pset7

September 7, 2017

1 18.06 pset 7

Due Wednesday, April 5.

1.1 Problem 1

Refer to the orthogonal polynomials notebook from class for this problem.

In class, we defined an dot product $f \cdot g = \int_{-1}^{1} f(x)g(x)dx$ for functions on $x \in [-1, 1]$, and using this we showed how we could apply Gram–Schmidt to the polynomials $\{1, x, x^2, \ldots\}$ to find an orthogonal of polynomials $p_k(x)$, the Legendre polynomials.

1.1.1 part (a)

In class, I claimed that by performing the orthogonal projection of *any* function $f(x)$ onto these polynomials, we obtain the **least-square fit polynomial** on the interval $[-1, 1]$. In this problem, you will apply basic calculus to show that explicitly.

Suppose we have an orthonormal basis $q_0(x), q_1(x), q_2(x), q_3(x)$ for all degree ≤ 3 polynomials (the vector space $\mathcal{P}_3$). i.e. $q_i \cdot q_j = 0$ if $i \neq j$ and $= 1$ if $i = j$, using our dot product from above. Given a real-valued function $f(x)$ on $[-1, 1]$ (with finite $f \cdot f$ — none of these integrals should blow up!), we want to find the *closest* degree-3 polynomial to f in the least-square sense:

$$\min_{p \in \mathcal{P}_3} \int_{-1}^{1} [f(x) - p(x)]^2 dx = \min_{p \in \mathcal{P}_3} (f - p) \cdot (f - p) = \min_{p \in \mathcal{P}_3} \|f - p\|^2$$

Write $p(x)$ in our orthonormal basis:

$$p(x) = c_0 q_0(x) + c_1 q_1(x) + c_2 q_2(x) + c_3 q_3(x)$$

At the minimum p (the least-square fit), basic calculus tells us that the partial derivative must be zero:

$$\frac{\partial}{\partial c_k} \|f - p\|^2 = 0$$

Show that this leads to the condition $c_k = q_k \cdot f$, which is exactly the coefficient of the orthogonal projection.

Hint: You can easily verify that the product rule $\frac{\partial}{\partial c}(f \cdot g) = \left(\frac{\partial f}{\partial c} \cdot g\right) + \left(f \cdot \frac{\partial g}{\partial c}\right)$ works for dot products of functions!

1.1.2 part (b)

Suppose that we have real-valued function $f(x)$ that is in the span of an infinite orthonormal basis $q_k(x)$ of functions (e.g. polynomials as above) on $[-1, 1]$ with the dot product from above, i.e.

$$f(x) = \sum_{k=0}^{\infty} c_k q_k(x)$$

for coefficients $c_k = q_k \cdot f$. Assuming $\|f\|$ is finite (i.e. the function f is square-integrable), **derive the identity:**

$$\|f\|^2 = f \cdot f = \sum_{k=0}^{\infty} c_k^2$$

(This result is called Parseval's theorem for Fourier series.)

How does this relate to problem 4 of pset 6?

(For people who have taken 18.100 or similar: assume you can freely interchange/re-order the infinite sums, limits, integrals, etcetera; doing this properly would involve establishing some technical conditions on the infinite series here.)

1.2 Problem 2

Apply Gram-Schmidt to the polynomials $1, x, x^2$ to find an orthonormal basis of polynomials under the *different* dot product:

$$f \cdot g = \int_0^{\infty} f(x)g(x)e^{-x}dx$$

There are lots of ways to define dot products in practice, and in real applications the choice of dot product depends a lot on the problem you are solving. For example, one might want to the weight the errors differently at different points (here, weighting by e^{-x}) in a least-square fit.

1.3 Problem 3

(Based on Strang, section 6.2, problem 33.) Consider the following four 2×2 matrices, which have very similar-looking entries:

```
In [ ]: A = [ 3.  2.
              1.  4. ]
        B = [ 3.  2.
             -5. -3. ]
        C = [ 5.  7.
             -3. -4. ]
        D = [ 5.  6.9
             -3. -4.  ]
        display(A); display(B); display(C); display(D)
```

1.3.1 (a)

Compute each matrix to the **100th power** in Julia, e.g. compute A^{100} in Julia by `A^100`. The results should be very different!

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

1.3.2 (b)

All of these matrices are diagonalizable (can be written as $X\Lambda X^{-1}$ as in lecture), with two distinct eigenvalues λ. The function `eigvals(A)` computes the eigenvalues of A in Julia. Using the built-in `eigvals` function, compute the eigenvalues of these four matrices, and use them to **explain the results** you observed in part (a).

Note that the eigenvalues may be complex numbers, even for real matrices, just as the roots of a real polynomial may be complex! The complex number $z = a + bi$ in Julia is written `z = a + b*im`. Complex numbers can also be written in polar form $z = re^{i\theta}$, where `r = abs(z)` and θ `= angle(z)` in Julia. Recall that $z^n = r^n e^{in\theta}$ blows up if $|z| = r =$ `abs(z)` is > 1.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

1.4 Problem 4

1.4.1 (a)

Based on Strang, section 5.1, problem 8. Prove that every orthogonal matrix ($Q^T Q = I$) has determinant $+1$ or -1, in two ways:

- Use the product rule $\det(AB) = (\det A)(\det B)$ and the transpose rule $\det Q = \det Q^T$.
- Use only the product rule. If $|\det Q| < 1$ then $\det Q^n = (\det Q)^n$ goes to zero: Q^n becomes nearly singular for large n. How do you know that this can't happen to Q^n?
- Hint: $(Q^n)^T(Q^n) =$??? so Q^n is ???.
- Alternatively, think about problem 4 of pset 6, and note that a nearly singular matrix A has a vector $x \neq 0$ that is nearly in a nullspace (Ax is nearly zero).

1.4.2 (b)

If $\det A = 1$, does that mean that A is orthogonal? Explain why or provide a counterexample if it is false.

1.4.3 (c)

If $\det A = 1234$, what is $\det R$ where R is the upper-triangular matrix in the QR factorization of A?

1.5 Problem 5

1.5.1 (a)

The function `X = randn(5,5)` in Julia generates a random 5×5 matrix. Given X, we can compute a new matrix $Y = \alpha X$ for some scalar α such that $\det Y = 1234$. What is α?

In [ ]:
```
X = randn(5,5)
# to make things easier, I'll force det(X) to be positive by flipping the sign of the first col
if det(X) < 0
    X[:,1] = -X[:,1]
end
det(X)
```

In [ ]:
```
α = ???    # fill in this line!
Y = α * X
det(Y)     # this should give 1234 (+ small roundoff error)
```

1.5.2 (b)

Using your matrix Y, compute its QR factorization by `Q, R = qr(Y)` and use this to check your answer from problem 4(c) above.

```
In [ ]: Q, R = qr(Y)
```

```
In [ ]:
```