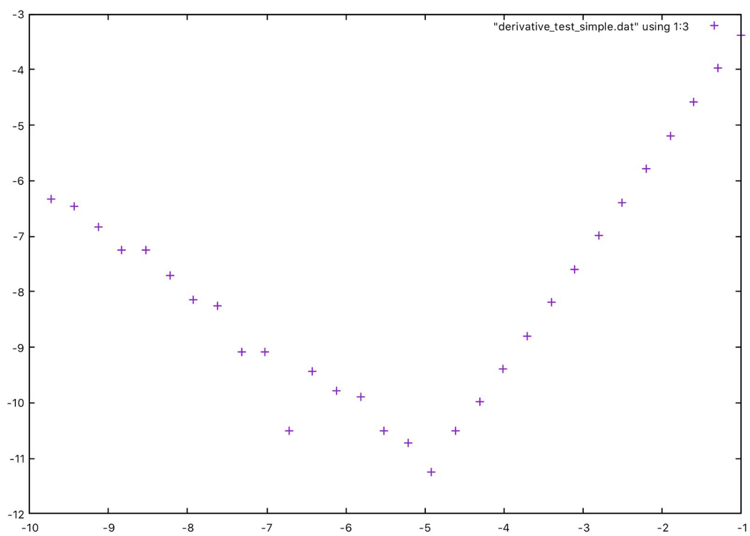
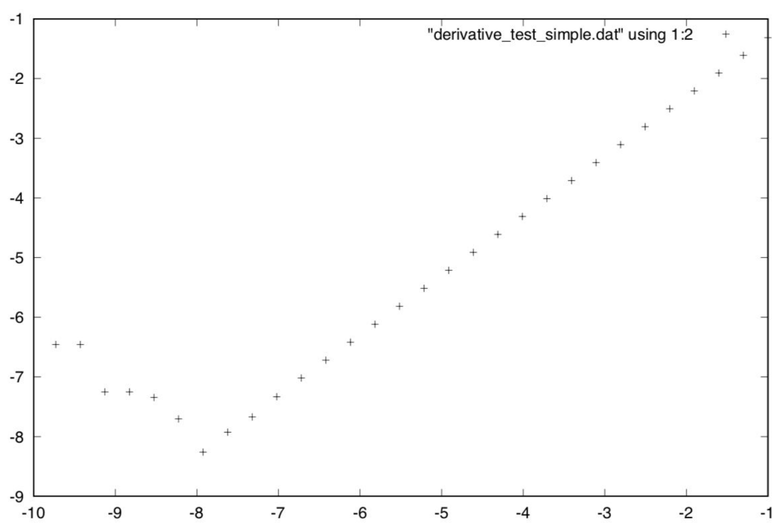
1) This program compares an approximate 1. Value to an explicit 1. It uses cos(pi/4.)/sin(pi/4.) with a pi value that has a precision of around 15. The values are then compared to se if they are considered equal.
2) We do not get the answer weee expect because of the approximated pi value. I added another cout with a setprecision and prints out x1 and x2 to the command prompt.
3) I would suggest to use a predefined higher set precision of pi, and lower the precision of the out come so the decimal point values that are so small they dont matter would be rounded out.

Numerical Derivatives

1) The code doesnt print anything to the command prompt but it does print a file called derivative_test_simple.dat, which stores all of the log(h) | rel fd | rel cd.



This is the central diff error vs log h



This the forward diff error vs log h

4) The slopes in the are consistent with our notes. The positive sections are 1 and 2 for the central and forward respectively. The analysis did mention we should have a slope close to 2 for larger h-values. So, this makes me believe the central diff is the better algorithm.

5) forward diff: Optimal H values from the notes should be around 10^-8. We find that when our h value is around that optimal value thee outcome is around the approximate error for the central diff.

central diff: Optimal H values from the notes should be around 10^-5. This agrees with the approx error around 10^-4 that is on the graph.

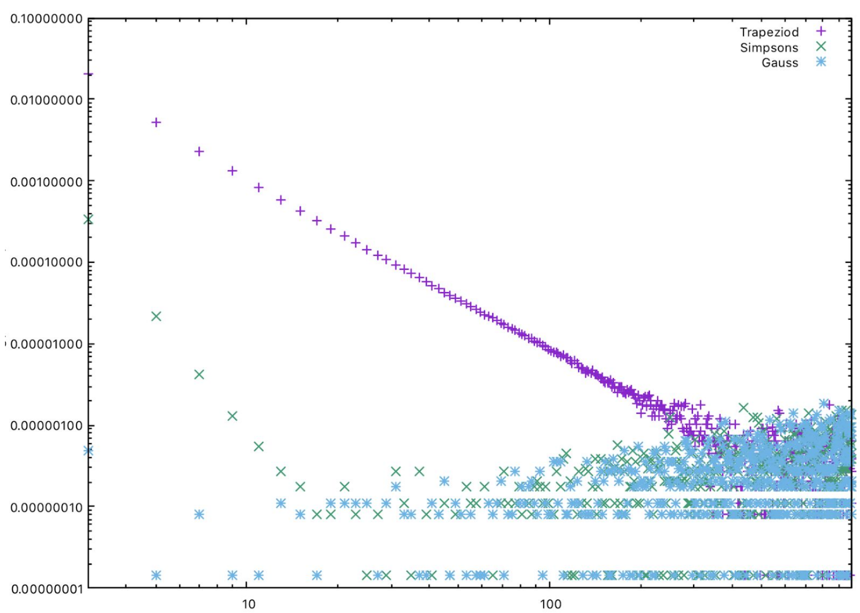
Which makes sense since H should be bigger for the central diff

6) The slopes should change, and the central diff should return a larger optimal H value.

Makefiles for mult.

1) Yes. Using multiple files is important to practice to keep the code easy to follow and read in chunks. This also helps with debugging, you do not have to run the whole script to test a modular section like a separate alg in its own file. In large scripts this is key, it saves computational resources running smaller pieces.
2) Yes the output file makes sense and is logically structured.

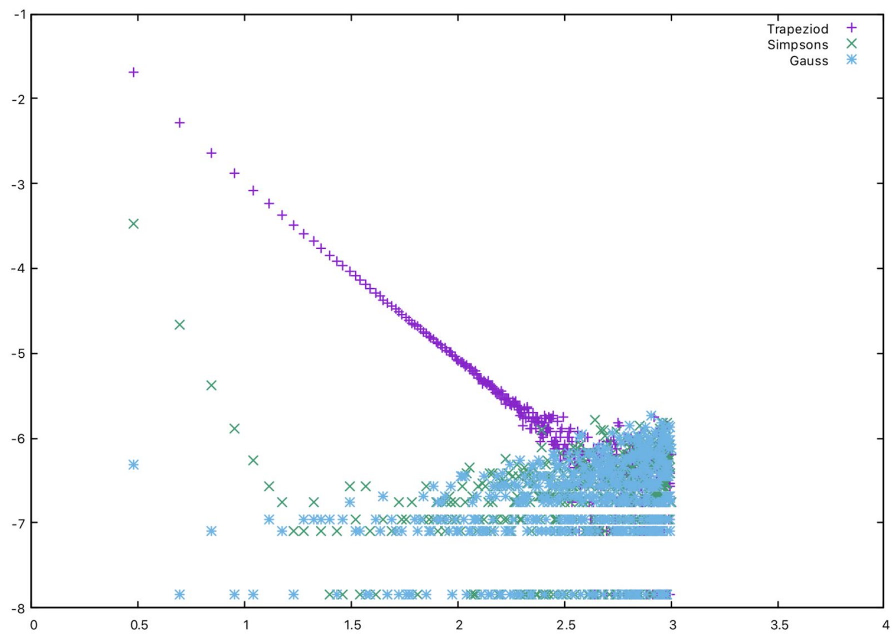
4)



Unevenly spaced

There are different regions where the slopes are different. You can clearly see for the trapezoidal section there is a distribution for the approximation that is linear and negative in slope. Which makes sense since we will get more precise outputs with larger values. That is until it becomes too large and we have issues with the round-off error.

5) To get an evenly spaced I removed the log scale and used log10 for each of the outputs. Which resulted in the following graph.



My group and I had issues plotting anything above 10^3.

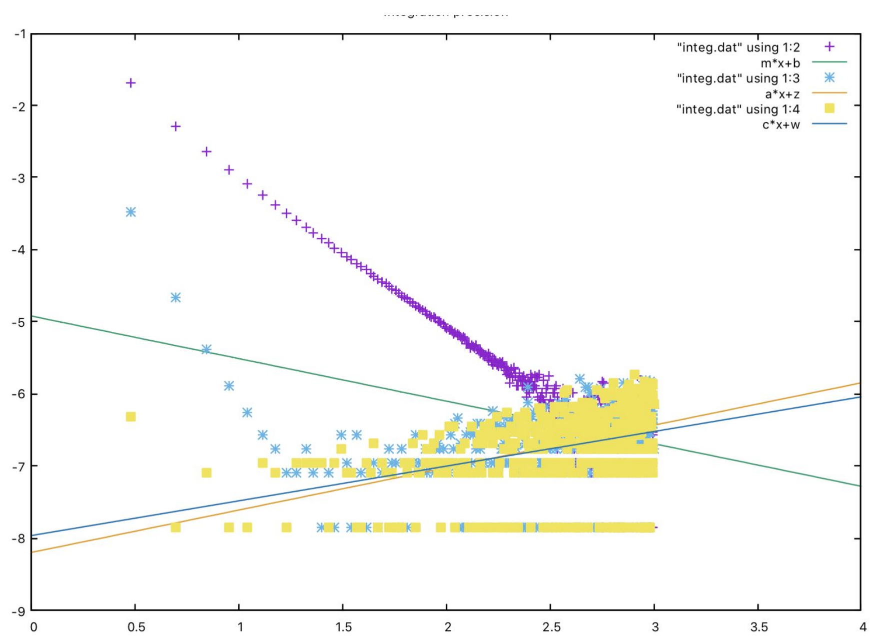
Finding approximation error

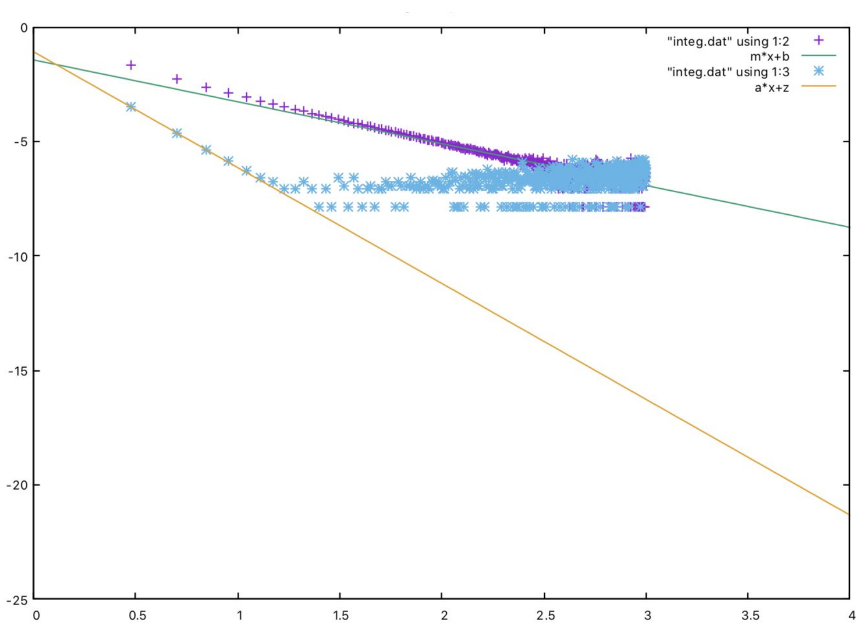
2) Trapezoidal linear slope was around -1.83 and the simpsons linear slope was around -5.06

3) The slopes do make sense with what we have learned in the notes. For trapezoidal it slopes as 1/N^2 and simpsons is 1/N^4. Which agrees with simpsons steeper slope.

   Fits:

   First is the fits for the evenly spaced followed by the fits of just simpsons and trap.


"integ.dat" using 1:2
m*x+b
"integ.dat" using 1:3
a*x+z
"integ.dat" using 1:4
c*x+w
-1
-2
-3
-4
-5
-6
-7
-8
-9
0
0.5
1
1.5
2
2.5
3
3.5
4


"integ.dat" using 1:2
m*x+b
"integ.dat" using 1:3
a*x+z
0
-5
-10
-15
-20
-25
0
0.5
1
1.5
2
2.5
3
3.5
4