Program 4

David Miller

MAD5403: Foundations of Computational Math I

December 6, 2017

1 Executive Summary

In this program we investigate extrapolation between Simpson's first and Simpson's second and Simpson's first and Open Netwon Cotes that uses 3 points. We alos investigate the accuracy and convergence between Romberg's method and Trapezoidal with refinement. An analysis is given for these and results are presented at the end to give numerical evidence of our analysis.

2 Statement of Problem

This program has two main parts

1. Compute error forms for two extrapolation methods: $I_{new} = \frac{9}{5}I_{ss} - \frac{4}{5}I_{sf}$ and $I_{new} = \frac{7}{15}Isf + \frac{8}{15}I_{o2}$ determine if they are Newton Cotes methods and/or if they are interpolatory quadrature

2. Numerically test and investigate Trapezoidal method with refinement vs Romberg's method and determine the relation between the two

The former is analysis that is done out in the section 3 and the latter is presented in sections 4, 5, and 6. Numerical tests were all carried out in double precision.

3   Description of Mathematics

In this assignment we use four quadrature methods : Simpson's first, Simpson's second, and Open Newton Cotes formula that uses 3 points, and Trapezoidal. We use the notation $f_i^{(m)}$ where $f(x_i)$ is the function evaluation for the mesh point x_i in method m.

Simpson's First Rule

$$I_{sf} = \frac{1}{3}h_{sf}\left[f_0^{(sf)4f_1^{(sf)}} + f_2^{(sf)}\right] = (b-a)\left[\frac{1}{6}f_0^{(sf)} + \frac{2}{3}f_1^{(sf)} + f_2^{(sf)}\right], \quad E_{sf} = -\frac{1}{2880}(b-a)^5 f^{(4)}(\eta)$$

where $h_{sf} = (b-a)/2$.

Simpson's Second Rule

$$I_{ss} = \frac{3}{8}h_{ss}\left[f_0^{(ss)} + 3f_1^{(ss)} + 3f_2^{(ss)} + f_3^{(ss)}\right] = (b-a)\left[\frac{1}{8}f_0^{(ss)} + \frac{3}{8}f_0^{(ss)} + \frac{3}{8}f_2^{(ss)} + \frac{1}{8}f_0^{(ss)}\right],$$

$$E_{ss} = -\frac{1}{6480}(b-a)^5 f^{(4)}(\xi)$$

where $h_{ss} = (b-a)/3$

Open Newton Cotes that uses 3 points

$$I_{o2} = \frac{4}{3}h_{o2}\left[2f_0^{(o2)} - f_1^{(o2)} + 2f_2^{(o2)}\right] = (b-a)\left[\frac{2}{3}f_0^{(o2)} - \frac{1}{3}f_1^{(o2)} + \frac{2}{3}f_2^{(o2)}\right], \quad E_{o2} = \frac{7}{23040}(b-a)^5 f^{(4)}(\mu)$$

where $h_{o2} = (b-a)/4$

Trapezoidal Rule

$$I_t = \frac{1}{2}h_t\left[f_0^t + f_1^{(t)}\right] = (b-a)\left[\frac{1}{2}f_0^{(t)} + \frac{1}{2}f_1^{(t)}\right], \quad E_t = -\frac{1}{12}f^{(2)}(\chi)$$

where $h_t = b - a$

We will also make use of the following theorem

Theorem 1. A quadrature formula using $n+1$ distinct points is an interpolatory quadrature formula if and only if it has degree of exactness greater than or equal to n

3.1 Simpson's First and Simpson's Second

Simpson's first and Simpson's second methods can be combined

$$I_{new} = \frac{9}{5}I_{ss} - \frac{4}{5}I_{sf} \tag{1}$$

where its error can be expressed as $I - I_{new} = Ch^d f^{(s)} + \mathcal{O}(h^{d+1})$. To figure out the constant C, power d, and order of derivative s we first need to determine an expression for I_{new}. Letting $h = \frac{b-a}{2}$ we have
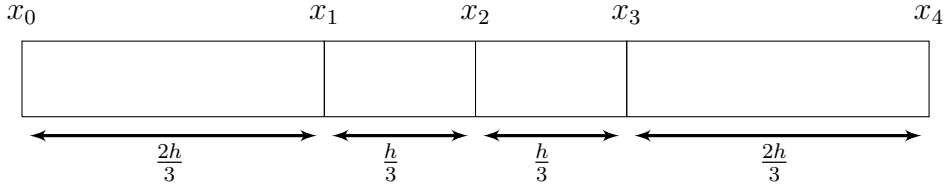
where x_0, x_2, and x_4 come from I_{sf} and x_0, x_1, x_3, and x_4 come from I_{ss}. We have that I_{new} is

$$
\begin{aligned}
I_{new} &= \frac{9}{5}I_{ss} - \frac{4}{5}I_{sf} \\
&= \frac{9}{5}\left[\frac{b-a}{8}\left(f_0 + 3f_1 + 3f_3 + f_4\right)\right] - \frac{4}{5}\left[\frac{b-a}{6}\left(f_0 + 4f_2 + f_4\right)\right] \\
&= \frac{9(b-a)}{40}\left(f_0 + 3f_1 + 3f_3 + f_4\right) - \frac{4}{30}\left(f_0 + 4f_2 + f_4\right) \\
&= \frac{b-a}{120}\left(11f_0 + 81f_1 - 64f_2 + 81f_3 + 11f_4\right) \\
&= \frac{h}{60}\left(11f_0 + 81f_1 - 64f_2 + 81f_3 + 11f_4\right)
\end{aligned}
$$

All that is left is to find an expression for I. We can do this by finding an expression for $f(x)$ by Taylor expanding around x_2 and integrating it.

$$
\begin{aligned}
f(x) &= f_2 + f_2^{(1)}\frac{(x-x_2)}{1!} + f_2^{(2)}\frac{(x-x_2)^2}{2!} + f_2^{(3)}\frac{(x-x_2)^3}{3!} + f_2^{(4)}\frac{(x-x_2)^4}{4!} \\
&\quad + f_2^{(5)}\frac{(x-x_2)^5}{5!} + f_2^{(6)}\frac{(x-x_2)^6}{6!} + \mathcal{O}((x-x_2)^7) \\
&= f_2 + f_2^{(1)}h + f_2^{(2)}\frac{h^2}{2!} + f_2^{(3)}\frac{h^3}{3!} + f_2^{(4)}\frac{h^4}{4!} + f_2^{(5)}\frac{h^5}{5!} + f_2^{(6)}\frac{h^6}{6!} + \mathcal{O}(h^7)
\end{aligned}
$$

To integrate we first do a change of variables by letting $x = x_2 + sh$ for $-1 \leq s \leq 1$. Therefore we have $dx = hds$ and end up with

$$I = h \int_{-1}^{1} f(s)\,ds$$

$$= h \int_{-1}^{1} f_2 + f_2^{(1)} \frac{sh}{1!} + f_2^{(2)} \frac{(sh)^2}{2!} + f_2^{(3)} \frac{(sh)^3}{3!} + f_2^{(4)} \frac{(sh)^4}{4!} + f_2^{(5)} \frac{(sh)^5}{5!} + f_2^{(6)} \frac{(sh)^6}{6!}\,ds$$

$$= h \left(2f_2 + f_2^{(2)} \frac{2h^2}{3!} + f_2^{(4)} \frac{2h^4}{5!} + f_2^{(6)} \frac{2h^6}{7!} \right)$$

Combining the expression for I and I_{new} we get

$$I - I_{new} = h \left(2f_2 + f_2^{(2)} \frac{2h^2}{3!} + f_2^{(4)} \frac{2h^4}{5!} + f_2^{(6)} \frac{2h^6}{7!} \right)$$
$$- \frac{h}{60} \left(11f_0 + 81f_1 - 64f_2 + 81f_3 + 11f_4 \right)$$
$$= h \left(2f_2 + f_2^{(2)} \frac{2h^2}{3!} + f_2^{(4)} \frac{2h^4}{5!} + f_2^{(6)} \frac{2h^6}{7!} \right)$$
$$- \frac{h}{60} \left[11 \left(f_2 - f_2^{(1)} \frac{h}{1!} + f_2^{(2)} \frac{h^2}{2!} - f_2^{(3)} \frac{h^3}{3!} + f_2^{(4)} \frac{h^4}{4!} - f_2^{(5)} \frac{h^5}{5!} + f_2^{(6)} \frac{h^6}{6!} \right) \right.$$
$$+ 81 \left(f_2 - f_2^{(1)} \frac{h}{1!3} + f_2^{(2)} \frac{h^2}{2!3^2} - f_2^{(3)} \frac{h^3}{3!3^3} + f_2^{(4)} \frac{h^4}{4!3^4} - f_2^{(5)} \frac{h^5}{5!3^5} + f_2^{(6)} \frac{h^6}{6!3^6} \right)$$
$$- 64f_2$$
$$+ 81 \left(f_2 + f_2^{(1)} \frac{h}{1!3} + f_2^{(2)} \frac{h^2}{2!3^2} + f_2^{(3)} \frac{h^3}{3!3^3} + f_2^{(4)} \frac{h^4}{4!3^4} + f_2^{(5)} \frac{h^5}{5!3^5} + f_2^{(6)} \frac{h^6}{6!3^6} \right)$$
$$\left. + 11 \left(f_2 + f_2^{(1)} \frac{h}{1!} + f_2^{(2)} \frac{h^2}{2!} + f_2^{(3)} \frac{h^3}{3!} + f_2^{(4)} \frac{h^4}{4!} + f_2^{(5)} \frac{h^5}{5!} + f_2^{(6)} \frac{h^6}{6!} \right) \right]$$

Gathering like terms and simplifying we end up with the expression

$$I - I_{new} = -\frac{1}{8505} h^7 f_2^{(6)} + \mathcal{O}(h^8) \tag{2}$$

which implies

$$C = -\frac{1}{8505}, \quad d = 7, \quad s = 6$$

Since the degree fo exactness is 5, by Theorem 1 $I_{new} = \frac{9}{5} I_{ss} - \frac{4}{5} I_{sf}$ is a polynomial interpolation quadrature method. However, I_{new} is not Newton Cotes method since it interpolates at non-uniform mesh points.
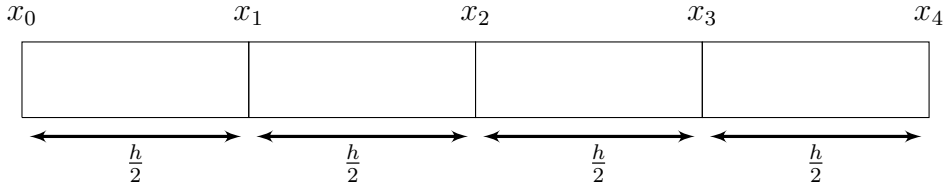
3.2 Simpson's First and Open-2

Simpson's first and Open-2 methods can be combined

$$I_{new} = \alpha_{sf} I_{sf} - \alpha_{O2} I_{O2} \tag{3}$$

where α_1 and α_2 are obtained via

$$\alpha_{sf} = \frac{C_{O2}}{C_{O2} - C_{sf}} = \frac{-\frac{7}{23040}}{-\frac{7}{23040} - \frac{1}{2880}} = \frac{7}{15}, \quad \alpha_{O2} = \frac{C_{sf}}{C_{sf} - C_{O2}} = \frac{\frac{1}{2880}}{\frac{1}{2880} + \frac{7}{23040}} = \frac{8}{15}$$

The error can be expressed as $I - I_{new} = Ch^d f^{(s)} + \mathcal{O}(h^{d+1})$. To figure out the constant C, power d, and order of derivative s we first need to determine an expression for I_{new}. Letting $h = \frac{b-a}{2}$ we have

$$
\begin{array}{ccccc}
x_0 & x_1 & x_2 & x_3 & x_4
\end{array}
$$

$$
\begin{array}{cccc}
\frac{h}{2} & \frac{h}{2} & \frac{h}{2} & \frac{h}{2}
\end{array}
$$

where x_0, x_2, and x_4 come from I_{sf} and x_1, x_2, x_3, and x_4 come from I_{o2}. We have that I_{new} is

$$
\begin{aligned}
I_{new} &= \frac{7}{15} I_{sf} + \frac{8}{15} I_{o2} \\
&= \frac{7}{15} \left[\frac{b-a}{6} \left(f_0 + 4f_2 + f_4 \right) \right] + \frac{8}{15} \left[\frac{b-a}{3} \left(2f_1 - f_2 + f_3 \right) \right] \\
&= \frac{7(b-a)}{90} \left(f_0 + 4f_2 + f_4 \right) + \frac{16(b-a)}{90} \left(2f_1 - f_2 + 2f_3 \right) \\
&= \frac{b-a}{90} \left(7f_0 + 32f_1 + 12f_2 + 32F_3 + 7f_4 \right) \\
&= \frac{h}{45} \left(7f_0 + 32f_1 + 12f_2 + 32F_3 + 7f_4 \right)
\end{aligned}
$$

All that is left is to find an expression for I. We can do this by finding an expression for $f(x)$ by Taylor expanding around x_2 and integrating it.

$$
\begin{aligned}
f(x) &= f_2 + f_2^{(1)} \frac{(x - x_2)}{1!} + f_2^{(2)} \frac{(x - x_2)^2}{2!} + f_2^{(3)} \frac{(x - x_2)^3}{3!} + f_2^{(4)} \frac{(x - x_2)^4}{4!} \\
&\quad + f_2^{(5)} \frac{(x - x_2)^5}{5!} + f_2^{(6)} \frac{(x - x_2)^6}{6!} + \mathcal{O}((x - x_2)^7) \\
&= f_2 + f_2^{(1)} h + f_2^{(2)} \frac{h^2}{2!} + f_2^{(3)} \frac{h^3}{3!} + f_2^{(4)} \frac{h^4}{4!} + f_2^{(5)} \frac{h^5}{5!} + f_2^{(6)} \frac{h^6}{6!} + \mathcal{O}(h^7)
\end{aligned}
$$

5

To integrate we first do a change of variables by letting $x = x_2 + sh$ for $-1 \leq s \leq 1$. Therefore we have $dx = h\,ds$ and end up with

$$I = h \int_{-1}^{1} f(s)\,ds$$

$$= h \int_{-1}^{1} f_2 + f_2^{(1)}\frac{sh}{1!} + f_2^{(2)}\frac{(sh)^2}{2!} + f_2^{(3)}\frac{(sh)^3}{3!} + f_2^{(4)}\frac{(sh)^4}{4!} + f_2^{(5)}\frac{(sh)^5}{5!} + f_2^{(6)}\frac{(sh)^6}{6!}\,ds$$

$$= h\left(2f_2 + f_2^{(2)}\frac{2h^2}{3!} + f_2^{(4)}\frac{2h^4}{5!} + f_2^{(6)}\frac{2h^6}{7!}\right)$$

Combining the expression for I and I_{new} we get

$$I - I_{new} = h\left(2f_2 + f_2^{(2)}\frac{2h^2}{3!} + f_2^{(4)}\frac{2h^4}{5!} + f_2^{(6)}\frac{2h^6}{7!}\right)$$
$$- \frac{h}{45}\left(7f_0 + 32f_1 + 12f_2 + 32F_3 + 7f_4\right)$$
$$= h\left(2f_2 + f_2^{(2)}\frac{2h^2}{3!} + f_2^{(4)}\frac{2h^4}{5!} + f_2^{(6)}\frac{2h^6}{7!}\right)$$
$$- \frac{h}{45}\Bigg[7\left(f_2 - \frac{h}{1!}f_2^{(1)} + \frac{h^2}{2!}f_2^{(2)} - \frac{h^3}{3!}f_2^{(3)} + \frac{h^4}{4!}f_2^{(4)} - \frac{h^5}{5!}f_2^{(5)} + \frac{h^6}{6!}f_2^{(6)}\right)$$
$$+ 32\left(f_2 - \frac{h}{1!2}f_2^{(1)} + \frac{h^2}{2!2^2}f_2^{(2)} - \frac{h^3}{3!2^3}f_2^{(3)} + \frac{h^4}{4!2^4}f_2^{(4)} - \frac{h^5}{5!2^5}f_2^{(5)} + \frac{h^6}{6!2^6}f_2^{(6)}\right)$$
$$+ 12f_2$$
$$+ 32\left(f_2 + \frac{h}{1!2}f_2^{(1)} + \frac{h^2}{2!2^2}f_2^{(2)} + \frac{h^3}{3!2^3}f_2^{(3)} + \frac{h^4}{4!2^4}f_2^{(4)} + \frac{h^5}{5!2^5}f_2^{(5)} + \frac{h^6}{6!2^6}f_2^{(6)}\right)$$
$$+ 7\left(f_2 - \frac{h}{1!}f_2^{(1)} + \frac{h^2}{2!}f_2^{(2)} - \frac{h^3}{3!}f_2^{(3)} + \frac{h^4}{4!}f_2^{(4)} - \frac{h^5}{5!}f_2^{(5)} + \frac{h^6}{6!}f_2^{(6)}\right)\Bigg]$$

Gathering like terms and simplifying we end up with the expression

$$I - I_{new} = -\frac{1}{15120}h^7 f_2^{(6)} + \mathcal{O}(h^8) \tag{4}$$

which implies

$$C = -\frac{1}{15120}, \quad d = 7, \quad s = 6$$

Since the degree fo exactness is 5 ,by Theorem 1 $I_{new} = \frac{7}{15}I_{sf} + \frac{8}{15}I_{o2}$ is a polynomial interpolation quadrature method. Also I_{new} is Newton Cotes method since it interpolates at uniform mesh points.

3.3 Romberg Method

The Romberg method is extrapolation of Trapezoidal rule with refined Trapezoidal. This leads to a nice pattern whic requires only three scalar operations

$$T_l^k = \frac{4^k T_{2l}^{k-1} - T_l^{k-1}}{4^k - 1} \tag{5}$$

To understand the algorithm better let us apply Romberg's method for $k = 3$. The algorithm first computes the Trapezoidal rule with refinements and populates the vector R with these results. Then the algorithm uses the relation to compute and store the diagonals of the Romberg triangle. A graphical evolution of the algorithm for $k = 3$ is shown below. The triangle is easily seen.
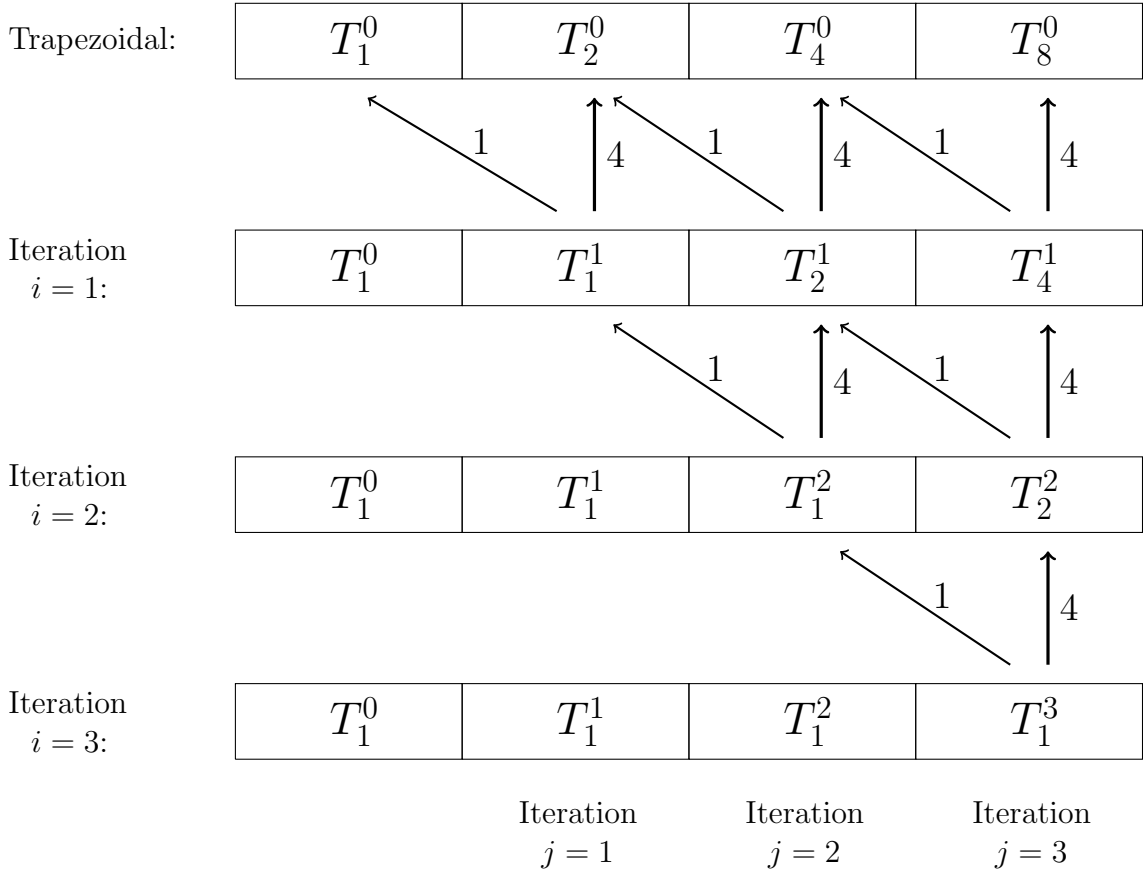
This algorithm is not interpolatory but has some nice features about it

1. $I(f) - T_l^k = \mathcal{O}(h_l^{2k+2})$

2. Each column has same order and converges as $h_l \to 0$

3. Order increases by 2 in the next column

4 Description of the Algorithm and Implementation

Inputs: Mesh X, Function $f(x)$, Max refinements N
Outputs: None
Storage: Mesh $X = [x_0, \ldots, x_n]$ is $\mathcal{O}(n)$, (Vector) Romberg results R
Time: For loop at line 3 is $\mathcal{O}(n)$, While loop at 6 is $\mathcal{O}(2^r nr)$, Everything else is $\mathcal{O}(1)$

Algorithm 1 Trapezoidal Method

1: $h \leftarrow x_1 - x_0, \ refinement \leftarrow 0, \ error \leftarrow \infty$
2: $I \leftarrow \frac{1}{2}(f(x_0) + f(x_n))$
3: **for** $i = 1 : 1 : n - 1$ **do**
4: $\quad I \leftarrow I + f(x_i)$
5: **end for**
6: $R_0 \leftarrow I * h$
7: **while** refinement $< N$ **do**
8: $\quad h \leftarrow \frac{h}{2}$
9: $\quad$ **for** $i = 1 : 1 : 2^{refinement}(n - 1)$ **do**
10: $\quad\quad I \leftarrow I + f(x_0 + (2i - 1)h)$
11: $\quad$ **end for**
12: $\quad R_{refinement} \leftarrow I * h$
13: $\quad$ refinement $\leftarrow$ refinement $+1$
14: **end while**
15: **return**

For the Trapezoidal algorithm the factor $2^r nr$ comes from the refinement. The $2^r n$ is the size of the new mesh at refinement iteration r. The code may seem to have a worst case of exponential but in fact it is just the size of a refined mesh, so it is still linear with respect to the size of the biggest mesh being evaluated.

Inputs: Mesh X, Function $f(x)$, Max refinements N
Outputs: None
Storage: Mesh $X = [x_0, \ldots, x_n]$ is $\mathcal{O}(n)$, (Vector) Romberg results R
Time: For loops at 2 and 3 are $\mathcal{O}(N^2)$, Everything else is $\mathcal{O}(1)$

Algorithm 2 Romberg Method

1: **trapezoidalMethod**$(X, N, f(x), R)$
2: **for** $i = 1 : 1 : N$ **do**
3: $\quad$ **for** $j = N : -1 : i$ **do**
4: $\quad\quad R_j \leftarrow (4^i R_j - R_{j-1})/(4^i - 1)$
5: $\quad$ **end for**
6: **end for**

---

5   Description of the Experiment Design and Results

For this program we use six functions to evaluate accuracy and error of our quadrature methods

$$f_1(x) = \int_0^3 e^x \, dx, \quad f_2(x) = \int_0^{\pi/3} e^{\sin(2x)} \cos(2x) \, dx, \quad f_3(x) = \int_{-2}^1 \tanh(x) \, dx$$

$$f_4(x) = \int_0^{3.5} x \cos(2\pi x) \, dx, \quad f_5(x) = \int_{0.1}^{2.5} x + \frac{1}{x} \, dx, \quad f_6(x) = \int_0^{\pi/4} \ln(\cos(x)) \, dx$$

where each function was chosen to highlight the differences between Trapezoidal refinement and Romberg's method.

6   Results

The error of the diagonal, T_1^k in Romberg's triangle is $E_{romberg} = \mathcal{O}(h^{2k+2})$. Taking the logarithm of both sides we get

$$\log(E_{romberg}) = (2k + 2) \log(\mathcal{O}(h)) \tag{6}$$

From this we expect $\log(E_{romberg})$ to scale quasi-linearly against k, with a slope of 2. The graphs below plot $-\log(E_{romberg})$ vs k and show this behavior, although some functions take longer to reach this behavior. It is also important to notice that the same was done with the Trapezoidal rule and refinement and its order is below that of Romberg's method, as we expected.

With Trapezoidal refinement, if we are moving from $I_n^0 \rightarrow I_{2n}^0$ it requires n more function evaluations, where the function can be computationally expensive to evaluate. However, with extrapolation it just requires three scalar operations and we get two more orders! Extrapolation is not only more cost effective when n is large, but it also converges to the analytical answer faster than refinement. This can be shown by looking at the tables below. However, some functions, such as function 4, may take a few iterations for Romberg to monotonically decrease in error and show the converging faster converging behavior with its low overhead computational cost.

6.1   Function 1

Iteration	Refinement Result	$error_{refinement}$	Romberg $T_1^{iteration}$	$error_{Romberg}$
1	31.628305384782	12.542768461594	31.628305384782	12.542768461594
2	22.536686297898	3.451149374710	19.506146602270	0.420609679082
3	19.971895038677	0.886358115490	19.091019153382	0.005482230194
4	19.308673108064	0.223136184876	19.085556071353	0.000019148165
5	19.141418847047	0.055881923859	19.085536940160	0.000000016972
6	19.099513540699	0.013976617511	19.085536923191	0.000000000004
7	19.089031461401	0.003494538213	19.085536923188	0.000000000000
8	19.086410581735	0.000873658548	19.085536923188	0.000000000000

Results and errors for Trapezoidal with refinements and the diagonal elements of
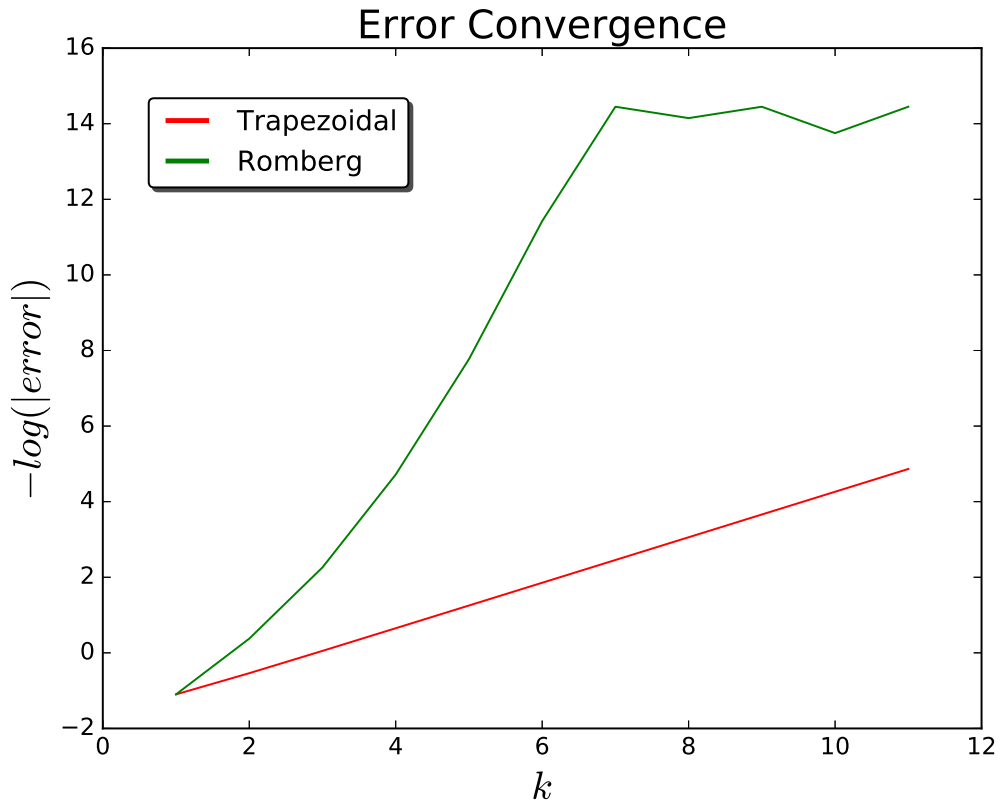Romberg's triangle.

Figure 1: Order of convergence for Trapezoidal with refinement and Romberg's method.

6.2 Function 2

Iteration	Refinement Result	$error_{refinement}$	Romberg $T_1^{iteration}$	$error_{Romberg}$
1	-0.098814261306	0.787535598924	-0.098814261306	0.787535598924
2	0.573005906251	0.115715431367	0.796945962104	0.108224624486
3	0.660309152200	0.028412185418	0.682241185655	0.006480151963
4	0.681666536161	0.007054801457	0.688847245837	0.000125908219
5	0.686960731814	0.001760605804	0.688720597268	0.000000740350
6	0.688281380503	0.000439957115	0.688721338853	0.000000001235
7	0.688611360499	0.000109977119	0.688721337618	0.000000000001
8	0.688693844098	0.000027493520	0.688721337618	0.000000000000

Results and errors for Trapezoidal with refinements and the diagonal elements of Romberg's triangle.
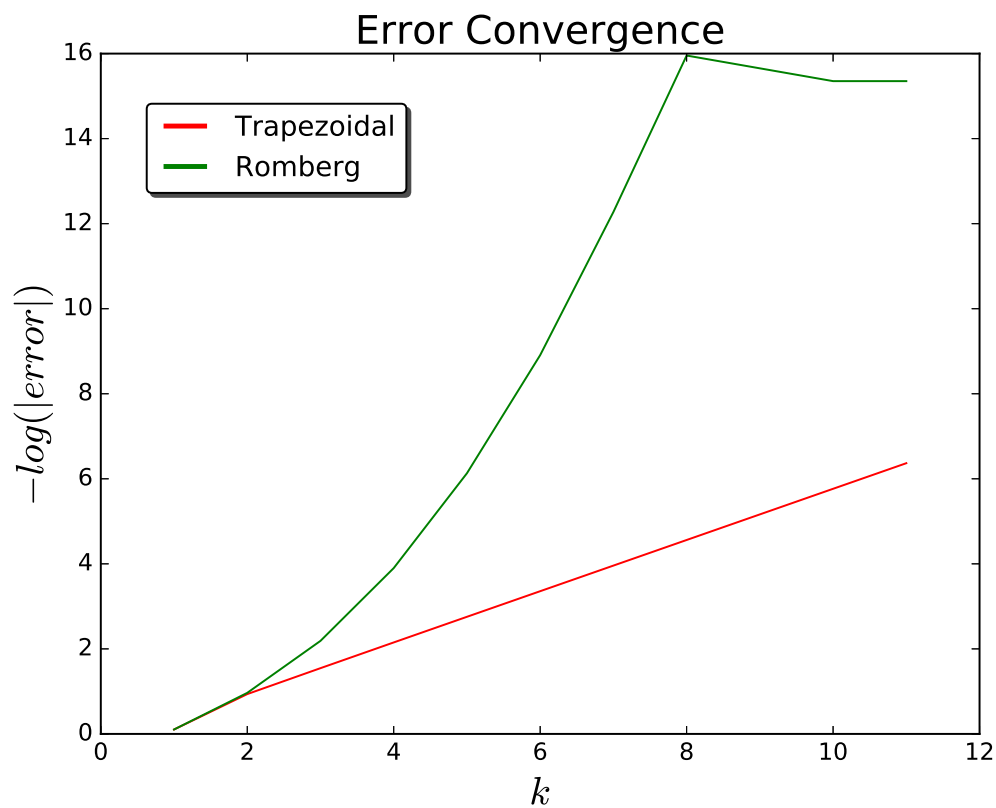

Figure 2: Order of convergence for Trapezoidal with refinement and Romberg's method.

6.3 Function 3

Iteration	Refinement Result	$error_{refinement}$	Romberg $T_1^{iteration}$	$error_{Romberg}$
1	-0.303650136180	0.587571780695	-0.303650136180	0.587571780695
2	-0.845000803980	0.046221112895	-1.025451026580	0.134229109705
3	-0.875024135155	0.016197781719	-0.875670637923	0.015551278952
4	-0.887138983329	0.004082933546	-0.891839596284	0.000617679409
5	-0.890199150345	0.001022766530	-0.891213764434	0.000008152441
6	-0.890966104414	0.000255812461	-0.891221954955	0.000000038080
7	-0.891157956305	0.000063960570	-0.891221916805	0.000000000070
8	-0.891205926268	0.000015990607	-0.891221916875	0.000000000000

Results and errors for Trapezoidal with refinements and the diagonal elements of
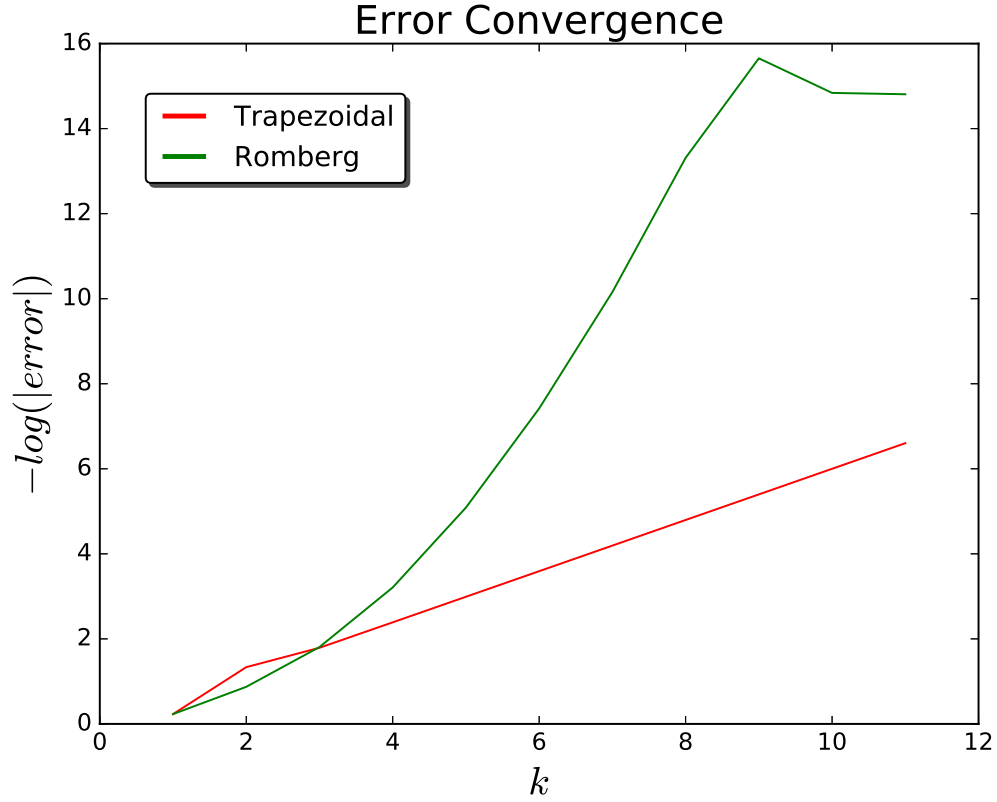Romberg's triangle.

Figure 3: Order of convergence for Trapezoidal with refinement and Romberg's method.

6.4   Function 4

Iteration	Refinement Result	$error_{refinement}$	Romberg $T_1^{iteration}$	$error_{Romberg}$
1	-6.125000000000	6.074339408179	6.125000000000	6.074339408179
2	-3.062500000000	3.011839408179	-2.041666666667	1.991006074845
3	-2.614007258692	2.563346666871	-2.492699212362	2.442038620541
4	-0.099489727275	0.048829135454	1.006910346653	1.057570938475
5	-0.059449605103	0.008789013282	-0.119498977791	0.068838385969
6	-0.052702385648	0.002041793827	-0.049662994963	0.000997596858
7	-0.051162002038	0.000501410217	-0.050664055098	0.000003463277
8	-0.050785389393	0.000124797572	-0.050660588882	0.000000002939

Results and errors for Trapezoidal with refinements and the diagonal elements of Romberg's triangle.
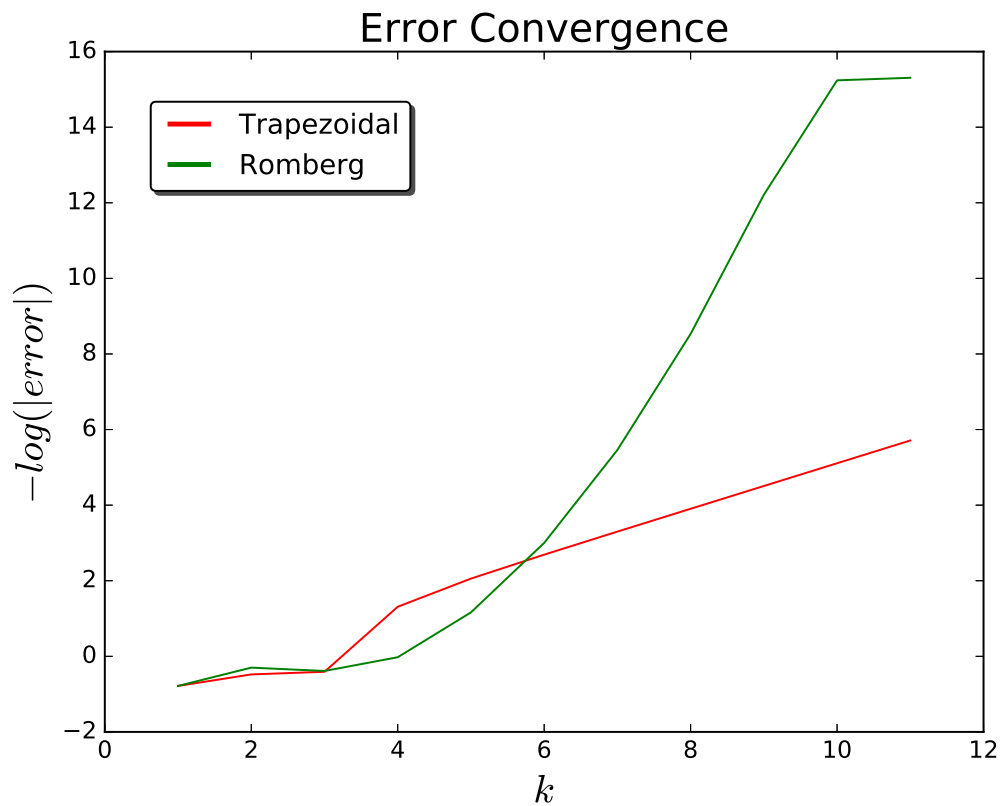

Figure 4: Order of convergence for Trapezoidal with refinement and Romberg's method.

6.5 Function 5

Iteration	Refinement Result	$error_{refinement}$	Romberg $T_1^{iteration}$	$error_{Romberg}$
1	15.600000000000	9.261124175132	15.600000000000	9.261124175132
2	10.283076923077	3.944201098209	8.510769230769	2.171893405901
3	7.874470792366	1.535594967497	6.975657605552	0.636781780684
4	6.871099032546	0.532223207678	6.493443158345	0.154567333477
5	6.501345240362	0.162469415494	6.364903703838	0.026027878969
6	6.383516684088	0.044640859220	6.341461860977	0.002586036109
7	6.350420744088	0.011544919220	6.339005937322	0.000130112454
8	6.341790691646	0.002914866778	6.338878739658	0.000002914790

Results and errors for Trapezoidal with refinements and the diagonal elements of
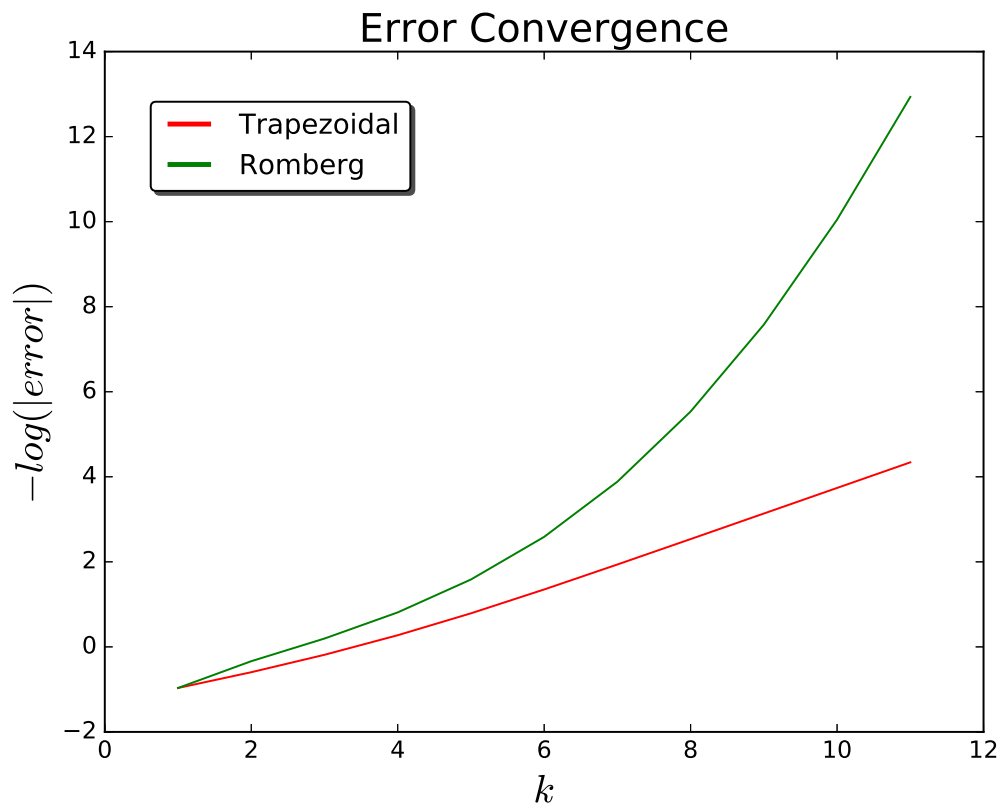Romberg's triangle.

Figure 5: Order of convergence for Trapezoidal with refinement and Romberg's method.

6.6   Function 6

Iteration	Refinement Result	$error_{refinement}$	Romberg $T_1^{iteration}$	$error_{Romberg}$
1	-0.136099130644	0.049685405157	-0.136099130644	0.049685405157
2	-0.099140962160	0.012727236673	-0.086821572665	0.000407847178
3	-0.089618374887	0.003204649399	-0.086419019560	0.000005294073
4	-0.087216402199	0.000802676712	-0.086413766931	0.000000041444
5	-0.086614490884	0.000200765397	-0.086413725635	0.000000000148
6	-0.086463922876	0.000050197388	-0.086413725488	0.000000000000
7	-0.086426275212	0.000012549725	-0.086413725487	0.000000000000
8	-0.086416862942	0.000003137455	-0.086413725487	0.000000000000

Results and errors for Trapezoidal with refinements and the diagonal elements of
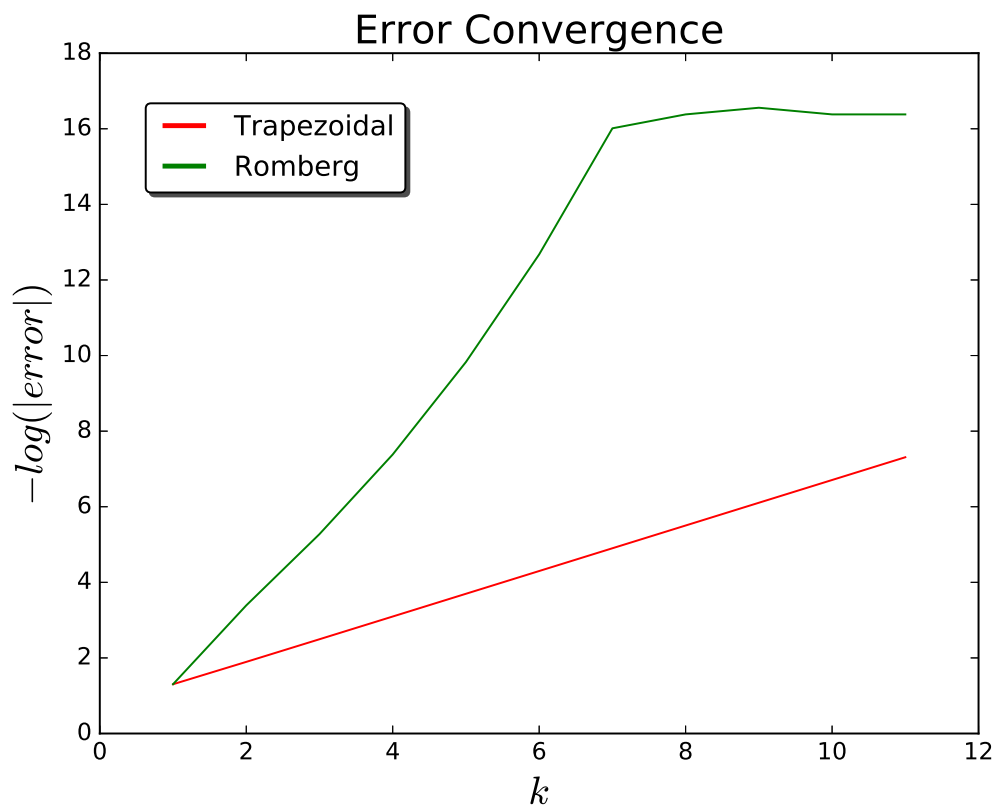Romberg's triangle.

Figure 6: Order of convergence for Trapezoidal with refinement and Romberg's method.

7   Conclusion

In this program we derived an analytical error form for the extrapolation methods $I_{new} = \frac{9}{5}I_{ss} - \frac{4}{5}I_{sf}$ and $I_{new} - \frac{7}{15}I_{sf} + \frac{8}{15}I_{o2}$ and showed that this method increases the order of infinitesimal, power of derivative, and degree of exactness. It was also shown that to be a Newton Cotes method it must interpolate at uniform mesh points. The former is not Newton Cotes but the latter is (both are interpolatory quadrature by theorem 1). We then when numerically investigated Trapezoidal method with refinement vs Romberg method with six integrals. It was shown that after some iteration Romberg method not only achieved higher order of accuracy, it did so with considerably lower computational cost. This leads to a very effecient way to achieve desired accuracy without low overhead.