

APPM 5720 Homework 3

Wil Boshell, Fortino Garcia, Alex Rybchuk, Parth Thakkar

February 12, 2018

1 Preliminaries

Consider a grid with $N + 1$ grid points $x_L = x_0 < x_1 < x_2 < \dots < x_N = x_R$, defining N elements $\Omega_i = \{x \in [x_{i-1}, x_i]\}$, $i = 1, \dots, N$. We seek to approximate a function $f(x)$ on $x \in [x_L, x_R]$ by L_2 projection onto the space of element-wise Legendre polynomials of degree q . This approximation takes the form

$$\int_{\Omega_i} P_l(r) \sum_{k=0}^q c(k, i) P_k(r) dx = \int_{\Omega_i} P_l(r) f(x) dx, \quad l = 0, \dots, q.$$

where $c(i, k)$ are the coefficients of the L_2 projection. Here $r \in [-1, 1]$ is a local variable such that on element Ω_i the affine map satisfies $x(-1) = x_{i-1}$, $x(1) = x_i$. This affine map (explicitly) is

$$x(r) = \frac{1}{2} ((x_i - x_{i-1}) r + (x_i + x_{i-1})) \implies dx(r) = \frac{1}{2} (x_i - x_{i-1}) dr.$$

Since the Legendre polynomials are orthogonal with respect to the weight functions $w(x) = 1$ and

$$\int_{-1}^1 P_l(x) P_n(x) dx = \frac{2}{2n+1} \delta_{l,n},$$

the above system of equations reduces to

$$\begin{aligned} \int_{\Omega_i} P_l(r) \sum_{k=0}^q c(k, i) P_k(r) dx &= \sum_{k=0}^q c(k, i) \int_{\Omega_i} P_l(r) P_k(r) dx \\ &= \sum_{k=0}^q \frac{1}{2} c(k, i) (x_i - x_{i-1}) \int_{-1}^1 P_l(r) P_k(r) dr \\ &= \frac{1}{2} c(l, i) (x_i - x_{i-1}) \frac{2}{2l+1}. \end{aligned}$$

On the right hand side of our formulation, we approximate this using the Gauss-Lobato-Legendre weights and nodes w_j, r_j to obtain

$$\int_{\Omega_i} P_l(r) f(x) dx = \frac{1}{2} (x_i - x_{i-1}) \int_{-1}^1 P_l(r) f(x(r)) dr \approx \frac{1}{2} (x_i - x_{i-1}) \sum_{j=0}^q w_j P_l(r_j) f(x(r_j)).$$

We note that both sides have $1/2 (x_i - x_{i-1})$ so that our system of equations reduces to

$$c(l, i) \frac{2}{2l+1} = \sum_{j=0}^q w_j P_l(r_j) f(x(r_j)) \implies c(l, i) = \frac{2l+1}{2} \sum_{j=0}^q w_j P_l(r_j) f(x(r_j)), \quad l = 0, \dots, q.$$

That is, the set of N decoupled linear systems of equations of size $(q+1) \times (q+1)$ forms a diagonal matrix which can easily be solved without calling a routine that performs Gaussian elimination.

2 Increasing Element Count

We begin by considering a set of functions $f(x)$ and showing that with $q = 1, 2, \dots$, the approximation described above is increasingly accurate with an increasing number of elements as measured in the uniform and L_2 norm. In the following plots, note that $q = 1$ is in red and increasing q is colored according to the usual ordering of the color spectrum (i.e. ROYGBV applies).

2.1 Smooth, Infinitely Differentiable Function

Let us first consider the function

$$f(x) = e^{(x-2)^2}, \quad x \in [0, 4].$$

We note that this function is continuous and infinitely differentiable, and so we expect that our approximation scheme should increase in accuracy as we increase q and the number of elements. Indeed, this appears to be the case as demonstrated by the following plot of data. Consistently it appears that for a fixed number of elements an increase in q decreases the error, and for fixed q an increase in number of elements (i.e. smaller average element sizes) decreases the overall error as expected.

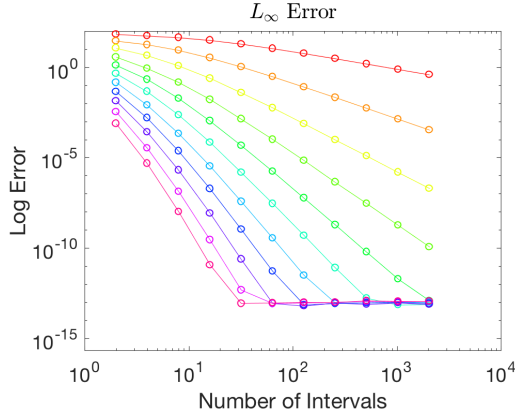


Figure 1: Max Error for an Exponential

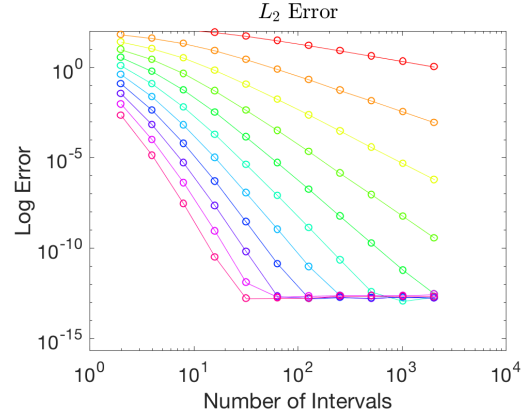


Figure 2: L_2 Error for an Exponential

The situation is much the same with respect to the L_2 error as depicted above in the right panel. We see that the L_2 error doesn't reach machine precision and is slightly worse compared to the uniform error.

2.2 Smooth, Infinitely Differentiable, Periodic Function

We move on to another infinitely differentiable function, this time to the periodic function

$$f(x) = \sin x, \quad x \in [-\pi, \pi].$$

Below is a plot of the error.

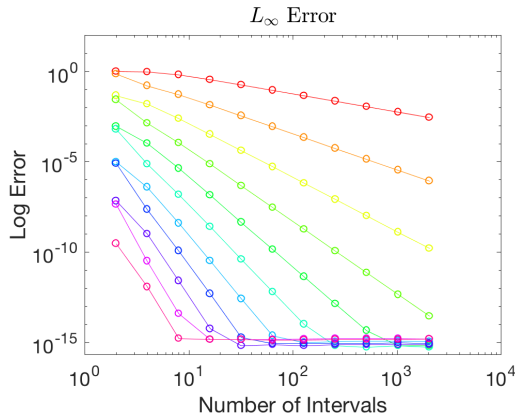


Figure 3: Max Error for $\sin x$

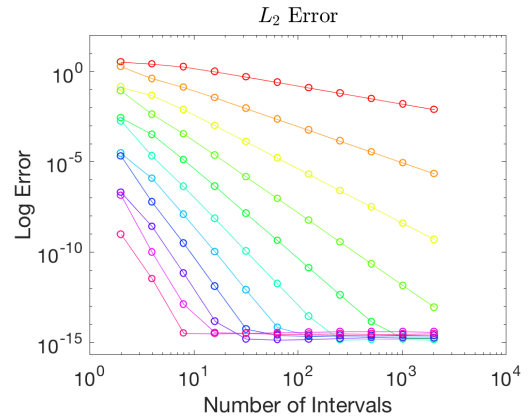


Figure 4: L_2 Error for $\sin x$

As before, the L_∞ error for each q is better than that of the L_2 error and decreases with increasing number of intervals used to build our approximation. One curiosity here is that it appears that choosing a slightly smaller q and increasing the number of intervals outperforms large q values past around 100 intervals.

2.3 Continuous, Non-Differentiable Function

So far we have only considered smooth, differentiable functions. Let us now examine a case of a continuous function that is not continuously differentiable,

$$f(x) = |x|, \quad x \in [-2, 2].$$

We might expect that the error away from the non-differentiable point $x = 0$ to be incredibly low since away from $x = 0$ the function is linear and for $q \geq 2$ we expect to approximate the function exactly (since choosing a quadrature with n nodes gives us accurate integration of polynomials of degree up to $2n - 3$ for Gauss-Lobato nodes). However, since $x = 0$ is in the region of interest, the error of approximating near the problem point will dominate the overall error in approximating the function globally.

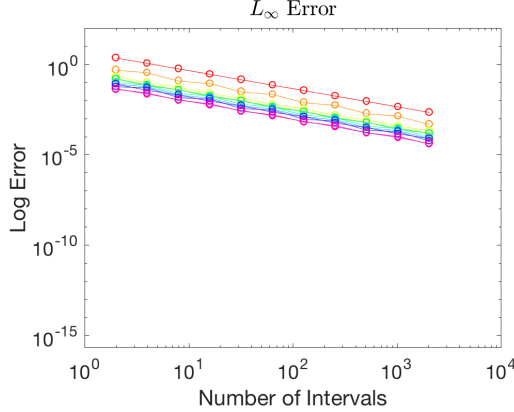


Figure 5: Max Error for $|x|$

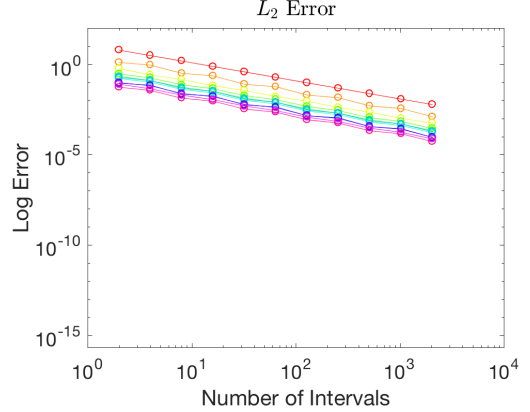


Figure 6: L_2 Error for $|x|$

The above error plot shows that, as before, increasing q and number of intervals used does decrease the overall error (and the L_∞ error is better than the L_2 error). In this case, however, we make few gains and don't achieve more than 5 digits of accuracy. The given formulation then is problematic for functions that are non-differentiable, even for a function that is not differentiable at a single point.

3 Increasing Polynomial Degree

4 Exploring Curiosities

Let us consider the following function

$$f(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$

We note that this function is not differentiable in the usual sense, nor is it even continuous. We then expect the prescribed approximation method to break down due to the jump discontinuity.

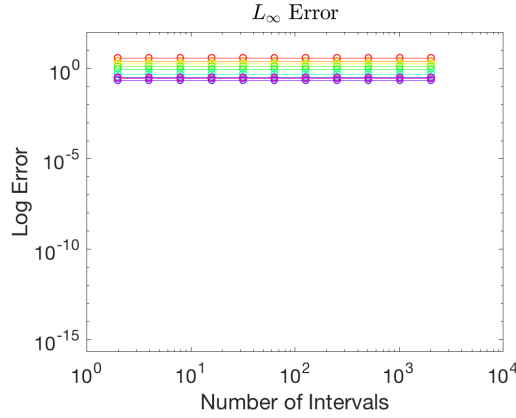


Figure 7: Max Error for $Sgn(x)$

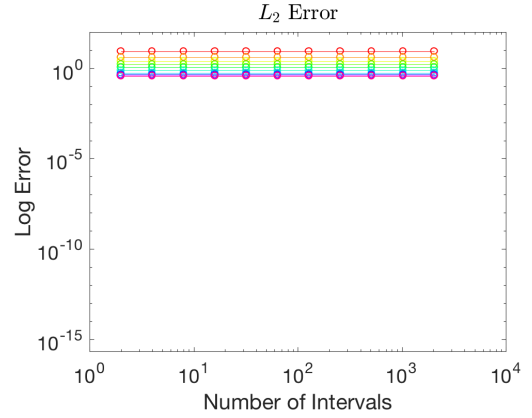


Figure 8: L_2 Error for $Sgn(x)$

Interestingly, the error does not decrease with increasing number of intervals used in our approximation. This suggests that the jump discontinuity cannot be well approximated by our method. As before, increasing q in general does decrease the overall error, but no more than a single digit of accuracy is attained for $q \leq 20$.