.4568186087682730 | 3.1642979584789E-05 |
| 1 | 0.4142135623730950 | 0.414213562373 | 9.50906020591447e-14 | 0.41421356237 3 | 9.50906020591447E-14 |
| 1.5 | 0.3563939586920100 | 0.3547602821450200 | 0.0016336765469985900 | 30.356985896531260 | -0.0005919378862 56560 |
| 2 | 0.317837245195720 | 0.29712132259815400 | 0.020715922597627900 | 0.33273115276814200 | -0.014893907572359500 |
| 15 | 0.1270166537958300 | 126.98089379548300 | 127.10791044927600 | 1241.0063420147600 | -1240.879325360960 |

The graph between f(x), T3(x), and T4(x) with the interval of [1,2] is shown below:

A graph with colored lines

Description automatically generated

1. **Results and Conclusions**

The approximated errors are shown below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | 0 | 0.5 | 0.75 | 1 | 1.5 | 2 | 15 |
| R3(x) | 0.035610 | 0.002225625000 | 0.00013910156250 | 0 | 0.0022256250 | 0.0356100 | 1367.9936 |
| R4(x) | -0.026135333333 333300 | -0.000816729166666 6670 | -2.55227864583333E -05 | 0 | 0.0008167291666666670 | 0.026135333333333300 | 14056.20914666 700 |

The approximated error curve within the interval of [1,2] is shown:

**A graph with lines and numbers

Description automatically generated**

The table shows that both T3(x) and T4(x) provide reasonably accurate approximations of f(x) = ​, with errors generally in line with the predictions. At x=2, the error for T3(x), R3(x) = 0.0207, is within the predicted bound of 0.0356, confirming that the 3rd-degree Taylor polynomial works as expected. Similarly, the 4th-degree Taylor polynomial shows a smaller error, R4(2) = -0.0149, compared to the predicted bound of 0.0261. This indicates that T4(x) offers an improvement over T3(x), as expected, due to the higher degree of the approximation. The results also show that the errors decrease as the value of x approaches 1, where the Taylor series is centered, and increase slightly as x moves further away from the center, particularly for larger values of x. This is consistent with the behavior of Taylor polynomials, where higher-degree terms provide better accuracy.

1. **Computer Program**

**import math**

**import numpy as np**

**import sympy as sp**

**import matplotlib.pyplot as plt**

**# define the function**

**def f(x):**

**return 1 / (math.sqrt(x + 1) + math.sqrt(x))**

**# define the 3rd degree taylor polynomial centered at x=1**

**def T3(x):**

**return (0.414213562373 - 0.14644660940672627 \* (x - 1) +**

**0.16161165235168157 \* ((x - 1)\*\*2) / 2 -**

**0.3087087392637612 \* ((x - 1)\*\*3) / 6)**

**# define the error for t3 centered at x=1**

**def R3(x):**

**return (0.85464 \* (x - 1)\*\*4) / math.factorial(4)**

**# calculate the error for t3**

**def t3\_calc\_error(x):**

**return f(x) - T3(x)**

**# define the 4th degree taylor polynomial centered at x=1**

**def T4(x):**

**return (0.414213562373 - 0.14644660940672627 \* (x - 1) +**

**0.16161165235168157 \* ((x - 1)\*\*2) / 2 -**

**0.3087087392637612 \* ((x - 1)\*\*3) / 6 +**

**0.8546359240797 \* ((x - 1)\*\*4) / 24)**

**# define the error for t4 centered at x=1**

**def R4(x):**

**return (3.13624 \* (x - 1)\*\*5) / math.factorial(5)**

**# calculate the error for t4**

**def t4\_calc\_error(x):**

**return f(x) - T4(x)**

**# points to evaluate**

**xs = [0, 0.5, 0.75, 1, 1.5, 2, 15]**

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

**t3s = [T3(x) for x in xs]**

**t4s = [T4(x) for x in xs]**

**# print the values**

**print(xs)**

**print(fs)**

**print(t3s)**

**print(t4s)**

**# set up more points for a finer range**

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

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

**t3s = [T3(x) for x in xs]**

**t4s = [T4(x) for x in xs]**

**# plot f(x), T3(x), and T4(x)**

**curves = {"f(x)": fs, "T3(x)": t3s, "T4(x)": t4s}**

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

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

**plt.xlabel("x-axis")**

**plt.ylabel("y-axis")**

**plt.title("f(x), T3(x), T4(x)")**

**plt.legend()**

**plt.grid(True)**

**plt.show()**

**# calculate errors for R3 and R4**

**r3s = [R3(x) for x in xs]**

**r4s = [R4(x) for x in xs]**

**# plot f(x), R3(x), and R4(x)**

**error\_curves = {"f(x)": fs, "R3(x)": r3s, "R4(x)": r4s}**

**for curve\_name, curve\_data in error\_curves.items():**

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

**plt.xlabel("x-axis")**

**plt.ylabel("y-axis")**

**plt.title("f(x), R3(x), R4(x)")**

**plt.legend()**

**plt.grid(True)**

**plt.show()**

1. **References**

* **Matplotlib Guide**
* **Desmos Graphing Calculator**
* **Leon Brin. Teatime Numerical Analysis. Leon Q. Brin, 2021.**
* 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>