1 Attribution

Numeric integrations NBs.

2 Timekeeping

Approx values

Sunday: 2 hours
I defined most of the functions I would need for the project and implemented the Trapezoid rule. Then I set up LaTeX.

Monday: 0 hours

Tuesday: 1.25 hours
I worked on figuring out how to use github.

Wednesday: 2.5 hours
I implemented the Gaussian quadrature and plotted the Legendre polynomials.

Thursday: 1 hour
Write up.

3 Languages, Libraries, Lessons Learned

I wrote all of my code in python from start to finish. I tried my best to mostly work in code.py so it would be easier to interpret my github commits, but I also used Code_NB.py for the quality of life that a notebook offers when it comes to things like debugging and running a segment of code at a time. I also implemented various math functions from the numpy library such as np.sin, np.abs (absolute value), and np.sum. I used pylab for plotting. Both of these libraries are pretty standard because of their usefulness and simplicity, and I was already fairly familiar with them. In addition, I used the scipy library as instructed for Legendre polynomials. I would consider this pretty remarkable as I did not have to calculate them myself which would be a pain. Meanwhile, numpy and pylab are pretty remarkable in their own way— they come in handy very often.

4 Figures

5 Extensions

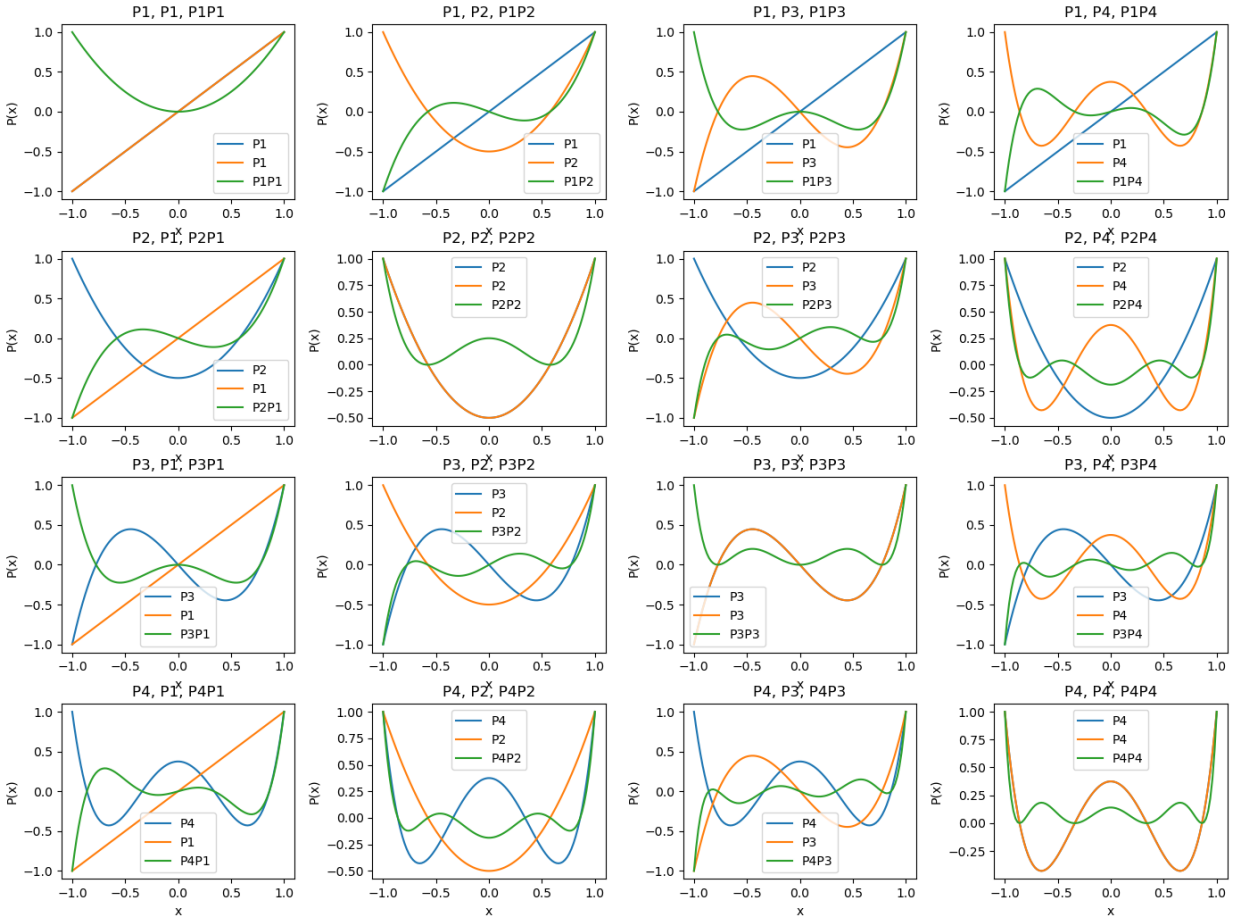

Figure 1: There are 16 subplots of Legendre polynomials P_i, P_j, and $P_i \cdot P_j$ for i and j in the range 1-4. The area under the curve can be observed for each of the i=j subplots on the diagonal as well as the i $\neq$ j subplots off diagonal. The areas under the i = j curves look to be approximately 1. Meanwhile, the areas under the i $\neq$ j seem to be about 0. This is the visualization of the integral of $P_i \cdot P_j$ from -1 to 1 equals the kroenecker delta function.