ASGN 2: Writeup

The final differences between my program and math.h for pi and e were:

e()=0.000000000000000, terms=17
pi_euler()=0.000000095493891, terms=10000000
pi_bbp()=0.000000000000000, terms=11
pi_madhava()=0.000000000000007, terms=27
pi_viete()=0.000000000000004, terms=24

As seen both pi_bbp and e() were able to correctly calculate their number with 100% accuracy up to 15 decimal places. Coincidentally those two were also the two fastest programs requiring 11 and 17 terms respectivelly. Of these 5 functions we can see that while 4 of them seem to be within the same ballpark as each other with an accuracy of 14-15 decimal places and less than 30 terms each. The 5th, pi_euler only had an accuracy of 7 decimal places and necessitated 10 million terms to calculate the required number.

The answer to why can be seen rather quickly within the function for pi_euler. Unlike each of the other presented functions the calculation per loop features an ever growing denominator of k^2 and a static numerator of 1. This means that in order to even get close to the value of 3, without even caring for accuracy, thousands of terms have to be calculated. Then of course once it approaches 3.14 it gets exponentially smaller thus requiring exponentially more terms.

The final differences between my program and math.h for the square root was:

For the first term(0): diff=0.000000000000007, terms=47
Terms .1 through 4.3: diff=0.000000000000000, terms=6
Remaining terms: diff=0.000000000000000, terms=7

As seen every term outside of 0 was able to calculate with an accuracy of 15 decimals in 6-7 terms. The first term however yielded an accuracy of 14 decimals and required 47 terms. The reason for the difference in both accuracy and number of terms likely stems from