pset6-sol

September 7, 2017

1 18.06 pset 6 - Solutions

1.1 Problem 1

(Strang, chapter 4.2, problem 13.) Suppose A is the 4 by 4 identity matrix with the last column removed, so that A is 4 by 3. Project $b = (1, 2, 3, 4)$ onto $C(A)$. What shape is the projection matrix P? What is P?

1.1.1 Solution

The matrix A is

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

The span of the columns of A is just the 3-dimensional subspace $V = \{x_4 = 0\}$ of those vectors who have the last coordinate equal to 0, so the orthogonal projection onto V is done by making the fourth coordinate equal to zero. That is, the projection of b is $Pb = (1, 2, 3, 0)^T$.

Since the columns of A are orthonormal we can compute the projection matrix onto $C(A)$ by

$$P = AA^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

1.2 Problem 2

(Based on Strang, chapter 4.2, questions 21–23.) If A is full column rank, the projection matrix onto $C(A)$ is $P = A(A^TA)^{-1}A^T$. Show:

- $P^2 = PP = P$. (Pb is in $C(A)$, so projecting again does not change it.)
- $P^T = P$. That is, show that P is symmetric. (Remember that $(B^{-1})^T = (B^T)^{-1}$ for any invertible B.)
- If A is invertible, then show that $P = I$.

1.2.1 Solution

- We can derive $PP = P$ just by direct multiplication:

$$P^2 = A(A^TA)^{-1}A^TA(A^TA)^{-1}A^T = A(A^TA)^{-1}A^T = P$$

But of course, the underlying result should be obvious: once you project onto a subspace, additional projections don't need to do anything to stay in the subspace. Projecting twice must therefore be the same as projecting once:

- Again, we can show $P^T = P$ by direct computation:

$$P^T = (A(A^TA)^{-1}A^T)^{-1} = (A^T)^T((A^TA)^{-1})^TA^T = A((A^TA)^T)^{-1}A^T = A(A^TA)^{-1}A^T$$

  where we used the fact that the transpose of the inverse is the inverse of the transpose: $((A^TA)^{-1})^T = ((A^TA)^T)^{-1} = (A^TA)^{-1}$
- If A is invertible, there are several ways of seeing that $P = I$.
- If A is invertible then $(A^TA)^{-1} = A^{-1}(A^T)^{-1}$ and so $P = A(A^TA)^{-1}A^T = AA^{-1}(A^T)^{-1}A^T = I \cdot I = I$
- If A is an invertible $m \times m$ matrix, then $C(A) = \mathbb{R}^m$ (the whole space): *every* vector is in $C(A)$ already. So, projection doesn't need to do anything, and we must have $P = I$.
- P is invertible since it is a product of invertible matrices. But $P^2 = P$. So by multiplying by P^{-1} on both sides gives $P = I$.

1.3 Problem 3

(Strang, chapter 4.4, problem 4.) Give an example of:

- A matrix Q with orthonormal columns but $QQ^T \neq I$
- Two orthogonal vectors that are not linearly independent. (Not orthonormal!)
- An orthonormal basis for $\mathbb{R}^3$, including the vector $q_1 = (1, 1, 1)/\sqrt{3}$.

1.3.1 Solution

- This is generally true for any non-square Q, because $\text{rank}(QQ^T) = \text{rank}(Q^TQ) = \text{rank}(I) = n$, where Q is an $m \times n$ matrix with $m \geq n$. That means the $m \times m$ matrix QQ^T is not full rank unless $m = n$, and therefore it cannot equal I. As a specific example, take Q to be the matrix A of problem 1

$$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

  The columns of Q are orthonormal but as we have seen $QQ^T \neq I$.
- At least one of the vectors simply has to be zero, since that is orthogonal to everything but not linearly independent of anything (since it is any vector times 0). For example, the vectors $v_1 = (1, 0)^T$ and $v_2 = (0, 0)^T$ in $\mathbb{R}^2$. Clearly $v_1^Tv_2 = 0$ and so they are orthogonal, but $v_2 = 0v_1$ so they are not independent.
- We just need to start with any set $\{q_1, v_2, v_3\}$ of independent vectors that includes q_1, and then apply Gram-Schmidt to the other two vectors. For example:

$$v_2 = (1, -1, 0),\ v_3 = (0, 1, -1)\ .$$

  To save ourselves work, we conveniently chose these vectors to already be orthogonal to q_1. So, on the first Gram–Schmidt step we simply need to normalize v_2 to obtain q_2:

$$q_2 = \frac{v_2}{\sqrt{v_2^Tv_2}} = (1/\sqrt{2}, -1/\sqrt{2}, 0)^T\ .$$

  On the second Gram–Schmidt step, we have to remove the q_1 and q_2 components from v_2, but since $q_1^Tv_2 = 0$ already we just need to remove the q_2 component:

$$w_3 = v_3 - q_2q_2^Tv_3 = (0, 1, -1) - (-1/2, 1/2, 0) = (1/2, 1/2, -1)$$

  Finally we normalize this to a unit vector:

$$q_3 = \frac{w_3}{\sqrt{w_3^Tw_3}} = (1/\sqrt{6}, 1/\sqrt{6}, -2/\sqrt{6})$$

1.4 Problem 4

If Q is an $m \times n$ matrix with orthonormal columns then \$(Qx)ˆT (Qy) = \$???? for any x and y in $\mathbb{R}^n$, and hence $\|Qx\| =$???? for any x.

1.4.1 Solution

We have

$$(Qx)^T(Qy) = x^T Q^T Q y = x^T y$$

and in particular

$$\|Qx\| = \sqrt{(Qx)^T Qx} = \sqrt{x^T x} = \|x\|$$

1.5 Problem 5

The standard ("classical") Gram–Schmidt algorithm is notoriously susceptible to roundoff errors when vectors are nearly parallel. (There are much better algorithms that are not covered in 18.06.) In this problem, you will explore that susceptibility.

Apply the Gram–Schmidt algorithm to find an orthonormal basis for the column space of

$$A = \begin{pmatrix} 1 & 1 & 1 \\ \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & 0 & \epsilon \end{pmatrix}$$

where $\epsilon = 10^{-10}$.

However, *round* your calculations (as you do each operation!) to about 15 significant digits. For example, approximate $1 + \epsilon^2 \approx 1$. Are your resulting basis vectors close to orthogonal?

In comparison, Julia's `qr` function uses a much more accurate method ("Householder reflectors"). Try it and see how close its vectors are to the ones you came up with:

```
In [1]: ϵ = 1e-10
        A = [1 1 1
             ϵ 0 0
             0 ϵ 0
             0 0 ϵ]
        Q = qr(A)[1]

Out[1]: 4×3 Array{Float64,2}:
         -1.0        7.07107e-11   4.08248e-11
         -1.0e-10  -0.707107      -0.408248
          0.0        0.707107     -0.408248
          0.0        0.0           0.816497
```

We can check how close Julia's Q is to orthonormal columns by computing $Q^T Q$:

```
In [2]: Q'*Q

Out[2]: 3×3 Array{Float64,2}:
         1.0   0.0          0.0
         0.0   1.0         -1.66533e-16
         0.0  -1.66533e-16   1.0
```

This is **very close to the identity**, up to the usual 16th-digit roundoff errors, so the `qr` function is clearly doing a good job… much better than what you should get by applying Gram–Schmidt below.

1.5.1 Solution

Let v_1, v_2, v_3 be the columns of A. First we renormalize the first column. The length of the first column is

$$\sqrt{v_1^T v_1} = \sqrt{1+\epsilon^2} \approx 1$$

(since $1+\epsilon^2 \approx 1$). So

$$\boxed{q_1 = (1, \epsilon, 0, 0)}$$

is just the first column. Now let us subtract the projection of v_2 onto q_1:

$$w_2 = v_2 - q_2(q_2^T v_2) = (1, 0, \epsilon, 0) - (1, \epsilon, 0, 0) \times 1 = (0, -\epsilon, \epsilon, 0)$$

The length of w_2 is $\sqrt{w_2^T w_2} = \sqrt{2\epsilon^2} = \sqrt{2}\epsilon$ so after normalization we get

$$\boxed{q_2 = \frac{w_2}{\sqrt{2}\epsilon} = (0, -1, 1, 0)/\sqrt{2}}$$

Let now remove from v_3 the projections onto q_1 and q_2. Since $q_2^T v_3 = 0$, we just need to project out q_1:

$$w_3 = v_3 - q_1(q_1^T v_3) - q_2(q_2^T v_3) = (1, 0, 0, \epsilon) - (1, \epsilon, 0, 0) \times 1 - q2 \times 0 = (0, -\epsilon, 0, \epsilon)$$

Its length is again $\|w_3\| = \sqrt{2}\epsilon$ so we get the unit vector

$$\boxed{q_3 = \frac{w_3}{\|w_3\|} = (0, -1, 0, 1)/\sqrt{2}}$$

That is

```
In [3]: Q̃ = [ 1     0      0
              ϵ   -1/√2 -1/√2
              0    1/√2   0
              0    0     1/√2 ]

Out[3]: 4×3 Array{Float64,2}:
         1.0        0.0         0.0
         1.0e-10  -0.707107   -0.707107
         0.0        0.707107    0.0
         0.0        0.0         0.707107
```

and

```
In [4]: Q̃'*Q̃

Out[4]: 3×3 Array{Float64,2}:
          1.0           -7.07107e-11  -7.07107e-11
         -7.07107e-11    1.0           0.5
         -7.07107e-11    0.5           1.0
```

This is **very different** from the expected I matrix. The problem is that $q_2^T q_3 = 1/2$, which is **very far from orthogonal**.

This is the basic problem with Gram–Schmidt, especially the "classical" Gram–Schmidt process as presented in class and in the textbook: if the starting vectors are close to being dependent (e.g. nearly parallel), then in the presence of roundoff errors it can produce very non-orthogonal results. There is a "modified" Gram–Schmidt process presented in the textbook that is much better, but in practice computers nowadays use an algorithm that is better still, called "Householder QR" — this is why the Julia `qr(A)` function did such a good job, above.

Gram–Schmidt is still quite a useful way to *think* about orthogonalization and QR factorization. But a lot of care is required in translating *conceptual* algorithms into *practical* algorithms. (Practical algorithms are not the focus of 18.06, but are covered in 18.335 and many textbooks, such as *Numerical Linear Algebra* by Trefethen and Bau. But is important to be at least vaguely aware of the difference between our conceptual approach and a serious numerical implementation.)

1.6 Problem 6

In class, we showed that applying Gram–Schmidt to the columns of A, **from left to right**, corresponds to a factorization $A = QR$ where R is upper-triangular and the columns of Q are our orthonormal basis ($Q^TQ = I$).

If you do Gram–Schmidt from **right to left** on the columns of A, what would the corresponding factorization of the matrix A look like?

1.6.1 Solution

Suppose we do Gram–Schmidt from right-to left on the columns of $A =
$\begin{pmatrix} a_1 & a_3 & \cdots & a_n \end{pmatrix}$
$. That is:

$$\tilde{q}_n = \frac{a_n}{\|a_n\|}\tilde{q}_{n-1} = \frac{a_{n-1} - \tilde{q}_n\tilde{q}_n^T a_{n-1}}{\|\cdots\|}\vdots$$

We clearly still get a matrix $ =
$\begin{pmatrix} \tilde{q}_1 & \tilde{q}_2 & \cdots & \tilde{q}_n \end{pmatrix}$
$ with orthonormal columns made from the columns of A. Furthermore since these are **column** operations, they corresponding to multiplying A by another matrix **on the right** to get $\tilde{Q}$ or vice-versa. So, we still must have a factorization of the form:

$$A = \tilde{Q}\tilde{R}$$

What does the matrix $\tilde{R}$ look like? The **k-th column** of $\tilde{R}$ indicates what **linear combination** of the columns of $\tilde{Q}$ gives a_k (the k-th column of A). Since we did Gram–Schmidt from right to left, by construction we know that a_k is in the span of $\{\tilde{q}_k, \tilde{q}_{k+1}, \ldots, \tilde{q}_n\}$, i.e. a_k is made from **columns k to n of Q˜**. That means that the k-th column of $\tilde{R}$ only has **nonzero entries in rows ≥ k**. But this is precisely a **lower-triangular matrix**.

So, Gram–Schmidt from right to left on an $m \times n$ matrix A (assumed full column rank, as in class) produces the factorization $A = \tilde{Q}\tilde{R}$ where the $m \times n$ matrix $\tilde{Q}$ has orthonormal columns ($\tilde{Q}^T\tilde{Q} = I$) and the $n \times n$ matrix $\tilde{R}$ is an invertible lower-triangular matrix.

Alternate solution Let

$$P = \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ & \vdots & & & \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

be the permutation matrix corresponding to the permutation that reverses the order of the components. Then applying Gram-Schmidt from right to left is the same thing as applying Gram-Schmidt to AP (A with columns in reverse order). That is let

$$AP = Q'R'$$

be the classical QR factorization of AP. Then the matrix Q that we obtain by doing Gram-Schmidt right to left is $Q'P$ (Q' with columns reversed to match A) and so the corresponding factorization of A is

$$A = (AP)P = Q'R'P = (Q'P)(PR'P)$$

Comparing to above, this is $A = \tilde{Q}\tilde{R}$, where $\tilde{Q} = Q'P$ and $\tilde{R} = PR'P$. The $\tilde{Q}$ matrix still has orthonormal columns, of course (explicitly: $\tilde{Q}^T\tilde{Q} = P^TQ'^TQ'P = P^TP = I$). The matrix $\tilde{R} = PR'P$ is lower triangular, since R' is upper triangular and $PR'P$ flips both its rows and columns as in problem 4a of pset 2.