Liam Murray

Lijamurr

CSE13s

Assignment 2 Write Up

Results Overview

Overall, the self implemented versions of most functions performed adequately when compared to the math.h library of functions. Naturally, there were a few, minute differences, likely due to rounding and truncation errors, however the self implemented functions typically worked to within 1e-10 of the library functions. There were a few notable exceptions, such as the arcsine and log functions, which seemed to have poorer performance. However, there were no functions that had significant inaccuracy in their calculations.

Sine Function

The self implemented sine function was very accurate. It had a total of 11 tests that detected a difference between the library and homemade function. In every case, the actual number outputted by the two functions was exactly the same. The difference detected was always in the range of 1e-10 to 1e-11. This means that the error was likely due to rounding inaccuracies. Given that it was around the smallest possible representable number by long double, it seems likely that the difference was actually below that threshold, and was thus rounded up to a number that could appear within the representable scale. Another factor is that the difference was below the cutoff value set for differences in added values (represented as “Epsilon”, which was set to 1e-10). This indicates that if that value was decreased, the self implemented function could achieve greater accuracy.

Cosine Function

The cosine implementation I used was directly derived from the Sine function, so it falls victim to the same issues the sine function did. It was very accurate, however there were small differences with some tests. These were also likely due to rounding and scale issues, and could be combated by decreasing Epsilon even further. Many of the same intervals where the sine function detected error, the cosine function did as well. This indicates that inaccuracy in the sine function likely translated to the cosine function.

Arcsine Function

The arcsine function detected zero difference between the library and the self made function in every instance except for one. The first test value, -1, showed a difference of around 0.00000029. This is surprising, especially given that this value is well above the difference cutoff threshold in the function definition. So, what could have caused this? Well, -1 is the very edge of this function's definition. Especially given that it did not struggle with any other value, it's likely that this edge case caused an issue in either calculation process or the algorithm itself. However, the difference is small enough to say that the function is generally working properly.

Arccosine Function

Very similar to the cosine function, the implementation of Arccosine is based on the Arcsine function. For this reason, it performed very similarly. It had no error across the board, except for when testing -1, which had an error value of 0.00000029. Given that these functions share both a similar implementation and the exact same error factor, it seems safe to say that the issue in the arccosine function stems directly from the issue in the arcsine function.

Arctangent function

Much like arccosine, the arctangent function is implemented as a manipulation of the arcsine function. However, it does not share its pitfalls. The arctangent tests came back with a much smaller error factor, with every difference value appearing below 1e-10. This means that the issues in arctangent are likely very similar to those in sine and cosine. Things such as rounding, truncation, and a limited difference threshold (Epsilon) all could contribute to the minute differences detected. The arctangent function also makes use of the exponent subfunction, which is also an approximation of its library counterpart. This means that inaccuracies in that function could also contribute to issues in arctangent. Overall, it performs very well.

Log function

The log function has the greatest consistent error delta of all the functions. Most of the values tested produce a small difference, usually under 1e-10. This has a few different explanations. The first, is that the log function uses a different computation method than the other functions. Log() uses the Newton-Raphson method, which could be less consistently accurate than the taylor series used in the other functions. Another explanation is that the e^x function, which is heavily used throughout the log implementation, has significant error in its calculations as well. However, what is more likely, is that some or all of these factors compound with things like rounding and truncation errors to create the differences shown in the tests. Overall, the function has some flaws, but is still sufficiently accurate.