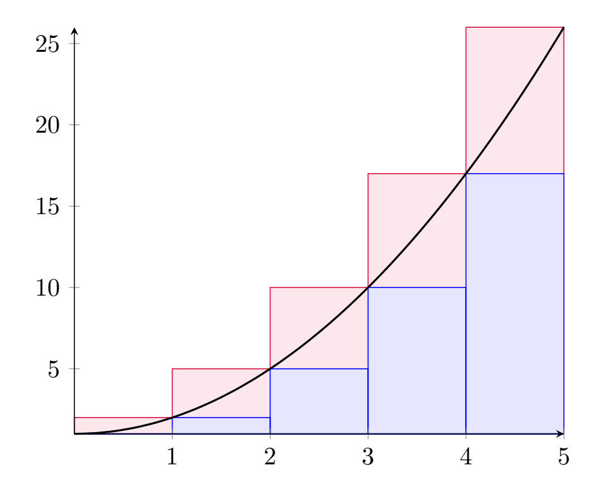
Hermite Numerical Integration

Jeff Rowell
Advisor: Dr.Henricus Bouwmeester
Metropolitan State University of Denver, Department of Mathematics & Computer Sciences



Methods

- Use of Trapezoidal rule for deriving the new Hermite_1 method using linear Hermite polynomials.
  - Includes derivative information only at the end points by using Hermite polynomials, "inner" derivatives telescope.

$$\begin{aligned}\int_{x_0}^{x_2} x^2dx &\approx \frac{h}{2}[\,x_0^2 + x_1^2\,] + \frac{h^2}{12}[\,2x_0 - \cancel{2x_1}\,] + \frac{h}{2}[\,x_1^2 + x_2^2\,] + \frac{h^2}{12}[\,\cancel{2x_1} - 2x_2\,] \\ &= \frac{h}{2}[\,x_0^2 + x_2^2 + 2\,x_1^2\,] + \frac{h^2}{12}[2x_0 - 2x_2\,]\end{aligned} \tag{1}$$

  - Generalize to a composite rule :

$$\int_{x_0}^{x_n} f(x)dx \approx \frac{h}{2}[\,f(x_0) + f(x_n) + 2\sum_{i=1}^{n-1} f(x_i)\,] + \frac{h^2}{12}[\,f'(x_0) - f'(x_n)\,] \tag{2}$$

  - Forward difference and backward difference schemes for approximating derivatives at the end points if derivative information is unobtainable.

$$\begin{aligned}f'(x_0) &\approx \frac{-50f(x) + 96f(x+h) - 72f(x+2h) + 32f(x+3h) - 6f(x+4h)}{24h} \\ f'(x_n) &\approx \frac{50f(x) - 96f(x-h) + 72f(x-2h) - 32f(x-3h) + 6f(x-4h)}{24h}\end{aligned} \tag{3}$$

  - We made these forward and backward difference schemes to be $\mathcal{O}(h^4)$ so that it is of the same order as our composite rule in (2).
- Use of Simpson's rule for deriving the new Hermite_2 method using quadratic Hermite polynomials this time.
  - Includes derivative information only at the end points by using Hermite polynomials, "inner" derivatives telescope.

$$\begin{aligned}\int_{x_0}^{x_4} x^5dx &\approx \frac{7h}{15}[\,x_0^5 + x_2^5 + x_2^5 + x_4^5\,] + \frac{16h}{15}[\,x_1^5 + x_3^5\,] + \frac{h^2}{15}[\,5x_0^4 - \cancel{5x_2^4}\,] + \frac{h^2}{15}[\,\cancel{5x_2^4} - 5x_4^4\,] \\ &= \frac{7h}{15}[\,x_0^5 + x_4^5\,] + \frac{14h}{15}[\,x_2^5\,] + \frac{16h}{15}[\,x_1^5 + x_3^5\,] + \frac{h^2}{15}[\,5x_0^4 - 5x_4^5\,]\end{aligned} \tag{4}$$

  - Generalize to a composite rule :

$$\int_{x_0}^{x_n} f(x)dx \approx \frac{7h}{15}[\,f(x_0) + f(x_n)\,] + \frac{14h}{15}\sum_{i=1}^{n-2} f(x_{2i}) + \frac{16h}{15}\sum_{i=1}^{n-1} f(x_{2i-1}) + \frac{h^2}{15}[\,f'(x_0) - f'(x_n)\,] \tag{5}$$

  - Forward difference and backward difference schemes for approximating derivatives at the end points if derivative information is unobtainable. We chose the difference schemes below to be of same order as the formula in (5), which is $\mathcal{O}(h^6)$.

$$\begin{aligned}f'(x_0) &\approx \frac{-274f(x) + 600f(x+h) - 600f(x+2h) + 400f(x+3h) - 150f(x+4h) + 24f(x+5h)}{120h} \\ f'(x_n) &\approx \frac{274f(x) - 600f(x-h) + 600f(x-2h) - 400f(x-3h) + 150f(x-4h) - 24f(x-5h)}{120h}\end{aligned} \tag{6}$$

- Use of cubic Hermite polynomials to see if "inner" derivatives will telescope for any order Hermite polynomial.
  - Requires derivative information at the "inner" points, derivatives do not telescope here.
- We chose to test our methods and compare them to the classical rules using the following integral :

$$\int_{-1}^{5} x \cdot sin(3x)dx = 1.6840782256556266186 \tag{7}$$

Abstract

The Hermite numerical integration rules are designed to exceed the classical numerical integration strategies such as the Trapezoidal and Simpson's rules, by reducing the number of computations required while attaining improved accuracy and efficiency. This research utilizes Hermite polynomials to derive two new numerical integration rules, rather than using Lagrange polynomials where the Trapezoidal and Simpson's methods originate. The difference between Lagrange polynomials and Hermite polynomials is the later include derivative information, whereas the former do not. The goal of this study will focus on the computational performance of the Trapezoidal and Simpson's techniques compared to the new rules we derived that use derivative information evaluated at the endpoints of the function. To do this we will deduce two separate integration rules from two different Hermite polynomials, such that one will be used for comparisons against the Trapezoidal rule, and the other will be used to compare against Simpson's rule. Our main difference is to include derivative information in our derivations. However, it is possible we cannot obtain derivative information at the endpoints of our function, so we will approximate the derivative using finite difference formulas. Our comparisons entail computational cost, number of operations, and the accuracy of the numerical integration techniques. Using Python we repeat an integral calculation 30,000 times with each rule, and then take the average time elapsed for each of the numerical integration rules to execute. We show that these derivative based methods are far superior to the classical methods of numerical integration.

Results

Table 1: Hermite_1 Error vs. Hermite_1 Finite Difference Error vs. Trapezoid Error

	Hermite_1		Hermite_1_FD		Trapezoid	
Tolerance	n	Error	n	Error	n	Error
5.0×10^{-4}	26	4.293426e-04	31	4.467123e-04	286	4.978732e-04
5.0×10^{-5}	45	4.752932e-05	38	3.752013e-05	903	4.994046e-05
5.0×10^{-6}	79	4.992647e-06	82	4.810067e-06	2854	4.999400e-06
5.0×10^{-8}	250	4.973470e-08	253	4.931378e-08	28539	4.999747e-08
5.0×10^{-10}	790	4.987344e-10	791	4.986906e-10	285385	4.999774e-10
5.0×10^{-12}	2497	4.997114e-12	2497	4.999778e-12	2859229	4.932943e-12

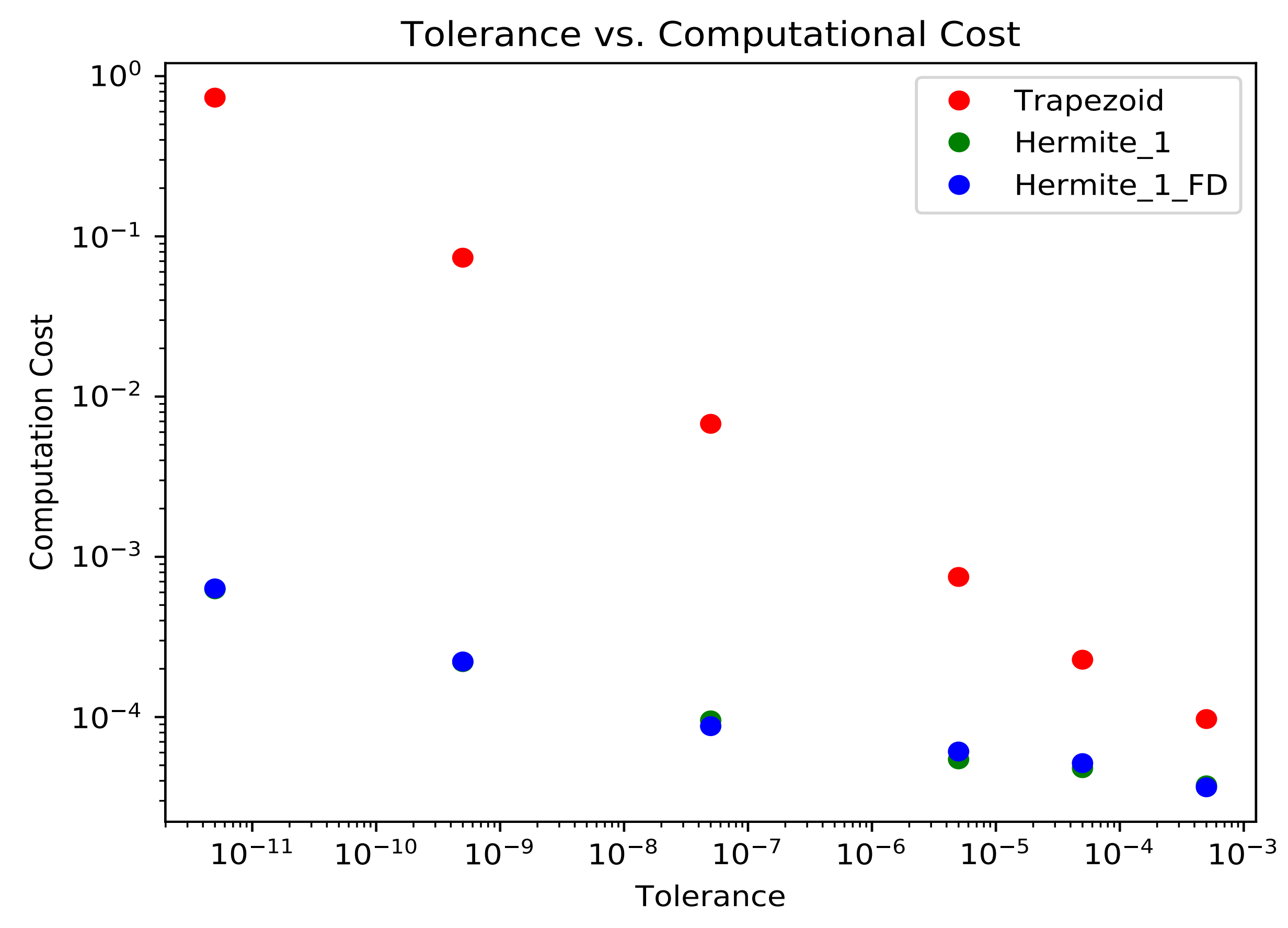
Table 2: Hermite_2 Error vs. Hermite_2 Finite Difference Error vs. Simpson's Error

	Hermite_2		Hermite_2_FD		Simpson's	
Tolerance	n	Error	n	Error	n	Error
5.0×10^{-4}	16	2.854510e-04	20	2.193576e-04	36	4.748412e-04
5.0×10^{-5}	22	3.947664e-05	44	4.280798e-05	64	4.669534e-05
5.0×10^{-6}	32	4.010167e-06	62	4.428051e-06	112	4.951322e-06
5.0×10^{-8}	68	4.241489e-08	120	4.526472e-08	354	4.949212e-08
5.0×10^{-10}	144	4.676086e-10	228	4.844649e-10	1118	4.973657e-10
5.0×10^{-12}	308	4.876544e-12	434	4.941381e-12	3532	4.994893e-12
5.0×10^{-16}	1054	4.440892e-16	1220	4.440892e-16	249994	4.440892e-16

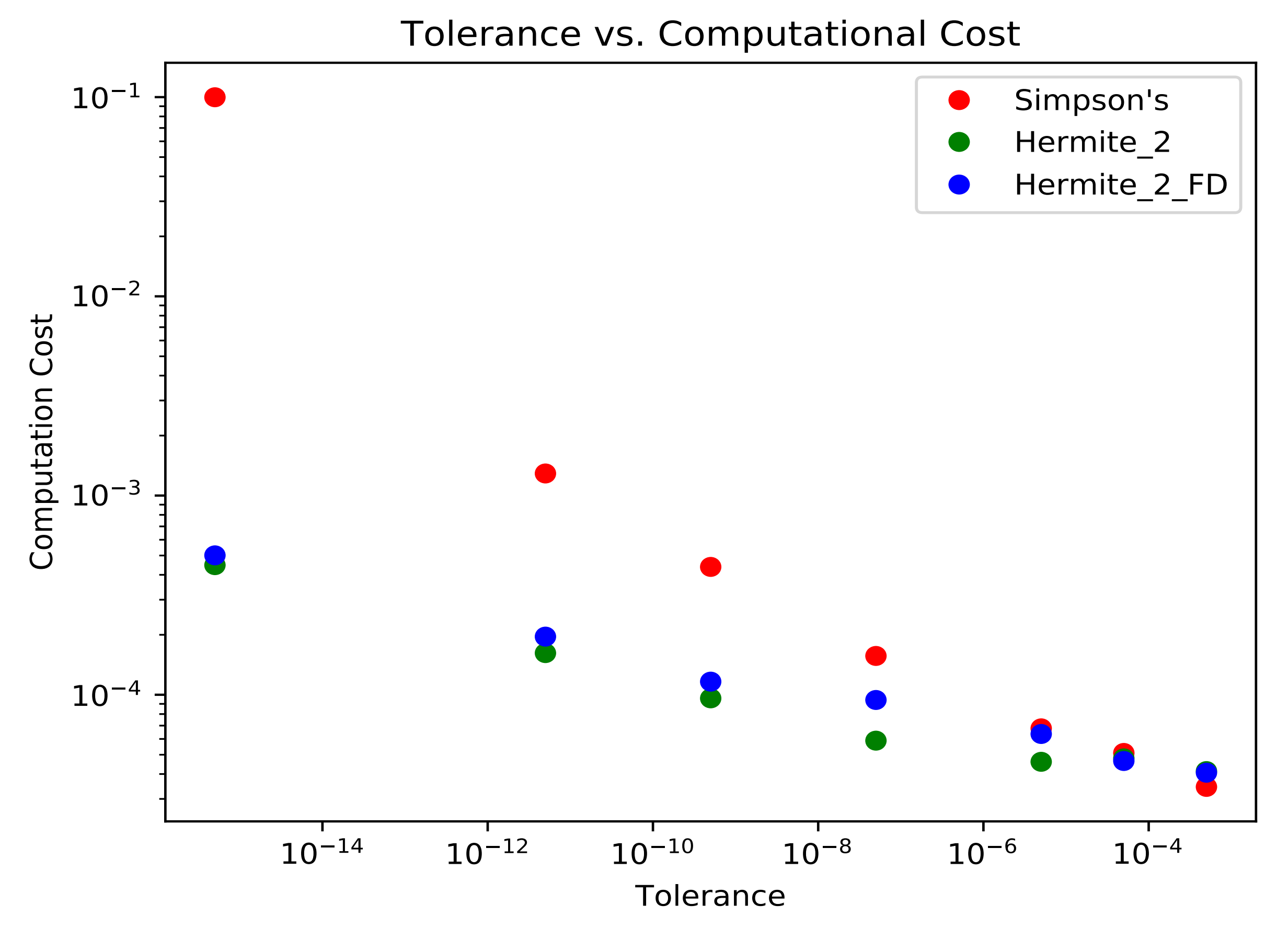
Table 1 shows the number of subintervals required to meet desired accuracies for our Hermite_1 rule versus the Hermite_1 Finite Difference rule versus the classical Trapezoidal rule. The columns of Table 1 and Table 2 are the methods used, and the rows are the absolute errors and number of subintervals. We decided to show the lowest number of subintervals required by each method to show the difference in computations required by each rule to achieve the specified accuracies. At tolerance 0.5×10^{-12} our Hermite_1 and Hermite_1_FD rules require the same number of subintervals, which is about 1,000 times less than the number of subintervals required by the Trapezoidal rule. In Table 2, at tolerance 0.5×10^{-16} our Hermite_2 and Hermite_2_FD rules require just about the same number of subintervals, which is roughly 235 times less than the number of subintervals required by Simpson's rule. Notice that for tolerance 0.5×10^{-16} the absolute errors are the exact same number for each method in Table 2. The absolute error for tolerance 0.5×10^{-16} is a special number because $4.440892e - 16$ is 2 times machine precision using double-precision floating-point arithmetic.

Conclusion

- Hermite_1
  - We benchmarked our Hermite_1 rules against the Trapezoidal rule as shown in the following plot. The independent axis is the absolute error, and the dependent axis is the average time in seconds for each method to execute.



- Hermite_2
  - In the plot below we benchmarked our Hermite_2 rules against the classical Simpson's rule. The independent axis is the absolute error, and the dependent axis is the average time in seconds for each method to execute.



- The above plots tell us how computationally expensive it is for each method studied in this research to reach a specified accuracy. We conclude that our Hermite_1 and Hermite_2 rules are far superior to the traditional numerical integration techniques.

Acknowledgements

I would like to acknowledge Dr. Henricus Bouwmeester for his patient help, guidance and encouragement throughout this research project. I consider myself very lucky to have had a research advisor who genuinely cares about my research and work, and who immediately got back to any questions I had throughout the process. I would also like to thank Dr. Andrew Bonham from the Department of Chemistry for helping print my poster.

Future Research

Even though the "inner" derivatives did not telescope with cubic Hermite polynomials, we would like to explore and see if the derivatives telescope when using Hermite polynomials of order 4. Additionally, Gaussian Quadrature is the next numerical integration strategy we would like to compare against. To do this we will derive an open numerical integration formula to warrant a fair comparison as Gaussian Quadrature is an open rule.