FCM 1 HW 4

Ryan Bausback

November 2020

1 Executive Summary

Our goal is to use two different numerical quadrature methods, composite trapezoidal rule and composite midpoint rule, to estimate the values of five different test integrals. Global mesh refinement will be employed in order to reduce computations in determining the accuracy of the estimated integrals value for different sizes of intervals. Our findings indicate that composite midpoint rule is more efficient than composite trapezoidal rule for error threshold less than 10^{-12}, as well as confirming a priori error bound predictions of the size of each subinterval H_m in most cases.

2 Statement of Problem

Some integrals are nearly impossible to compute analytically, especially when dealing with differential equations, and that is why it is important to be able to compute them numerically using quadrature methods. Two simple quadrature methods will be implemented: composite trapezoidal rule and composite midpoint rule. Once it has been determined that the codes are working properly on two simple problems, they will be used on five more difficult examples. The results of quadrature will be compared to the exact analytical solutions and errors computed. Global mesh refinement will be employed as well in order to allow for complete reuse of previous function evaluations in the computation of same integrals with smaller local intervals (H_m v. H_{3m}). The error from the true integral value will be estimated by comparing the coarse mesh result to the fine mesh result, which will in turn be compared to the true error.

Once these bounds are computed, the a priori error bounds will be analytically derived and the interval sizes to get to a desired error will be computed. This will then be empirically validated by the global mesh refinement algorithm, which will compare the computed error against a threshold. Mesh refinement will continue until the error is less than the threshold and then the interval size reported, which we predict will be very similar to the analytical estimates. Comparison of the number of function evaluations to achieve these error thresholds will also take place.

3 Description of Mathematics

3.1 Composite Trapezoidal Rule

The composite trapezoidal rule is defined in the following way:

$$I_m^{ctr} = \sum_{i=1}^{m} \left(\int_{a_i}^{bi} p_{1,i}(x)dx \right) = \sum_{i=1}^{m} \left(\int_{a_i}^{bi} l_{i-1}^{(1)} f(a_i) + l_i^{(1)} f(b_i)dx \right)$$

$$= \sum_{i=1}^{m} \frac{h_{tr}}{2}(f(a_i) + f(b_i)) = \frac{H_m}{2} \left[f(x_0) + f(x_m) + \sum_{i=1}^{m-1} f(x_i) \right]$$

where $l_i^{(1)}$ is the lagrange basis since $p_{1,i}(x)$ is a local piecewise linear polynomial. [1] $H_m = \frac{b-a}{m}$ and $x_i = a + H_m$ for $0 \leq i \leq m$.

It can also be shown that for the global mesh refinement, to halve the size of each local interval (i.e. $H_{2m} = \frac{H_m}{2}$) while reusing all of the previous function evaluations, the formula for the trapezoidal rule becomes:

$$I_{2m}^{ctr} = \frac{1}{2} \left[I_m^{ctr} + H_m \sum_{m \ newpts} f(x_i) \right]$$

as seen on slides 22 and 23 of set 25.[2] These two quadrature methods can then be used together in order to estimate the true error from the exact integral. In order to find this method, we must first use knowledge of the general a priori error bound for the composite trapezoidal rule. In general it is given as $E = Kh^r f^{(m+1)}(\eta)$, and specifically for the composite trapezoidal rule as $E_m^{ctr} = I(f) - I_m^{ctr}(f) = -(b-a)H_m^2 f'' + O(H_m^3)$. This will be used to estimate the interval size needed to achieve certain levels of error analytically later on. If we assume that in this case the max norm of the second derivative doesn't change when changing the interval sizes, we can drop it and are left with $E_c = Ch^r$, with the half size interval error as $E_f = C(\frac{h}{2})^r$. Then, if we assume that the true integral $I(f) = I_c + E_c = I_f + E_f$, then $0 = (I_c - I_f) + (E_c - E_f)$ and we can solve for either error to get the following forms:

$$E_m^{ctr} = \frac{2^r}{2^r - 1}(I_{2m}^{ctr} - I_m^{ctr})$$

$$E_{2m}^{ctr} = \frac{1}{2^r - 1}(I_{2m}^{ctr} - I_m^{ctr})$$

These will be used to estimate the error and will compared against the true error later on.

3.2 Composite Midpoint Rule

Composite midpoint rule is defined in the following way:

$$I_m^{cmp} = \sum_{i=1}^{m} H_m f(x_i), \quad x_i = \frac{(b_i + a_i)}{2}$$

and the a priori error expression is given as:

$$E_m^{cmp} = I(f) - I_m^{cmp}(f) = (b-a)\frac{H_m^2}{24} f'' + O(H_m^3)$$

as is similar to the composite trapezoidal rule. [1] Using an identical method, we can also get the error estimates from the global mesh refinement algorithm by subsituting 3 in for 2 in the above expressions, giving us:

$$E_m^{ctr} = \frac{3^r}{3^r - 1}(I_{3m}^{cmp} - I_m^{cmp})$$

$$E_{3m}^{ctr} = \frac{1}{3^r - 1}(I_{3m}^{cmp} - I_m^{cmp})$$

The reason we would want to use an estimate on I_{3m}^{cmp} in the first place is because thirding the interval gives us complete reuse of previous function evaluations in an almost identical way to the composite trapezoidal rule above. For instance, if we want to estimate $I_9 = h_9(f_1+f_2+f_3+f_4+f_5+f_6+f_7+f_8+f_9)$, the composite midpoint rule with 9 subintervals, we can use the following in efficient forms on the same functions to define the same interval: $I_1 = h_1(f_5)$ since f_5 is the original midpoint on the interval and $I_3 = h_3(f_2 + f_5 + f_8)$ as f_2 and f_8 are the midpoints of the intervals on either side of f_5.

However, since we known the relationship between $h_{3i} = \frac{h_i}{3}$, we can define I_3 in terms of I_1 and I_9 in terms of I_3:

$$I_3 = h_3 f_2 + h_3 f_8 + \frac{I_1}{3}$$

$$I_9 = h_9(f_1 + f_2 + f_4 + f_6 + f_7 + f_9) + \frac{I_3}{3}$$

The same can be seen in an even case with $I_6 = h_6(f_1+f_2+f_3+f_4+f_5+f_6)$ since $I_2 = h_2(f_2 + f_5)$. Therefore, in general, the global mesh refinement with complete reuse is given by

$$I_{3m}^{cmp} = \frac{1}{3} I_{m}^{cmp} + H_{3m} \sum_{2mnewpts} f(x_i)$$

Since we are splitting each interval into thirds, we also know that the relationship between each of the nodes in new $3m$ mesh is plus or minus $\frac{H_m}{3}$ of the old nodes from the m mesh. This will be validated in the experiments.

4 Algorithm and Implementation

The implementation of the algorithm is very straight forward, since the design of the global refinement algorithm reduces the complexity of additional function evaluations to begin with to $O(m)$. Specifically, for composite trapezoidal rule, there are m+1 function evaluations and for composite midpoint rule, there are m function evaluations, which are done in for-loops respectively. These are then passed to methods that use for-loops to perform the summations described above in Section 3 within the respective formulas. During the mesh refinement, there are only 2m+1 (or 3m) function evaluations for a particular level, meaning that the additional function evaluations needed are m for composite trapezoidal and 2m for composite midpoint, all done in separate for-loops. Additional methods then perform the global mesh refinement formulas in a similar manner to the initial quadrature. To simplify the program's architecture, a while loop iterates through more and more mesh refinements until the estimated error is less than a specified threshold, at which point the respective programs terminate and output the results to file.

5 Experimental Design and Results

In order to first check that the algorithms were performing accurately, the function $y = 2x$ was integrated on $[-1, 1]$, which should give an exact answer of 0. This was confirmed to be the case for both quadrature methods on the very first iteration on a single interval.

The next function used to confirm the accuracy was $y = x^2$ on the same interval of $[-1, 1]$. The expected integral value was $\frac{2}{3}$, which was easy to quickly verify in decimal as 0.6666666... . As is visible in Figure 1, this was achieved, with a estimated errors of $-4.96705E-9$ and $3.87176E-9$ for composite trapezoidal and composite midpoint respectively, with final interval sizes H_m of $2.44140625E-4$ and $4.57247E-4$ (threshold in this case was 10^{-8}).

An additional test was run to see if the algorithms could accurately reproduce an example integration of e^x on [0,1] (set 25, slide 30). [2] This was also confirmed to be the case, as Figure 2 illustrates.

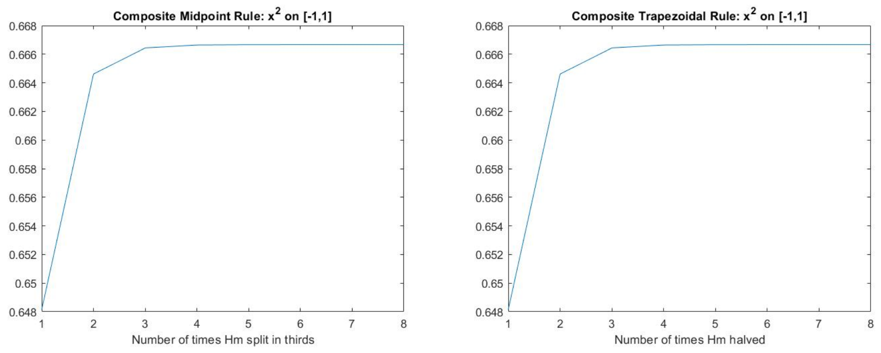
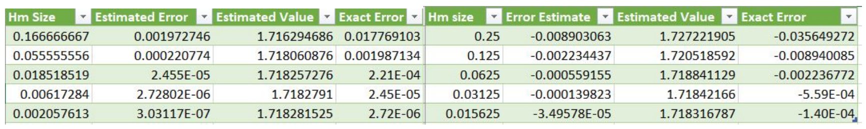


Figure 1: Numerical Quadrature Estimates of x^2 on [-1,1]



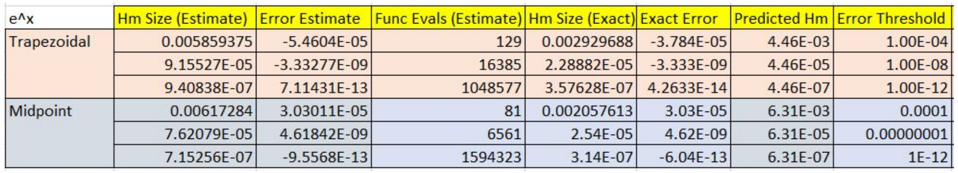
Hm Size	Estimated Error	Estimated Value	Exact Error	Hm size	Error Estimate	Estimated Value	Exact Error
0.166666667	0.001972746	1.716294686	0.017769103	0.25	-0.008903063	1.727221905	-0.035649272
0.055555556	0.000220774	1.718060876	0.001987134	0.125	-0.002234437	1.720518592	-0.008940085
0.018518519	2.455E-05	1.718257276	2.21E-04	0.0625	-0.000559155	1.718841129	-0.002236772
0.00617284	2.72802E-06	1.7182791	2.45E-05	0.03125	-0.000139823	1.71842166	-5.59E-04
0.002057613	3.03117E-07	1.718281525	2.72E-06	0.015625	-3.49578E-05	1.718316787	-1.40E-04

Figure 2: Numerical Quadrature of e^x on [0,1] using Composite Midpoint Rule (left) and Composite Trapezoidal Rule(Right)

5.1 $\int_0^3 e^x dx$

The first integral to be investigated was $\int_0^3 e^x dx = e^3 - 1 \approx 19.085553692$. For this function and all subsequent functions, three different error thresholds will be investigated: $10^{-4}, 10^{-8}$, and 10^{-12}. When computing the a priori error bound, the max norm, $||f''||_\infty$, on [0,3] was easily observed to be e^3, resulting in a predicted subinterval size of $H_m \leq \sqrt{4/e^3 * 10^{-k}}$ for composite trapezoidal rule, where k is the threshold of the estimated error. For composite trapezoidal rule, the H_m estimates were $4.46*10^{-3}$, $4.46*10^{-5}$, and $4.46*10^{-7}$ respectively.

For composite midpoint rule, the predicted subinterval size was $H_m \leq \sqrt{8/e^3 * 10^{-k}}$, and specifically $4.46 * 10^{-3}$, $4.46 * 10^{-5}$, and $4.46 * 10^{-7}$ respectively.



e^x	Hm Size (Estimate)	Error Estimate	Func Evals (Estimate)	Hm Size (Exact)	Exact Error	Predicted Hm	Error Threshold
Trapezoidal	0.005859375	-5.4604E-05	129	0.002929688	-3.784E-05	4.46E-03	1.00E-04
	9.15527E-05	-3.33277E-09	16385	2.28882E-05	-3.333E-09	4.46E-05	1.00E-08
	9.40838E-07	7.11431E-13	1048577	3.57628E-07	4.2633E-14	4.46E-07	1.00E-12
Midpoint	0.00617284	3.03011E-05	81	0.002057613	3.03E-05	6.31E-03	0.0001
	7.62079E-05	4.61842E-09	6561	2.54E-05	4.62E-09	6.31E-05	0.00000001
	7.15256E-07	-9.5568E-13	1594323	3.14E-07	-6.04E-13	6.31E-07	1E-12

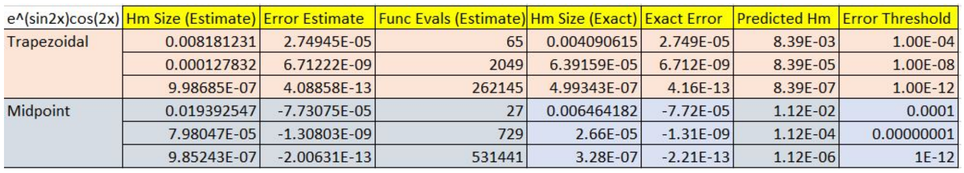
Figure 3: Numerical Quadrature of e^x on [0,3]

Figure 3 shows the computed exact error and the error estimates for particular sizes of H_m. As the third and fourth columns show, for the first instance of the exact error being less than each threshold, the exact size of H_m is less than the a priori estimate, thereby reinforcing theory. For the estimates of the error, the H_m size is of the same order predicted size but is slightly larger, do to inexact nature of the estimate.

As discussed in section 4, there are m additional function evaluations required each time the subintervals are split in half in the composite trapezoidal rule, while there are $2m$ additional function evaluations each time the intervals are split into thirds. This would seem to indicate that the composite midpoint rule is less efficient at producing results less than a particular error threshold. However, we also must consider the number of times that the global meshes were refined in order to total up the overall number of function evaluations to calculate efficiency. To achieve the error thresholds above for composite trapezoidal rule, the mesh was refined 7 times to achieve 10^{-4} while only 4 times for the same threshold for midpoint. This means that there were $2^7 + 1 = 129$ function evaluations for composite trapezoidal, while only $3^4 = 81$ for composite midpoint. This indicates that while there are definitely more evaluations per each refinement for composite midpoint, it can potentially produce more accurate results cheaper. One counterexample is that for the 10^{-12} threshold, there were $3^{13} = 1594323$ function evaluations for midpoint while only $2^{20} + 1 = 1048577$ for trapezoidal, meaning that composite midpoint may not perform better when the threshold for error is very fine.

5.2 $\int_0^{\pi/3} e^{sin(2x)} cos(2x) dx$

The second integral to be evaluated was $\int_0^{\pi/3} e^{\sin(2x)} \cos(2x) dx \approx 0.688721337618$. When computing the a priori error bound, the max norm, $||f''||_\infty$, on $[0, \pi/3]$ was found to be ≈ 16.283 after computing the second derivative as $e^{\sin(2x)}(-12\cos(2x)\sin(2x) + 4\cos^3(2x) - 4\cos(2x))$. This resulted in the estimated H_m sizes as $H_m \leq \sqrt{\frac{36}{16.283\pi} * 10^{-k}}$ for composite trapezoidal rule and $H_m \leq \sqrt{\frac{72}{16.283\pi} * 10^{-k}}$ for composite midpoint rule.



e^(sin2x)cos(2x)	Hm Size (Estimate)	Error Estimate	Func Evals (Estimate)	Hm Size (Exact)	Exact Error	Predicted Hm	Error Threshold
Trapezoidal	0.008181231	2.74945E-05	65	0.004090615	2.749E-05	8.39E-03	1.00E-04
	0.000127832	6.71222E-09	2049	6.39159E-05	6.712E-09	8.39E-05	1.00E-08
	9.98685E-07	4.08858E-13	262145	4.99343E-07	4.16E-13	8.39E-07	1.00E-12
Midpoint	0.019392547	-7.73075E-05	27	0.006464182	-7.72E-05	1.12E-02	0.0001
	7.98047E-05	-1.30803E-09	729	2.66E-05	-1.31E-09	1.12E-04	0.00000001
	9.85243E-07	-2.00631E-13	531441	3.28E-07	-2.21E-13	1.12E-06	1E-12

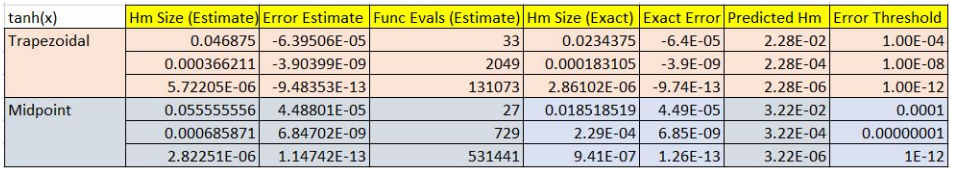
Figure 4: Numerical Quadrature of $e^{\sin(2x)} \cos(2x)$ on $[0, \pi/3]$

As Figure 4 shows, the H_m sizes at the points when the exact error was less than each respective error threshold are less than the compute a priori bounds once again. In this case, the estimated H_m sizes from the error estimates for the composite trapezoidal rule are also less than the predicted threshold. However, this is most likely just a function of a different interval $[a, b]$ on which when the interval was halved, the estimated error was just above the threshold and then the next interval halving resulted in much smaller subintervals, where the error estimate was much farther away than with e^x.

The composite midpoint rule continues to perform better than the composite trapezoidal rule when the error threshold is low, while performing worse when the error threshold is tight, repeating the trends indicated in section 5.1.

5.3 $\int_{-2}^{1} \tanh dx$

The third integral to be evaluated was $\int_{-2}^{1} \tanh dx = \ln\left(\frac{\cosh(1)}{\cosh(2)}\right) \approx -0.8912219$. When computing the a priori error bound, the max norm, $||f''||_\infty$, on $[-2, 1]$ was found to be 0.77 after computing the second derivative as $-2\tanh(x)sech^2(x)$. This resulted in the predicted step sizes being $H_m \leq \sqrt{\frac{4}{0.77} * 10^{-k}}$ for composite trapezoidal rule and $H_m \leq \sqrt{\frac{8}{0.77} * 10^{-k}}$ for composite midpoint rule.



tanh(x)	Hm Size (Estimate)	Error Estimate	Func Evals (Estimate)	Hm Size (Exact)	Exact Error	Predicted Hm	Error Threshold
Trapezoidal	0.046875	-6.39506E-05	33	0.0234375	-6.4E-05	2.28E-02	1.00E-04
	0.000366211	-3.90399E-09	2049	0.000183105	-3.9E-09	2.28E-04	1.00E-08
	5.72205E-06	-9.48353E-13	131073	2.86102E-06	-9.74E-13	2.28E-06	1.00E-12
Midpoint	0.055555556	4.48801E-05	27	0.018518519	4.49E-05	3.22E-02	0.0001
	0.000685871	6.84702E-09	729	2.29E-04	6.85E-09	3.22E-04	0.00000001
	2.82251E-06	1.14742E-13	531441	9.41E-07	1.26E-13	3.22E-06	1E-12

Figure 5: Numerical Quadrature of $\tanh(x)$ on $[-2, 1]$

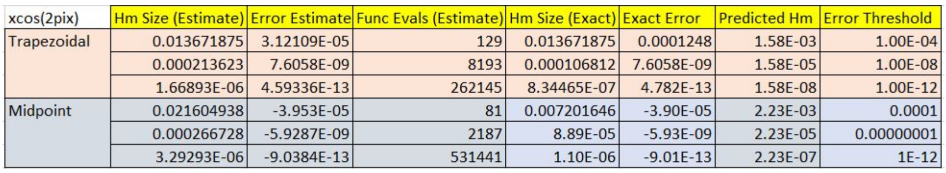
Exact H_m size once again confirms theory by being less than the a priori bound of the step size in all cases. When the H_m sizes based on the error estimates were computed, they were slightly above of below or above, which is expected due to the inexact nature of the estimates. However, they were all still of the same order of magnitude as the predictions.

In this case, the composite midpoint rule performs just slightly better than the composite trapezoidal rule in terms of efficiency at 10^{-4}, much better at 10^{-8}, and significantly worse for 10^{-12}, as column 3 of Figure 5 shows. This falls in line with our previous observations.

5.4 $\int_{0}^{3.5} x\cos(2\pi x)dx$

The fourth integral to be computed was $\int_{0}^{3.5} x\cos(2\pi x)dx$ which evaluates analytically to $1/2\pi^2 \approx -0.0506607$. The max norm of the second derivative was computed to be ≈ 138.174, where the second derivative was calculated as $-4\pi\sin(2\pi x) - 4\pi^2 x\cos(2\pi x)$. This resulted in the a priori bound on H_m being $H_m \leq \sqrt{\frac{r}{3.5*138.174} * 10^{-k}}$ where $r = 12$ for composite trapezoidal and $r = 24$ for composite midpoint rule.

The size of H_m empirically determined in the program appears to be one order of magnitude larger than the H_m predicted by the a priori bound. We



xcos(2pix)	Hm Size (Estimate)	Error Estimate	Func Evals (Estimate)	Hm Size (Exact)	Exact Error	Predicted Hm	Error Threshold
Trapezoidal	0.013671875	3.12109E-05	129	0.013671875	0.0001248	1.58E-03	1.00E-04
	0.000213623	7.6058E-09	8193	0.000106812	7.6058E-09	1.58E-05	1.00E-08
	1.66893E-06	4.59336E-13	262145	8.34465E-07	4.782E-13	1.58E-08	1.00E-12
Midpoint	0.021604938	-3.953E-05	81	0.007201646	-3.90E-05	2.23E-03	0.0001
	0.000266728	-5.9287E-09	2187	8.89E-05	-5.93E-09	2.23E-05	0.00000001
	3.29293E-06	-9.0384E-13	531441	1.10E-06	-9.01E-13	2.23E-07	1E-12

Figure 6: Numerical Quadrature of $x\cos(2\pi x)$ on $[0, 3.5]$

suspect that this has to do with the extreme oscillations of the function as well as it's second derivative over a very small interval, as seen in Figure 7. This causes there to be significant cancellations that result in inaccurate floating point computations when the numerical quadrature is being computed. [3]

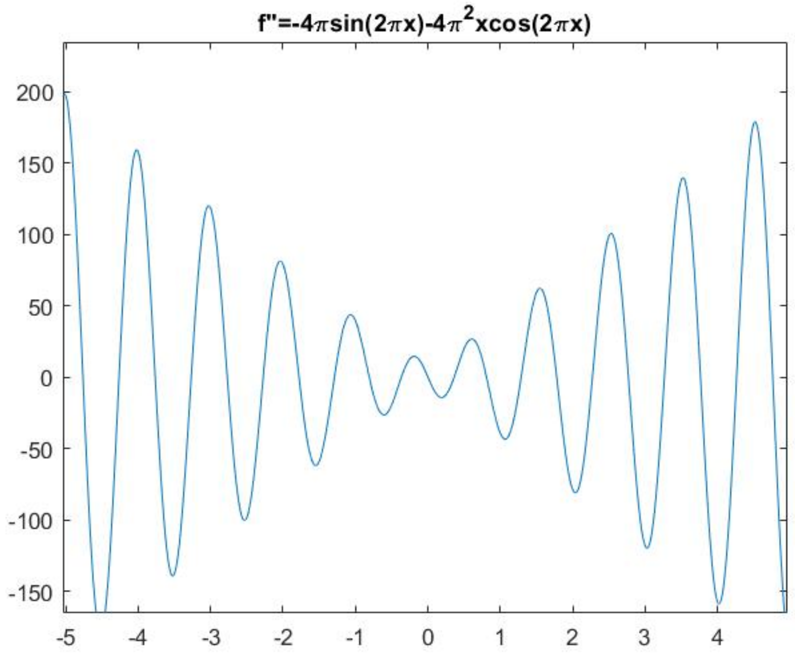


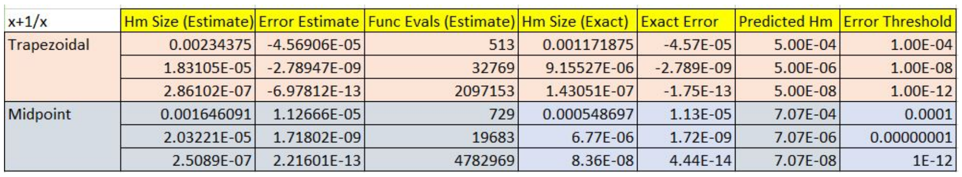
Figure 7: Second derivative used in the calculation of the max norm

The same trends in terms of the function evaluations once again repeat themselves, with midpoint better for looser error thresholds and worse for tighter.

5.5 $\int_{0.1}^{2.5} x + \frac{1}{x} dx$

The final integral to be investigated was $\int_{0.1}^{2.5} x + \frac{1}{x} dx$ which evaluates analytically to ≈ 6.3388758. Computation of $||f''||_\infty$ on the interval was determined to be

2000 with f'' computed as $\frac{2}{x^3}$. This gave a bound on H_m of $H_m \leq \sqrt{\frac{1}{400} * 10^{-k}}$ for composite trapezoidal rule and $H_m \leq \sqrt{\frac{1}{200} * 10^{-k}}$ for composite midpoint rule.



x+1/x	Hm Size (Estimate)	Error Estimate	Func Evals (Estimate)	Hm Size (Exact)	Exact Error	Predicted Hm	Error Threshold
Trapezoidal	0.00234375	-4.56906E-05	513	0.001171875	-4.57E-05	5.00E-04	1.00E-04
	1.83105E-05	-2.78947E-09	32769	9.15527E-06	-2.789E-09	5.00E-06	1.00E-08
	2.86102E-07	-6.97812E-13	2097153	1.43051E-07	-1.75E-13	5.00E-08	1.00E-12
Midpoint	0.001646091	1.12666E-05	729	0.000548697	1.13E-05	7.07E-04	0.0001
	2.03221E-05	1.71802E-09	19683	6.77E-06	1.72E-09	7.07E-06	0.00000001
	2.5089E-07	2.21601E-13	4782969	8.36E-08	4.44E-14	7.07E-08	1E-12

Figure 8: Numerical Quadrature of $x + \frac{1}{x}$ on $[0.1, 2.5]$

Predicted H_m size was again one order of magnitude smaller than empirically determined H_m size from the error estimates for both methods, which may be due to the inexact nature of the estimate, as seen in almost all previous examples. However, this was also true for the H_m sizes based on the exact error for the trapezoidal rule, while not so for the midpoint rule, which remained less than the predicted H_m for all error thresholds except 10^{-12}. We expect that these differences come from how the mesh is evaluated for this particular function. The magnitude $x + 1/x$ gets very, very large near 0 due to the asymptote, which might cause significant cancellations for large magnitude differences. Since the trapezoidal rule is evaluated on a close mesh, the values of f at x=0.1 is used even when there is only 1 interval.This value and those close to x=0.1 will dominate the values close to x=2.5, causing computational inaccuracy, meaning that the error labelled as exact is only an approximation of the true error since the value output by the program is only an approximation of what the true output of the trapezoidal rule should be for a particular step size. This results in the appearance that our value is closer is to the true value faster than an exact computation of the trapezoidal rule would actually produce.

However, since the midpoint rule is open, the x values are far away from the endpoints until there are many, many subdivisions of the interval. The numerical inaccuracies therefore only present themselves when the error threshold is very tight (i.e. after many subdivisions have occurred).

Interestingly, the midpoint rule only performed better in terms of function evaluations for the 10^{-8}. This may also be attributed to the computational inaccuracies of the trapezoidal underestimating the H_m size needed to achieve a particular error threshold, as described above. If the computations for the trapezoidal rule were actually correct in every sense, then we would expect to the midpoint rule to perform better. However, in this case, the fact that the results are not correct makes the trapezoidal rule look like it is converging to the true integral's value quicker than we would expect from an analytical evaluation of the trapezoidal rule. There is no reason to assume that this would happen for all functions, as seen in previous examples.

6 Conclusions

In conclusion, our experiments seem to indicate that the composite midpoint rule is generally more efficient than the composite trapezoidal rule, despite the fact that there are $3m$ function evaluations for midpoint, while only $2m + 1$ for composite trapezoidal rule. For the error thresholds less than 10^{-12} investigated, this was because the subintervals had to be divided in thirds significantly fewer times for composite midpoint than they were divided in half for composite trapezoidal. For the error threshold equal to 10^{-12}, the number of subinterval divisions caught up, possibly because of the exponential increase in the number of evaluations for an additional split of the current intervals. For example, if the exact error was just slightly above the threshold, then another split would push it significantly lower but with significantly more computations since the number of computations per split increases more rapidly with midpoint than with composite trapezoidal.

One counterexample was with the final integral, where the computational inaccuracies of floating point associated with the function $x + 1/x$ near zero pushed it close to the true value than perhaps an exact analytical calculation of composite trapezoidal rule would determine, resulting in fewer computations at the 10^{-4}.

Our experiments were also able to confirm the a priori bounds on the size of H_m to achieve various error levels in most cases. While in most cases the actual H_m size for the error estimate was slightly larger than the predictions, it was generally of the same order of magnitude as the prediction, perhaps indicating that the small numerical inaccuracies in the error estimation contributed to it being slightly off.

For the last two integrals, where the magnitudes of the estimates of H_m were one larger than the prediction, properties of the functions can explain the discrepancy. For $x \cos(2)$, the extremely wild oscillations of the function on the interval $[0, 3.5]$ results in cancellations during floating point operations that lead to an underestimate of the true error the quadrature methods from the exact integral value. Cancellations due to the significantly differing magnitudes of values during the quadrature of $x + 1/x$ near and away from 0 result in inaccuracies in floating point operations as well, leading to the same result.

7 Program Files

All programs were written in java and compiled using "javac FILENAME.java", "java FILENAME". CompTrapRule.java has the code for composite trapezoidal rule while CompMdptRule.java has the composite midpoint rule code. All test functions were hard coded in and then commented out for easy reuse by uncommenting.

References

[1] www.math.fsu.edu/ gallivan/courses/FCM1new/Locked/Sets/set22.pdf

[2] www.math.fsu.edu/ gallivan/courses/FCM1new/Locked/Sets/set25.pdf

[3] www.math.fsu.edu/ gallivan/courses/FCM1new/Locked/Sets/set4.pdf