APPM 4600 Lab 10
Building L^2 approximations

1 Overview

In this lab, you will build a code that allows you to build L^2 approximations. You will use quadrature algorithm that is built into `SCIPY`.

2 Before Lab

1. The Legendre polynomials can be evaluated via the following three term recursion:

$$\begin{aligned}\phi_0(x) &= 1\\ \phi_1(x) &= x\\ \phi_{n+1}(x) &= \frac{1}{n+1}\left((2n+1)x\phi_n(x) - n\phi_{n-1}(x)\right)\end{aligned}$$

   Write a subroutine named `eval_legendre` that takes in an order n and value x where the polynomials are to be evaluated at and returns a vector $\mathbf{p}$ of length $n+1$ whose entries are the values of the Legendre polynomials at x.

   Note that due to the recursion formula, the values of $P_n(x)$ require you to compute P_j from $j = 0, 1, \ldots, n-1$. We might want to save those to be more efficient. If you have time, write down a routine that, given n and a vector of m values $\mathbf{x}$, returns a matrix $\mathbf{P}$ of size $(n+1) \times m$, with $P[i, j] = P_i(x_j)$.

3 Lab Day: Building the L^2 approximations

During lab, you will write a code that evaluates L^2 approximations of functions.

3.1 Creating the L^2 approximation

Recall from class that the polynomial of degree n ($p_n(x)$) that approximates a function $f(x)$ with respect to a weight function $w(x) \geq 0$ on an interval I is given by

$$p_n(x) = \sum_{j=0}^{n} a_j \phi_j(x)$$

where

$$a_j = \frac{\langle \phi_j, f\rangle_{L^2_w}}{\langle \phi_j, \phi_j\rangle_{L^2_w}} = \frac{\int_I \phi_j(x) f(x) w(x) dx}{\int_I \phi_j^2(x) w(x) dx}$$

and $\phi_j(x)$ are a set of polynomials orthogonal on I with respect to $w(x)$.

3.2 Exercises

1. Using `scipy.integrate.quad`, create a one line code that evaluates a coefficient a_j. Note: you have to create a subroutine that evaluates the $f(x)\phi_j(x)w(x)$ to feed into this, and also a subroutine $\phi_j^2(x)w(x)$ for evaluating the normalization. You should not use any symbolic packages.

   You may want to use the following to import the package:
   `from scipy.integrate import quad`

2. Take the method you developed in the prelab and the coefficient evaluator in problem 1 and insert them into the partially completed code legendre_expansion.py, to be found in the Modules tab on Canvas.

3. Change the function being approximated to $f(x) = \frac{1}{1+x^2}$. Does the accuracy of the approximation change? If so, how so?

3.3 Additional Exercises

As an additional exercise write a new code that creates an L^2 approximation using the Chebychev polynomials. They are also defined on the interval $[-1, 1]$ but the weight function is different

$$w(x) = \frac{1}{\sqrt{1-x^2}}.$$

The three term recursion is defined as follows

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= 2xT_n - T_{n-1}(x) \end{aligned}$$

Note that the code is the same except for: 1- You need a T_n evaluator and 2- write a new function that gets called in the coefficient evaluator.

3.4 Deliverables

All codes should be pushed to git and your responses to the questions should be entered into Canvas.