Homework 9 or something like that, quadratures

I do believe I have done all parts.

As always the output from the tests are in the OutA.txt and so on files in the folders. Part A als contains an image with the error function estimate

PART A

Output OutA.txt, I print the results of various integrals, and check against the analytical solutions. erf.png, I print the error function calculated.

Part A is easy, I calculate the integrals asked, and check that they are within error of the true result. I also use quad to estimate the error function as asked for.

PART B

Output OutB.txt, I print the results of various integrals, using IntegralCC and the old Integral from before, I already tested that the result is correct in A, so this time I don't test errors. Here, however, I also print the number of steps. In the same file, I also print the result from the python version

In Part B, I do implement a function IntegralCC which uses Clenshaw–Curtis variable transformation variable transformation, I print how many calls to the functions are being made, and the "improved" version gets away with many fewer calls … but there is a catch.

The variable transformation is somewhat unstable, and liable to get stuck in an infinite loop of recursions (where the tolerance decreases faster than the error, eventually leading to an actual stack-overflow). To be fair, this only happens if I use a very tight tolerance (relative and absolute accuracy = 1e-8 or better). The not-improved version do work at these precisions, without getting stuck, and for this reason, I honestly prefer the version I implemented in part A. Sure, the non-improved version might make a few thousand calls where the improved version makes a few dozen, but at least I have never seen it get stuck.

I did also compare against the scipy package quadrature, which made much fewer calls than the not variable-shifted version, but returned much worse results, I suspect the scipy version does not scale the absolute accuracy with 1/sqrt(2) as you suggested.

PART C

Output OutC.txt, I print the results of various integrals, I print the result, what we should have gotten, and whether it technically passes. In the same file, I also print the result from the python version

In part C I include the error estimate (assuming unrelated errors) and allow infinite limit, in the C# version I test 1/xˆ3, 1/xˆ4 integrated from 1 to infinity and the gaussian integral exp(-xˆ2) from - to + infinity, and ffrom -infinity to 0 just to test all the limits. The results are **ALMOST** all correct, notably 1/xˆ3 gives the exact result instantly, while the remaining examples require a couple hundred iterations (at this high precision at least)

NOTE, one of the integral, the gaussian from - to + infinity does technically **FAIL**, but it is exceedingly close to the calculated error. I do not think this should disqualify the entire part, or even the entire homework, but you decide that

I test against Pythons scipy package: I only test 1/xˆ4 from 0 to infinity and exp(-xˆ2) from - to + infinity, the result is, needless to say, correct. The number of iterations is much smaller, and the error is smaller as well. Oh well, the goal was not to beat scipy.