Homework 9
APPM 4600 Numerical Analysis, Fall 2025

Due date: Saturday, November 1, before midnight, via Gradescope. **Instructor**: Prof. Becker
Revision date: 10/25/2025

Theme: approximation and orthogonal polynomials.

Instructions Collaboration with your fellow students is OK and in fact recommended, although direct copying is not allowed. The internet is allowed for basic tasks (e.g., looking up definitions on wikipedia) but it is not permissible to search for proofs or to *post* requests for help on forums such as `http://math.stackexchange.com/` or to look at solution manuals. Please write down the names of the students that you worked with. Please also follow our AI policy.

An arbitrary subset of these questions will be graded.

Turn in a PDF (either scanned handwritten work, or typed, or a combination of both) to **Gradescope**, using the link to Gradescope from our Canvas page. Gradescope recommends a few apps for scanning from your phone; see the Gradescope HW submission guide.

We will primarily grade your written work, and computer source code is *not* necessary (and you can use any language you want). You may include it at the end of your homework if you wish (sometimes the graders might look at it, but not always; it will be a bit easier to give partial credit if you include your code). For nicely exporting code to a PDF, see the APPM 4600 HW submission guide FAQ.

Problem 1: Orthogonal polynomials Consider the interval $[-1, 1]$ and the weight function $w(x) = |x|$. Find an **orthogonal basis** of polynomials $\{\phi_k\}$ (with respect to the weighted inner product $\langle f, g\rangle = \int_{-1}^{1} w(x)f(x)g(x)\,dx$) for the space of polynomials of degree at most 4, $\mathcal{P}_4$.

Problem 2: Multivariate discrete ℓ_2 approximation

a) Download the csv dataset Homework/Data/HW09_multivariate_data.csv from our github website[1]. This has $m = 50$ rows, and 4 columns. The first column is the response variable y, and the next columns are variables x_1, x_2 and x_3. Do a multivariate polynomial regression of y with these variables, up to second order; that is, fit the data y, in the least-squares sense, to a polynomial of the form $p(\boldsymbol{x}) = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_1x_2 + a_5x_1x_3 + a_6x_2x_3 + a_7x_1^2 + a_8x_2^2 + a_9x_3^2$. Write down your polynomial, rounding each coefficient to two decimal places. *While for 1D we've emphasized using more clever bases, for this problem, it's fine to stick with the canonical basis.*

b) If we have d variables and a degree n polynomial (this means that we have terms like $x_1^{\alpha_1}x_2^{\alpha_2}\ldots x_d^{\alpha_d}$ where $\alpha_1 + \ldots + \alpha_d \le n$) then there are $\binom{n+d}{n} = \frac{(n+d)!}{n!d!}$ possible terms/coefficients[2]. For $d = 1$ this is just $n + 1$. In signal processing, statistical and data science applications, it's not unusual to have large dimensional datasets (e.g., one can think of a 2048×2048 image as a having $d = 2^{22} = 4194304$ variables). For each row in the table below, **calculate** the number of terms in a d-dimensional polynomial of degree n using the above formula. After doing this, **write a sentence** or two about implications of what you've found. *Hint: for large d but small n, you might want to simplify the formula by hand otherwise your computer may not be able to calculate the factorial.*

[1]Note: you can click this link, but some browser or canvas PDF viewers don't let you do that, in which case you can download the PDF first and then click the link; or just navigate our github website manually.
[2]There are some clever ways to derive this; a nice one is at https://mathoverflow.net/a/225963.

Dimension d	Degree n
3	10
100	4
10,000	3
2^{22}	2

Problem 3: Continuous L^2 approximation Let $f(x) = \sin(2.5x)$. Over the domain $[-1, 3]$, compute both

a) the best L^2 norm approximation of f, i.e., with respect to $\|f\|_2^2 = \int_{-1}^{3} f(x)^2 \, dx$, *(Hint: use Legendre polynomials)*, and

b) the best weighted L^2 norm approximation of f with weight $w(x) = \tilde{w}(\frac{1}{2}(x-1))$ for $\tilde{w}(x) = \frac{1}{\sqrt{x^2-1}}$, i.e., with respect to $\|f\|_w^2 = \int_{-1}^{3} w(x) f^2(x) \, dx$, *(Hint: use Chebyshev polynomials)*,

from among the set of polynomials of degree at most 5, $\mathcal{P}_5$. Turn in (1) a plot showing f and the two polynomial approximations (labeled), and (2) the coefficients (in the monomial basis) of both the polynomial approximations, with two decimal places for the coefficients. You are allowed to use a computer for both numerical and symbolic computation, e.g,. you may use Desmos / Mathematica / SymPy / etc.

Hints: See our github website Demos/Ch8_ContinuousL2.ipynb for some starter Python code that gets you part way. We suggest making full use of the numpy Polynomial class. We also recommend debugging your code by simplifying the problem before tackling the full problem:

- *Try this with f being a polynomial, since then your best approximation p should be exactly f, so you can check your work;*
- *Try on the domain $[-1, 1]$ before moving onto $[-1, 3]$ so that you don't have to do any transformations;*
- *Do the Legendre case before the Chebyshev case, so that you don't have to worry about the weight function.*

Problem 4: Continuous L^2 vs discrete ℓ^2 approximation For the same setup as the previous problem, but without weights (so only Legendre, no Chebyshev), sample the function f at both $m = 6$ and $m = 100$ equally spaced datapoints, and do a (discrete) least squares best fit, still among $\mathcal{P}_5$. *Hint: in Python, the `numpy.polynomial.Legendre.fit(...)` function should make this easy.*

a) Plot these two approximations along with the continuous L^2 approximation you found in the previous problem.

b) As $m \to \infty$, does the discrete ℓ^2 approximation converge to the continuous L^2 approximation?

c) If you were to do this with the Chebyshev polynomials, can you think of any issues that would complicate the discrete weighted ℓ^2 approximation converging to the continuous weighted L^2 approximation?