Tu-An Nguyen
tunhnguy@ucsc.edu
01/24/2021

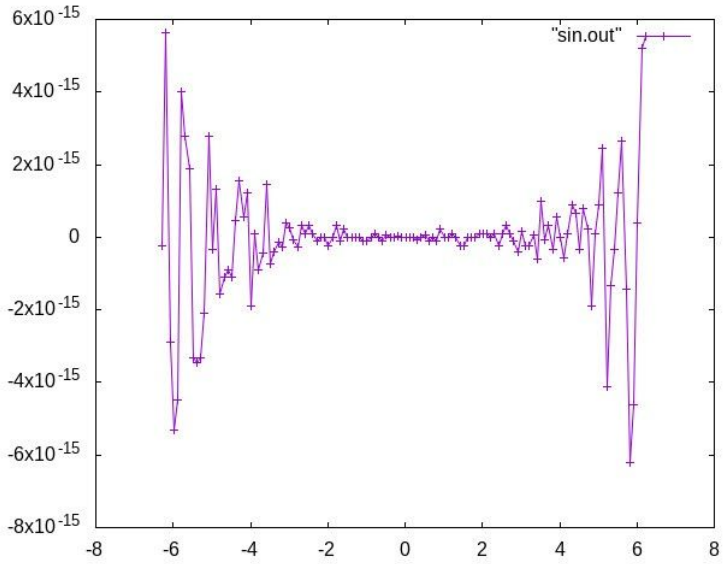
CSE 13S Spring 2020
Assignment 2: A Small Numerical Library
Writeup Document

Differences

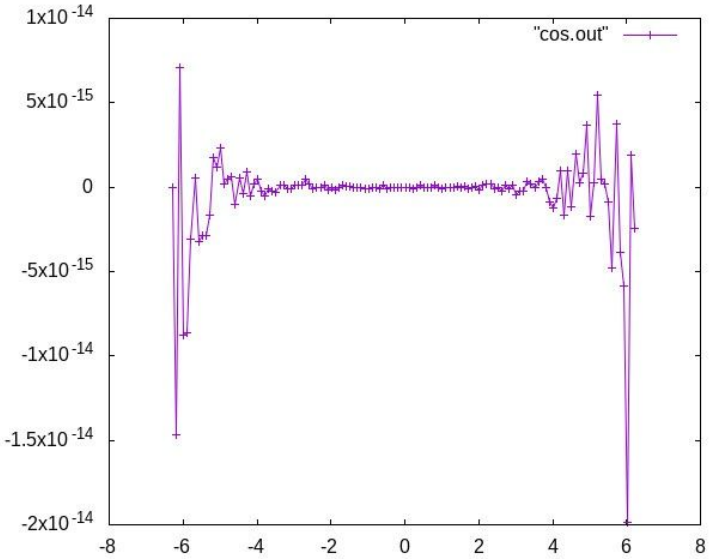
There are a few differences in output between my program and the `<math.h>` library. First, there is the approximation in my program. For this assignment, I set the epsilon value to `1e-14`. This means after a term in the Taylor series of a function falls below `1e-14`, the program stops. I don't know how the `<math.h>` library accounts for error, but I doubt it is exactly the same as how I programmed it. However, there are no differences in the terminal output when running the program. This is because we only output a certain number of digits. I believe if we extend the number of digits outputted, we can eventually see a difference between my program and the `<math.h>` library. I graphed the difference between my program and the `<math.h>` library for the sine, cosine, tangent, and exponent functions after extending the number of significant digits in the "Difference" column.

Graphs

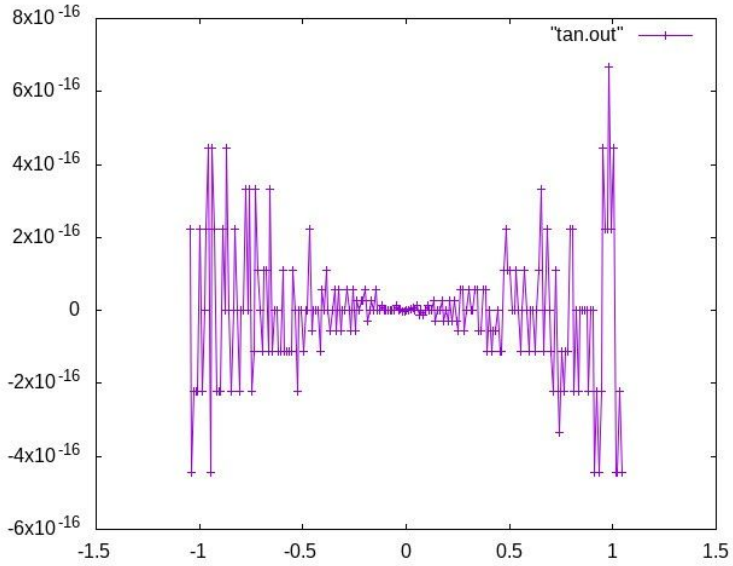
Sine



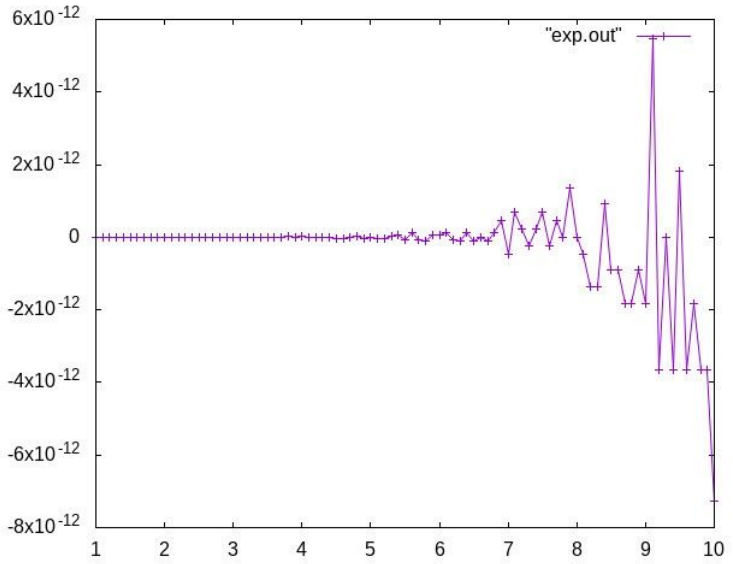
Cosine



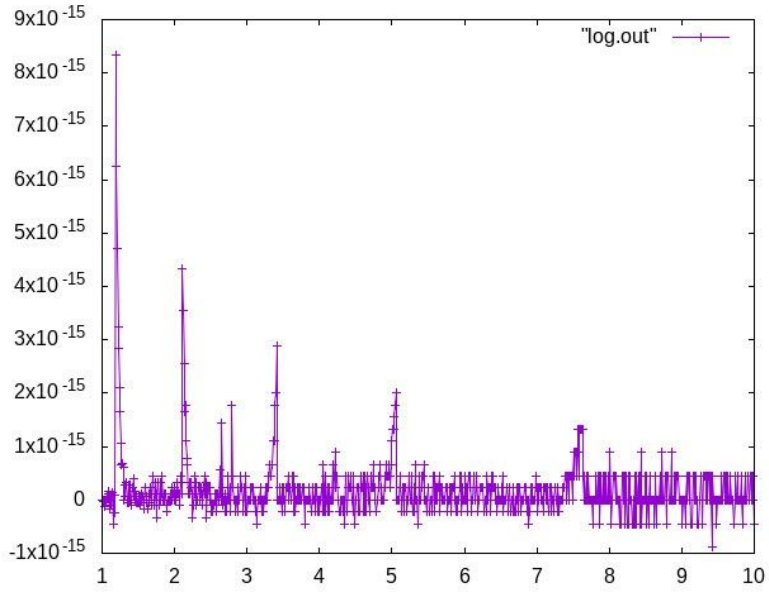
Tangent



Exponent



Logarithm



Conclusion

All in all, my reasoning aligns with the results in the graphs. The differences between my program and the C library can be clearly shown if I increase the number of significant digits. What is interesting about the graphs is that we can visualize at which inputs the differences grow.

For the sine and cosine functions, the differences diverge as the input strays further away from 0. This is the same result for the tangent function, as it is just a ratio of the previous two, but it seems to diverge sooner. However, I had to reduce the interval of the input to make the trend more noticeable. Lastly, the exponent function seems to diverge greatly with inputs larger than 7. Lastly, the logarithm function is most interesting as the trend isn't as clear. If I were to guess, it seems like the difference diverges as the input gets closer to 0. However, even if the input was closer to 10, the difference is still oscillating. I suspect this is because we used Newton's method to implement the logarithm function instead of calculating the Taylor series.