Week 8: Inner products and model fit

Amath 301

TA Session

Please start up a Matlab/Python instance

Today

0. Orthogonality revisited
1. What's an inner product?
2. A fun discovery on the Homework

Orthogonal functions and vectors: a crash course

Two functions are orthogonal (with respect to a weight function w) if

$$\langle f, g \rangle_w = \int_0^a f(x)g(x)w(x)dx = 0$$

Sometimes $w(x) = 1$. Other times, we are less lucky.

Two vectors are orthogonal (with respect to a weight matrix M) if

$$\langle x, y \rangle_M = x^T M y = 0$$

To obtain the "dot product", let M be an identity matrix, so $\langle x, y \rangle = x^T y$.

Supplemental slides on algebraic structure of inner products will be on Piazza.

The integral is the map the vector product is trying to be.

Your application will tell you which weight function/matrix to choose.

For HW6, $A = I$

Other choices include:

- M is a "Mass matrix" (Engineering).
- A related to $Ax = b$ for gradient decent proofs.
- $w(x) = \frac{1}{\sqrt{1-x^2}}$ on $[-1, 1]$ (Semicircle weights)
- $w(x) = e^{-\frac{x^2}{2}}$ (Hermite polynomials)
- $w(x) = 1$ (aka Legandre weight function)

Orthogonal polynomials

- Orthogonal systems are very useful. These (usually) are a well conditioned basis.
- Orthogonal polynomials with leading coefficient 1 (a "Monic polynomial") are unique with respect to a weight function.
- There exists **exactly one** family of monic (leading coefficient is 1) polynomials that are orthogonal with respect to a weight function.
- The "usual" basis: $1, x, x^2, x^3, \ldots$ are not orthogonal with respect to $w(x) = 1$! (Check this by computing $\langle 1, x^2 \rangle$ on $[-1, 1]$).
  This is the root of many, many curve fitting issues.

Orthogonal polynomials

Orthogonal polynomials are an infinite family of polynomials $\{p_n\}_{n=0}^{\infty}$ where p_n is a degree n polynomial and all p_j, p_k satisfy:

$$\langle p_j, p_k \rangle = \int_a^b p_j(x) p_k(x) w(x) dx = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases}$$

To within a normalization.

All orthogonal polynomial families can be specified by *some* 3-term recurrence:

$$x p_n(x) = a_n p_{n+1}(x) + b_n p_n(x) + c_n p_{n-1}(x)$$

In fact, there exists a 1 to 1 correspondence between the weight function w and the 3-term recurrence. Finding a way to go between w to a_n, b_n, c_n and back is an open problem over 100 years old (collect your tenured professorship if you can figure this one out).
The recurrence is usually how we compute the families of polynomials.

Orthogonal polynomials in math

Big applications in:

- Numerical Integration (Powerful quadrature rules)
- Numerical differentiation (Spectral methods)
- Harmonic oscilators (pendulums, quantum waves, digital signals)
- Probability theory
- Combinatorics
- Analytical solutions of ODEs (Chebyshev DE, Hemite DE, Legandre DE)
- Climate models (good basis for numerics)
- Give solutions to problems in E&M. $\frac{1}{|r-r_0|}$ expands nicely.

What about other orthogonal systems?

Sins and cosines of different periods are orthogonal with $w(x) = 1$. You can verify

$$\int_0^{2\pi} \cos\left(j\frac{x}{2}\right)\cos\left(k\frac{x}{2}\right)dx = \pi\begin{cases}1 & j = k\\ 0 & j \neq k\end{cases}$$

Same story with $\sin$

$$\int_0^{2\pi} \sin\left(j\frac{x}{2}\right)\sin\left(k\frac{x}{2}\right)dx = \pi\begin{cases}1 & j = k\\ 0 & j \neq k\end{cases}$$

- Finding a finite collection of objects that are orthogonal to each other is straightforward.
- Finding an infinite collection is much more difficult.
- HW6 (a) and (b) are designed around this.

What happens in a computer

Computers don't handle continuous objects well. Data only!

- Natural question: How well does the dot product resemble the integral if we discretize?
    - Depends on the data (not a good sign)
    - Compute

$$\langle 1, \sin(x) \rangle = 0$$

  Discretize onto 100 uniformly spaced points on $[0, 2\pi]$. Call these x and y. Take a dot product. What do you get?
    - What about:

$$\langle 1, \sin(99x) \rangle = 0$$

I promised we would break things. Replace $\sin$ with $\cos$ and repeat. Do you still get 0?

This isn't entirely fair to $\cos$. On the homework, we did try to normalize.
Divide both data by their norm to normalize.

Repeat. Do you still get 1 Do we finally get 0?

What's going on??

Ok, one more.

- Plot both pieces of data.
- Plot the difference between the two
- Does $\cos(99x)$ discretized look like a high-frequency wave?

We don't have enough data! Try it again with `10001` points.

Now let's explain what we're seeing.

We can check (e.g. Wolfram Alpha) that

$$\int_0^{2\pi} \cos(x)\cos(99x)dx = 0$$

But we're computing (call our two data vectors a and b)

$$a \cdot b = \sum_{j=0}^{n} \frac{1}{c_2} \cos\left(\frac{j}{n} 2\pi\right) \frac{1}{c_{198}} \cos\left(99 \frac{j}{n} 2\pi\right)$$

$$= \frac{1}{c_2} \frac{1}{c_{198}} \sum_{j=0}^{n} \cos\left(\frac{j}{n} 2\pi\right) \cos\left(99 \frac{j}{n} 2\pi\right)$$

$$= \frac{1}{c_2 c_{198}} \left(1 + \sum_{j=1}^{n} \cos\left(\frac{j}{n} 2\pi\right) \cos\left(99 \frac{j}{n} 2\pi\right)\right)$$

What does this right sum look like?

That's a right point Riemann sum!

$$
\begin{aligned}
&= \frac{1}{c_2 c_{198}} \left(1 + \sum_{j=1}^{n} \cos\left(\frac{j}{n} 2\pi\right) \cos\left(99 \frac{j}{n} 2\pi\right) \right) \\
&\approx \frac{1}{c_2 c_{198}} + \frac{1}{c_2 c_{198}} \frac{n}{2\pi} \int_0^{2\pi} \cos(x) \cos(99x) dx
\end{aligned}
$$

(Set aside convergence assumptions for now)

2 questions:

- How good is that approximation of the integral?
- Does $\frac{n}{c_2 c_{198}} \to 0$ as $n \to \infty$? $\to 1$?

Error bound for Rieman integral?

Bound is approximately:

$$\frac{1}{2}\frac{(b-a)^2}{n}f'(c)$$

for some $c \in [a, b]$. In our case, this bound is (approximately)

$$\approx \frac{(2\pi)^2}{2n}(100)$$

In practice, the approximation is quite decent and this bound is overly pessimistic.

This is a computaional class! Let's see how we actually do with calculating the tail.

Compute the over the $j = 1$ to n (don't include the first term in the sum).

If the sum is about 0 (as the integral is 0), we end up with

$$\approx \frac{1}{c_2 c_{198}}$$

This goes to 0 pretty slowly.

How slow is "pretty slow"? Let's go to the computer and fit a model

So $c_2 \approx a_2\sqrt{n}$ and $c_{198} \approx a_{198}\sqrt{n}$ for some constants a_2 and a_{198}. These are just rate constants, we don't really care about their value.