Adaptive Quadrature Analysis, Program 4

Colton Piper

December 4th, 2017

1 Executive Summary

In this report we will combine different quadrature methods and find there errors. We will also use Richardson extrapolation and more specifically Romberg extrapolation. The Romberg method is extrapolation of the trapezoid rule with itself, but with its refined method. We will analyze the errors of these different quadrature methods.

2 Statement of the Problem

First find the general solution of these new quadrature methods using extrapolation, which makes the new quadrature method a higher order approximation. Then find the coefficients of the error of the new quadrature methods. Also find the order of infinitesimal of the quadrature method and the degree of exactness of the quadrature methods. Then code the Romberg extrapolation which extrapolates the trapezoidal rule with itself but the amount of sub-intervals in the main interval is doubled. Thus the step size is double. We will see that Romberg extrapolation increases the order of the two quadratures being combined and does not use any more function evaluations.

3 Description of the Mathematics

Extrapolation is the mathematics of two methods of the degree of exactness being combined to yield a new method that has a higher degree of exactness. A new quadrature method we will define is $I_{new} = \alpha_1 I_1 + \alpha_2 I_2$ where we have

$$\alpha_1 = \tfrac{C_2}{C_2 - C_1} \quad \alpha_2 = \tfrac{C_1}{C_1 - C_2}.$$

This is where C_i is the coefficient of the first term of the error of the quadrature method. There are a few different methods to get the errors of the new methods. To get these error terms though require a Taylor expansion of the function and that requires that the function being approximated is sufficiently smooth.

The way to get the error of the newly formed quadrature method is to Taylor expand the function around one of the mesh that is used in one of the combined quadrature methods. Then we integrate that Taylor expansion subtract it from the quadrature method with the function evaluations using the Taylor expansion around our point and we have the error.

3.1 Simpson's Methods Combined

First let us combine Simpson's quadrature methods. The first method uses the function evaluations at the endpoints and at the midpoint and the second rule uses the evaluations also at the midpoints but also at two evenly space points inside the interval.

$$I \approx I_{first} = I_f = \frac{b-a}{6}\big[f(x_0) + 4f(x_2) + f(x_4)\big] \quad E_f = -\frac{1}{2880}(b-a)^5 f^{(4)}(\zeta)$$
$$I \approx I_{second} = I_s = \frac{b-a}{8}\big[f(x_0) + 3f(x_1) + 3f(x_3) + f(x_4)\big] \quad E_f = -\frac{1}{6480}(b-a)^5 f^{(4)}(\eta)$$

Thus we will have

$$C_f = -\frac{1}{2880} \qquad C_s = -\frac{1}{6480}$$
$$\alpha_f = -\frac{4}{5} \qquad \alpha_s = \frac{9}{5}$$

$$\begin{aligned} I_{new} = \frac{9}{5}I_s - \frac{4}{5}I_f &= (b-a)\left[\frac{11}{120}f_0 + \frac{27}{40}f_1 - \frac{8}{15}f_2 + \frac{27}{40}f_3 + \frac{11}{120}f_4\right] \\ &= (b-a)\left[\frac{11}{120}(f_0+f_4) + \frac{27}{40}(f_1+f_3) - \frac{8}{15}f_2\right]. \end{aligned}$$

Now from the notes we know that we know that the Taylor expansion around the midpoint is

$$f(x) = f_2 + f_2'(x-x_2) + \frac{1}{2}(x-x_1)^2 f_1'' + \frac{1}{3!}(x-x_1)^3 f_1^{(3)} + \frac{1}{4!}(x-x_1)^4 f_1^{(4)} + \frac{1}{5!}(x-x_1)^5 f_1^{(5)} + \frac{1}{6!}(x-x_1)^6 f_1^{(6)} + \frac{1}{7!}(x-x_1)^7 f_1^{(7)} + O(h^8)$$

and integrating it yields to the result of

$$I = 2hf_2 + \frac{2}{3!}h^3 f_2'' + \frac{2}{5!}h^5 f_2^{(4)} + \frac{2}{7!}h^7 f^{(6)} + O(h^8)$$

where $h = (b-a)/2$ and $f_i = f(x_i)$. Now using the Taylor expansion for the other function evaluations yields

$$\begin{aligned}
f_0 &= f_2 - f_2'h + \frac{1}{2}h^2 f_2'' - \frac{1}{3!}h^3 f_2^{(3)} + \frac{1}{4!}h^4 f_2^{(4)} - \frac{1}{5!}h^5 f_2^{(5)} + \frac{1}{6!}h^6 f_2^{(6)} + O(h^7)\\
f_1 &= f_2 - \frac{1}{3}f_2'h + \frac{1}{2\cdot 3^2}h^2 f_2'' - \frac{1}{3!\cdot 3^3}h^3 f_2^{(3)} + \frac{1}{4!\cdot 3^4}h^4 f_2^{(4)} - \frac{1}{5!\cdot 3^5}h^5 f_2^{(5)} + \frac{1}{6!\cdot 3^6}h^6 f_2^{(6)} + O(h^7)\\
f_3 &= f_2 + \frac{1}{3}f_2'h + \frac{1}{2\cdot 3^2}h^2 f_2'' + \frac{1}{3!\cdot 3^3}h^3 f_2^{(3)} + \frac{1}{4!\cdot 3^4}h^4 f_2^{(4)} + \frac{1}{5!\cdot 3^5}h^5 f_2^{(5)} + \frac{1}{6!\cdot 3^6}h^6 f_2^{(6)} + O(h^7)\\
f_4 &= f_2 + f_2'h + \frac{1}{2}h^2 f_2'' + \frac{1}{3!}h^3 f_2^{(3)} + \frac{1}{4!}h^4 f_2^{(4)} + \frac{1}{5!}h^6 f_2^{(5)} + \frac{1}{6!}h^6 f_2^{(6)} + O(h^7).
\end{aligned}$$

Then substituting those into our new method and then subtract it from the actual we have

$$\begin{aligned}
I_{new} &= h\Bigg[\frac{11}{60}\Bigg(2f_2 + h^2 f_2'' + \frac{2}{4!}h^4 f_2^{(4)} + \frac{2}{6!}h^6 f_2^{(6)} + O(h^8)\Bigg)\\
&\quad + \frac{81}{60}\Bigg(2f_2 + \frac{1}{3^2}h^2 f'' + \frac{2}{4!\cdot 3^4}h^4 f_2^{(4)} + \frac{2}{6!\cdot 3^6}h^6 f_2^{(6)} + O(h^8)\Bigg) + \frac{64}{60}f_2\Bigg]\\
I_{new} &= 2hf_2 + \frac{1}{3}h^3 f_2'' + \frac{1}{60}h^5 f_2^{(4)} + \frac{1}{1944}h^7 f^{(6)} + O(h^8)\\
E_{new} &= -\frac{1}{8505}h^7 f_2^{(6)} + O(h^8)
\end{aligned}$$

where $h = (b-a)/2$. From the new method we can see that it is not a Newton-Cotes Method because it does not use a uniform spacing mesh. But from Theorem 17.1 in the class notes this is an interpolatory quadrature method due to its degree of exactness being at least 1 higher than the number of points being used in the method.

3.2 Three Point Newton-Cote Methods Combined

Next we are going to combine the two methods of Simpson's First Rule and the three point open Newton-Cotes method. First these two methods are

$$I \approx I_{first} = I_f = \frac{b-a}{6}\big[f(x_0) + 4f(x_2) + f(x_4)\big] \quad E_f = -\frac{1}{2880}(b-a)^5 f^{(4)}(\zeta)$$
$$I \approx I_3 = \frac{(b-a)}{3}\big[2f(x_1) - f(x_2) + 2f(x_4)\big] \quad E_3 = \frac{7}{23040}(b-a)^5 f^{(4)}(\eta)$$

Thus we will have

$$C_f = -\frac{1}{2880} \qquad C_3 = \frac{7}{23040}$$
$$\alpha_f = \frac{7}{15} \qquad \alpha_3 = \frac{8}{15}$$

$$\begin{aligned} I_{new} &= \frac{7}{15}I_f + \frac{8}{15}I_3 = (b-a)\left[\frac{7}{90}f_0 + \frac{16}{45}f_1 + \frac{6}{45}f_2 + \frac{16}{45}f_3 + \frac{7}{90}f_4\right] \\ &= (b-a)\left[\frac{7}{90}(f_0+f_4) + \frac{16}{45}(f_1+f_3) + \frac{6}{45}f_2\right]. \end{aligned}$$

Then doing the same as we did with combining both Simpson methods we get the function evaluations of

$$\begin{aligned} f_0 &= f_2 - f_2'h + \frac{1}{2}h^2 f_2'' - \frac{1}{3!}h^3 f_2^{(3)} + \frac{1}{4!}h^4 f_2^{(4)} - \frac{1}{5!}h^5 f_2^{(5)} + \frac{1}{6!}h^6 f_2^{(6)} + O(h^7) \\ f_1 &= f_2 - \frac{1}{3}f_2'h + \frac{1}{2\cdot 2^2}h^2 f_2'' - \frac{1}{3!\cdot 2^3}h^3 f_2^{(3)} + \frac{1}{4!\cdot 2^4}h^4 f_2^{(4)} - \frac{1}{5!\cdot 2^5}h^5 f_2^{(5)} + \frac{1}{6!\cdot 2^6}h^6 f_2^{(6)} + O(h^7) \\ f_3 &= f_2 + \frac{1}{3}f_2'h + \frac{1}{2\cdot 2^2}h^2 f_2'' + \frac{1}{3!\cdot 2^3}h^3 f_2^{(3)} + \frac{1}{4!\cdot 2^4}h^4 f_2^{(4)} + \frac{1}{5!\cdot 2^5}h^5 f_2^{(5)} + \frac{1}{6!\cdot 2^6}h^6 f_2^{(6)} + O(h^7) \\ f_4 &= f_2 + f_2'h + \frac{1}{2}h^2 f_2'' + \frac{1}{3!}h^3 f_2^{(3)} + \frac{1}{4!}h^4 f_2^{(4)} + \frac{1}{5!}h^6 f_2^{(5)} + \frac{1}{6!}h^6 f_2^{(6)} + O(h^7). \end{aligned}$$

Then when we substitute those into our new method and subtract it from the actual we have

$$\begin{aligned}
I_{new} &= h\Bigg[\frac{7}{45}\Bigg(2f_2 + h^2 f_2'' + \frac{2}{4!}h^4 f_2^{(4)} + \frac{2}{6!}h^6 f_2^{(6)} + O(h^8)\Bigg) \\
&\quad + \frac{32}{45}\Bigg(2f_2 + \frac{1}{2^2}h^2 f'' + \frac{2}{4!\cdot 2^4}h^4 f_2^{(4)} + \frac{2}{6!\cdot 2^6}h^6 f_2^{(6)} + O(h^8)\Bigg) + \frac{12}{45}f_2\Bigg] \\
I_{new} &= 2hf_2 + \frac{1}{3}h^3 f_2'' + \frac{1}{60}h^5 f_2^{(4)} + \frac{1}{2160}h^7 f^{(6)} + O(h^8) \\
E_{new} &= -\frac{1}{15120}h^7 f_2^{(6)} + O(h^8)
\end{aligned}$$

where $h = (b-a)/2$. We can see from the notes that the new method actually produces a familiar method which is the five point closed Newton-Cotes method.

3.3 Romberg Method

The Romberg Method is extrapolation with trapezoidal rule and with trapezoidal refining methods. This method though has a nice pattern to it when you extrapolate the new methods. You can keep extrapolating and getting higher and higher order methods. Lets have notation being I_s^k where s is the number of extrapolations and k is the level of extrapolation. From the notes we have that each new approximation is going to have the form

$$I_s^k = \frac{4^k I_{2s}^{k-1} - I_s^{k-1}}{4^k - 1}$$

Having a recursive formula for extrapolation mixed with refining is very powerful and useful. That formula comes from the Euler-Maclaurin error formula for the error of the trapezoidal rule

$$I - I_s^0 = C_2^0 h_s^2 + C_4^0 h_s^4 + C_6^0 h_s^6 + C_8^0 h_s^8 + O(h^{10}).$$

4 Description of the Algorithms and Implementation

The majority of the code for this program came from program 3. The refining trapezoidal rule was modified to complete an empty passed in array that will be used as the first column of Romberg triangle displaying what the method is doing. The code uses the trapezoidal rule which is integrating a linear interpolating polynomial over each of the sub-intervals using the endpoints as the interpolating points. Now because this is made for a uniform sub-interval we know that each interval width dx will be the same. We also know that the single trapezoidal rule has the form $\frac{dx}{2}(f(x_1)+f(x_2))$. So a composite trapezoidal rule will be $dx(0.5*f(x_1)+f(x_2)+f(x_2)+\cdots+f(x_{n-1})+0.5*f(x_n))$. Thus we can add the two evaluations at the end points divide by two, then implement a simple for loop to add the evaluations inside the interval to the sum and lastly multiply by dx.

The refinement code first uses the simple composite trapezoidal rule to solve for the initial sum. Then we found smallest factor we could multiply the amount of sub-intervals by and still reuse all the already evaluated points is 2. Double the intervals still yields with the old evaluations being used because the boundaries of the old sub-intervals are still boundaries for the new sub-intervals, thus they will be used for evaluations in this iteration. Now a new iteration yields new evaluations as well. Now these evaluations will be spaced apart by the dx of the last iteration. So we can implement a for loop that evaluates at $x+dx/2+i*dx$ for $i=0$ to the last new evaluation point. Those new evaluations added together then multiplied by this iteration's dx will be the new stuff needed to add the old sum divided by 2. The recursive formula looks like this

$$sum_{i+1} = 0.5*(sum_i) + dx_{i+1}\left(\sum_{k=1}^{2^{i-1}*n} f_{i+1,k}\right)$$

where n is the number of initial sub-intervals, and f_i is the new evaluations of the i^{th} iteration.

The Romberg method then just uses the recursive formula to determine each column until you get to the final column which will be the best approximation. This all together returns the array which is the first approximation in each column or level of extrapolation.

While computing the extrapolated columns you will need to use the already computed approximations twice except for the first and last in the column which only get used once. Now because of this it is better to compute the columns in an upward direction or the most refined approximation first then replace the same level of refinement with that new approximation. Doing this instead of computing the worst approximation of the column first saves you from having to create a temporary variable and makes the code less cluttered.

5 Description of the Experimental Design and Results

The five integrals given are the first five and the others are my own functions to check my routines. They are

$$\int_0^3 e^x dx = e^3 - 1 \approx 19.0855369232 \tag{1}$$

$$\int_0^{\pi/3} e^{\sin(2x)} \cos(2x) dx = 0.5(e^{\sqrt{3}/2} - 1) \approx 0.6887213376 \tag{2}$$

$$\int_{-2}^{1} \tanh(x) dx = ln(\cosh(1) - ln(\cos(-2)) \approx -0.89122191687 \tag{3}$$

$$\int_0^{3.5} x \cos(2\pi x) dx = -\frac{1}{2\pi^2} \approx -0.050660591821169 \tag{4}$$

$$\int_{0.1}^{2.5} x + \frac{1}{x} dx = 3.12 + \ln(\frac{2.5}{0.1}) \approx 6.3388758248682 \tag{5}$$

Let us try getting within about 0.1%. Thus for the first integral let us get within 0.02, 0.0007, 0.0009, 0.00005, 0.006, for the first, second, third, fourth, fifth, and sixth functions respectively. First let us only use refining to see how many iterations it takes to get a good approximation.

Iteration	Sub-Intervals	Error
0	1	12.5427684616
1	2	3.4511493747
2	4	0.8863581155
3	8	0.2231361849
4	16	0.0558819238
5	32	0.0139766175
0	1	0.7875355989
1	2	0.1157154313
2	4	0.0284121854
3	8	0.0070548014
4	16	0.0017606058
5	32	0.0004399571
0	1	0.5875717807
1	2	0.0462211129
2	4	0.0161977817
3	8	0.0040829335
4	16	0.0010227665
5	32	0.0002558125
0	1	6.0743394082
1	2	3.0118394082
2	4	2.5633466669
3	8	0.0488291355
4	16	0.0087890133
5	32	0.0020417938
6	64	0.0005014102
7	128	0.0001247976
8	256	0.0000311649
0	1	9.2611241751
1	2	3.9442010982
2	4	1.5355949675
3	8	0.5322232077
4	16	0.1624694155
5	32	0.0446408592
6	64	0.0115449192
7	128	0.0029148668

Table 1: Errors for each refinement of trapezoidal rule for each function in order as listed above.

Now let us see how many iterations it takes of the Romberg Method it takes to get within the same error tolerance.

Func	Level 1	2	3	4	5	6
1	0.4206096791	0.0054822302	0.0000191482	0.0000000170		
2	0.1082246245	0.0064801519	0.0001259082	0.0000007403	0.0000000013	
3	0.1342291097	0.0155512789	0.0006176794	0.0000081524	0.0000000381	
4	1.9910060748	2.4420386205	1.0575709385	0.0688383860	0.0009975969	0.0000034633
5	2.1718934059	0.6367817807	0.1545673335	0.0260278790	0.0025860361	0.0001301125

Table 2: The best extrapolation result for number of refinements.

From table 2 we can see that the Romberg method can get you fantastic results compared to just refining alone. For the first function it took us using the composite trapezoidal rule with 32 sub-intervals to get an error within 0.02. But with extrapolation we only need need the composite rule with 4 sub-intervals. Then the highest level of extrapolation gets us within that error tolerance of 0.02. We can see that the same thing also occurs with the other functions and the amount of sub-intervals needed is in the following table

Func Refining	1	2	3	4	5
Sub-Intervals	32	32	32	256	128
Function Evals	33	33	33	255	129
Func Romberg	1	2	3	4	5
Sub-Intervals	4	8	8	64	64
Function Evals	5	9	9	65	65

Table 3: Shows the difference between needed sub-intervals to get within a given error tolerance for refining method and Romberg method.

Getting within that error tolerance with half the amount of function evaluations can be very useful. As some functions might have hundreds of operations in it. Thus in these types of cases we could save a lot of time and in some cases quality also as floating point error might come into play if the step sizes for the sub-intervals get small enough.

Lastly let us try to see how the order of convergence is compared with the Romberg solution. Because the order theoretically should go from h^2 and increase by two every level of extrapolation we should see that in each column in the extrapolation. So for function one here are the errors for all the Romberg Triangle with six iterations

	0	1	2	3	4	5
1	12.5427684616					
2	3.4511493747	0.4206096791				
4	0.8863581155	0.0314276957	0.0054822302			
8	0.2231361849	0.0020622080	0.0001045088	1.91482E-5		
16	0.0558819238	0.0001305035	0.0000017232	9.17E-8	1.7E-8	
32	0.0139766175	0.0000081820	0.0000000273	4E-10	small	small

Table 4: This shows the errors for the full Romberg Triangle up to 5 levels for function 1.

From this we can see that as you refine by half in the first regular composite you get the errors decreasing by a fourth each time which makes sense because the error term is h^2 and when you half your step size it gets squaed and becomes a fourth. Then in the first level of extrapolation we expect that each will decrease on the order of about one-sixteenth, which it does. Then for the second level we are expecting the error to decrease by a factor of sixty four and it does for the few values we have. The third level also decreases by a factor on the order of 256. Thus we can see that the columns of our romberg methods do decrease on the order that we were expecting, but it does not have to decrease. We can see from the next table. In general though it may take a 4 or five iterations for Romberg to kick in but it can improve the error drastically down the line.

	0	1	2	3	4	5
1	6.0743394082					
2	3.0118394082	1.9910060748				
3	2.5633466669	2.4138490864	2.4420386205			
4	0.0488291355	0.7893433750	1.0028895391	1.0575709385		
5	0.0087890133	0.0045576941	0.0477613513	0.0644383495	0.0688383860	
6	0.0020417938	0.0002072793	0.0000827483	0.0009293977	0.0009293977	0.0009975969

Table 5: This shows the errors for the full Romberg Triangle up to 5 levels for function 4.

6 Conclusions

We showed how you can compute a new higher degree of exactness method with two lower degree methods. Then we found the degree and order of infinitesimal for those new methods. After we discussed a more precise method of extrapolation combing a single rule with its refined method. We could use these refined methods to get more and more precise and higher degree methods. This led us to test how these higher degrees compare to just refining alone.

We saw that the Romberg method decreased the number of function evaluations needed to get to the given error tolerance for all five of the functions being tested. That is wonderful if the function is large and complex as it can save a lot of time. Thus what we saw is that because the Romberg method is so easy to implement and takes so little time as each extrapolation requires little flops you should use it if you are already going to be using the trapezoid rule.

7 Program Files

The program is simple enough. First add the function to the functions routine. The romberg routine accepts an array for the x values being used, the number of mesh, the number of the function declared in the functions routine, the number of refinements the user wants to be used, and the exact integral if the user wants to see what the error is of the approximation if the user has the actual result. Then the Romberg algorithm returns the highest degree of extrapolation that can be achieved with the amount of refinements used.