Assignment 2 - WRITEUP.pdf

Victor Nguyen

January 20, 2022

1 Introduction:

This writeup will be analyzing math functions and their relative errors. We will be focusing on comparing and contrasting a college student (me) implementations of some math functions versus what is believed to be a reliable math library for the c programming language. Some of these functions include:

- $\sqrt{1-x^4}$
- $cos(x^2)$
- $\frac{e^{-x}}{x}$

The main focus of this analysis is to understand why we can't represent all real numbers through the usage of floating point numbers. Note: analysis is based on the assumption that the students' functions work as intended.

2 Plots and Analysis:

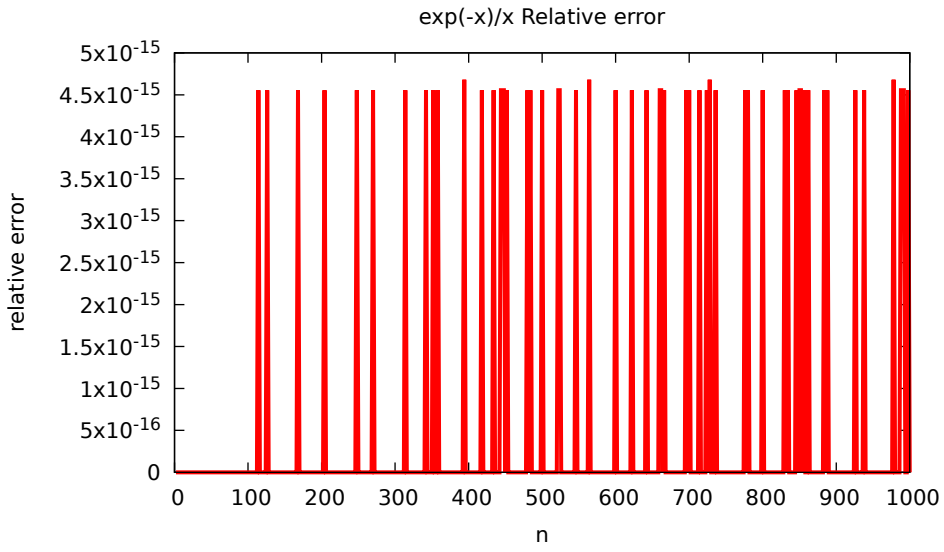

Figure 1: exp(-x)/x Relative error graph

This graph is interesting because we have zero errors for the first 100 n partitons. This is a really good sign, but our code starts showing "small" differences when we use partitions above 100. Looking at the graph, the trend for errors seem to occur more frequently as we test higher partition numbers. We can tell by looking at the spaces between the red lines and how those gaps gets smaller as we use higher n values. Does this trend seem to follow through? We can see that in the next graph.

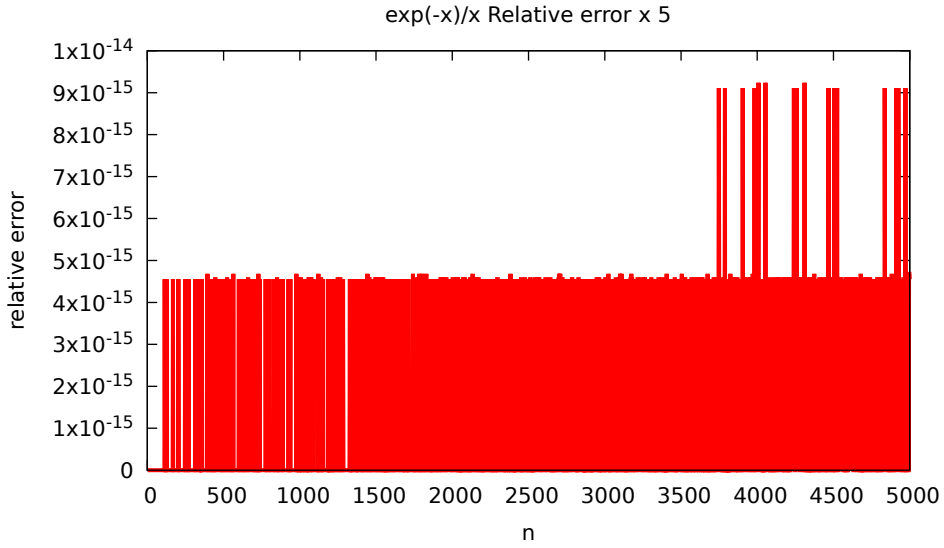

Figure 2: exp(-x)/x Relative error graph x 5

Well, looks like it occurs more frequently, and what's even worse is that the error gets bigger with the higher number partitions. You can probably imagine what an even higher partition and it's error could look like. At this point, some may suggest that we should just use a set amount of partitions (i.e. just use a partition number that is less than 100, and all is good right?). This is something worth trying, but it doesn't apply to all functions. For example, lets look at integrating log(log(x)).

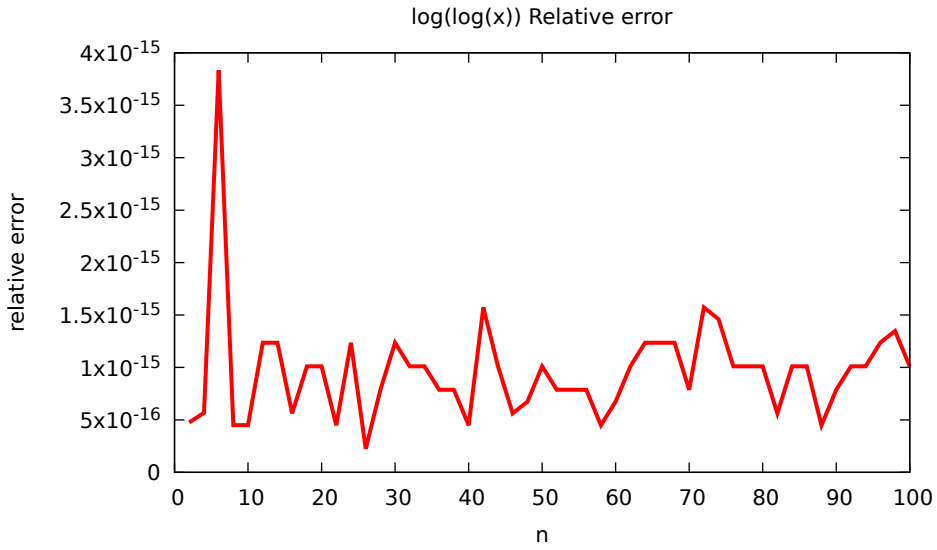

Figure 3: log(log(x)) Relative error graph

Looking at this graph, even with smaller n partitions, we still get errors. In fact, for this function, the error is quite frequent and doesn't even have a partition number where the error value is zero (using partition numbers 2 - 100). These trends seem to also persist in the other math functions that I tested so I didn't include any graphs for those. So, it's not a matter of using a specific number of partitions, there is another underlying reason. That reason is that we can't represent all real numbers using floating point numbers. We can only ever approximate and get close enough to a specific value in the real numbers. So what, we could also look at the numbers and realize that the numbers have such minuscule error values, thus we can just disregard it and use these functions, right? It really depends on what we are doing. Let's look at the $cos(x^2)$ function so that I can point something else out.

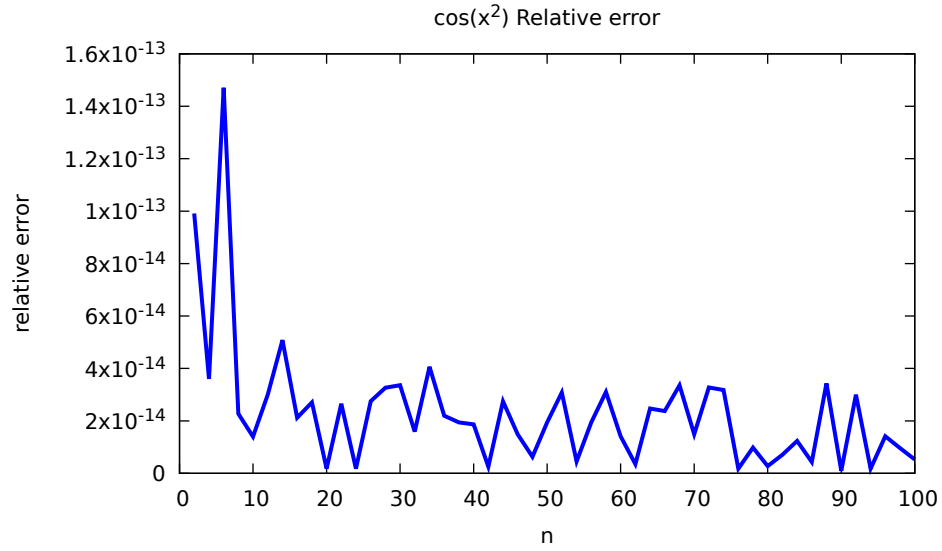

Figure 4: $cos(x^2)$ Relative error graph

Looking at this graph, we can notice that it's similar to the previous graph (refer to previous figure) in the way that it's error starts around the same number of partitions with one key difference. My $cos(x^2)$ function has an error that is a power higher than compared to the previous figure. These differences and accuracies can be important for certain projects such as self driving cars, surgical machines, and much more. We shouldn't ignore these errors for the above stated reasons.

3 Conclusion/What I learned:

1. We can't represent all real numbers using floating point numbers. We can only approximate to the real numbers.

2. Increasing our partiton count doesn't always result in higher precision. We need to test and search for the best number possible for our given problems.

3. To increase the accuracy in our calculations for any project, always aim to get numbers below a certain epsilon. Find out the accuracy needed to make any project a success and calculate numbers to be below this certain epsilon threshold.

4. Don't be ignorant with the code we write, as it could mean the life or death of someone. I certainly wouldn't want to be responsible for someone's death due to an oversight in the code I wrote for self driving cars.