Math 307: Problems for section 3.1–3.2

October 25, 2009

1. **Use the Cauchy–Schwarz inequality for real vectors to show**

$$\|\mathbf{x}+\mathbf{y}\|^2 \le (\|\mathbf{x}\|+\|\mathbf{y}\|)^2$$

Under what circumstances is the inequality an equality?

$$\begin{aligned}
\langle \mathbf{x}+\mathbf{y}, \mathbf{x}+\mathbf{y}\rangle &= \langle \mathbf{x},\mathbf{x}\rangle + \langle \mathbf{x},\mathbf{y}\rangle + \langle \mathbf{y},\mathbf{x}\rangle + \langle \mathbf{y},\mathbf{y}\rangle \text{ (linearity of the inner product)}\\
&= \langle \mathbf{x},\mathbf{x}\rangle + 2\langle \mathbf{x},\mathbf{y}\rangle + \langle \mathbf{y},\mathbf{y}\rangle \text{ (symmetry of inner products for real vectors)}\\
&= \|\mathbf{x}\|^2 + 2\langle \mathbf{x},\mathbf{y}\rangle + \|\mathbf{y}\|^2\\
&\le \|\mathbf{x}\|^2 + 2\,|\langle \mathbf{x},\mathbf{y}\rangle| + \|\mathbf{y}\|^2\\
&\le \|\mathbf{x}\|^2 + 2\|\mathbf{x}\|\|\mathbf{y}\| + \|\mathbf{y}\|^2 \text{ (Cauchy–Schwarz)}\\
&= (\|\mathbf{x}\| + \|\mathbf{y}\|)^2.
\end{aligned}$$

The inequality is an equality if both $\langle \mathbf{x},\mathbf{y}\rangle$ is positive and $|\langle \mathbf{x},\mathbf{y}\rangle| = \|\mathbf{x}\|\|\mathbf{y}\|$. Combining these and using the relation for real vectors $\langle \mathbf{x},\mathbf{y}\rangle = \|\mathbf{x}\|\|\mathbf{y}\|\cos\theta$ where θ is the angle between the vectors we have equality if $\cos\theta = 1$. In other words if the two vectors have the same direction.

2. **(a) Show that if P is an orthogonal projection, then so is $Q = I - P$.**
   (b) Show that $N(P) = R(Q)$.

   (a) $Q^2 = (I-P)(I-P) = I - P - P + P^2 = I - 2P + P = I - P = Q$

   (b) Suppose $\mathbf{x} \in N(P)$, i.e., $P\mathbf{x} = \mathbf{0}$ Then $\mathbf{x} = I\mathbf{x} = (P+Q)\mathbf{x} = Q\mathbf{x}$ so $\mathbf{x} \in R(Q)$. Conversely suppose that $\mathbf{x} \in R(Q)$, i.e., $\mathbf{x} = Q\mathbf{y}$ for some $\mathbf{y}$. Then $P\mathbf{x} = PQ\mathbf{y} = \mathbf{0}$ since $PQ = 0$. Thus $\mathbf{x} \in N(P)$.

3. **Using MATLAB/Octave or otherwise, compute the matrix P for the projection onto the line spanned by $\mathbf{a} = \begin{bmatrix}1\\2\\0\\1\end{bmatrix}$ in $\mathbb{R}^4$. Compute the matrix Q for the projection onto the (hyper-)plane orthogonal to a. (Provide the commands used.)**

```
a=[1;2;0;1];
P = (a'*a)^(-1)*a*a'

P =

   0.16667   0.33333   0.00000   0.16667
   0.33333   0.66667   0.00000   0.33333
   0.00000   0.00000   0.00000   0.00000
   0.16667   0.33333   0.00000   0.16667
```

```
Q=eye(4)-P

Q =

   0.83333  -0.33333   0.00000  -0.16667
  -0.33333   0.33333   0.00000  -0.33333
   0.00000   0.00000   1.00000   0.00000
  -0.16667  -0.33333   0.00000   0.83333
```

4. **Using MATLAB/Octave or otherwise, compute the matrix P for the projection onto the plane spanned by $\begin{bmatrix}1\\1\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\2\\3\end{bmatrix}$ and $\begin{bmatrix}1\\0\\-1\\-3\end{bmatrix}$. (Careful: these vectors are not linearly independent so A^TA is not invertible. This means you can't use the formula $P = A(A^TA)^{-1}A^T$ directly for the matrix A containing all the vectors above as columns.) (Provide the commands used.)**

   We define $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$ to be these three vectors and check for dependence using `rref`

```
a1=[1;1;1;0]; a2=[0;1;2;3]; a3=[1;0;-1;-3];
rref([a1 a2 a3])

ans =

   1   0   1
   0   1  -1
   0   0   0
   0   0   0
```

   This shows that these three vectors are linearly dependent, and we can throw out $\mathbf{a}_3$ without affecting the span. The desired projection matrix is the projection onto the range of `B = [a1 a2]`. The matrix B does have linearly independent columns so we may use our formula for the projection.

```
B=[a1 a2];
P=B*(B'*B)^(-1)*B'

P =

   0.42424   0.33333   0.24242  -0.27273
   0.33333   0.33333   0.33333   0.00000
   0.24242   0.33333   0.42424   0.27273
  -0.27273   0.00000   0.27273   0.81818
```

5. **Using least squares, find the best quadratic fit for the points $(1,5)$, $(2,3)$, $(3,3)$, $(4,4)$, $(5,5)$. To do this, write down the system of linear equations for a, b and c that expresses the condition that $p(x) = ax^2 + bx + c$ passes through these points. These equations have no solution, Use MATLAB/Octave to find the least squares solution. Provide the commands you used and a plot of the result together with the points.**

The equations we wish to solve are

$$ax_1^2 + bx_1 + c = y_1 \tag{1}$$
$$ax_2^2 + bx_2 + c = y_2 \tag{2}$$
$$\vdots \tag{3}$$
$$ax_n^2 + bx_n + c = y_n \tag{4}$$
$$\tag{5}$$

or

$$\begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n^2 & x_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

The least squares equations, obtained by multiplying both sides by the adjoint of the matrix on the left are

$$\begin{bmatrix} \sum_{i=1}^n x_i^4 & \sum_{i=1}^n x_i^3 & \sum_{i=1}^n x_i^2 \\ \sum_{i=1}^n x_i^3 & \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i & n \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n x_i^2 y_i \\ \sum_{i=1}^n x_i y_i \\ \sum_{i=1}^n y_i \end{bmatrix}$$

In the example of this question, we put all the x values in one vector, and the y values in other and compute the solution $C = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ like this. We will use `sum(X)` to add up all the components of a vector. Recall that `X.^4` takes the fourth power of each component of X and `X.*Y` multiplies the components of vectors `X` and `Y`.

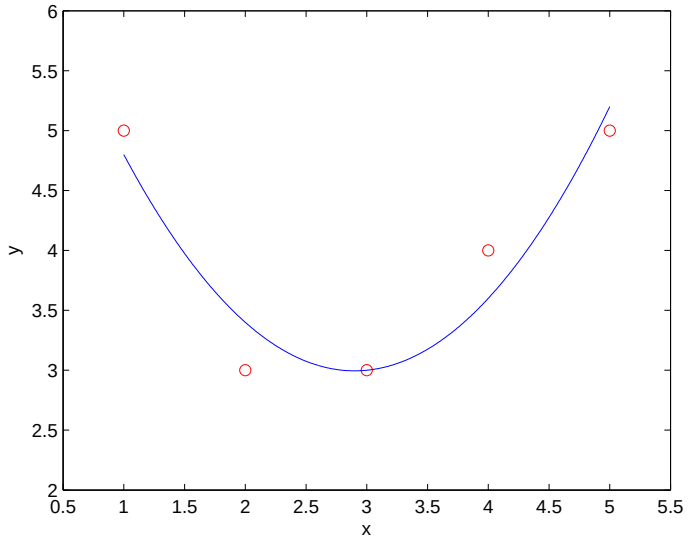
```
X=[1 2 3 4 5];
Y=[5 3 3 4 5];
L=[sum(X.^4) sum(X.^3) sum(X.^2); sum(X.^3) sum(X.^2) sum(X); sum(X.^2) sum(X) length(X)];
B=[sum(X.^2.*Y); sum(X.*Y); sum(Y)];
C=L\B
```

```
C =

   0.50000
  -2.90000
   7.20000
```

Now we perform the plot. We first plot the points and then the quadratic curve.

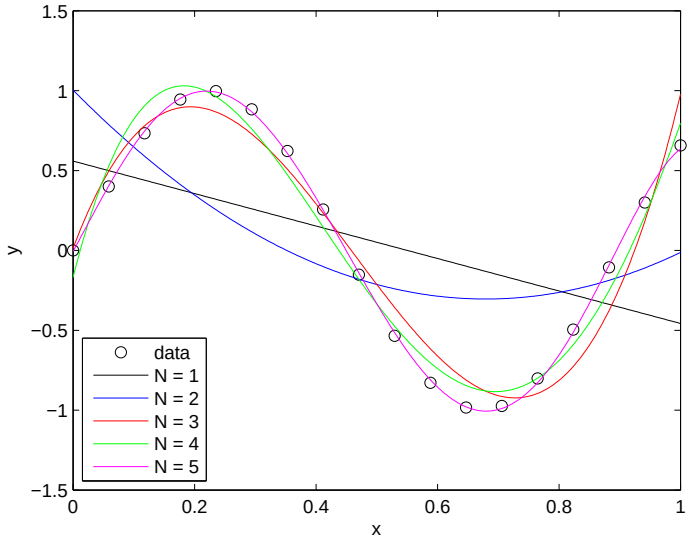
```
plot(X,Y,'or')
hold on
XX=linspace(1,5,500);
plot(XX,C(1)*XX.^2 + C(2)*XX + C(3)*ones(1,length(XX)))
axis([.5,5.5,2,6])
```



6. **Write a MATLAB/Octave m file that plots (on the same plot) the five least squares polynomial fits for polynomials of order 1 to 5 through the points (x_i, y_i) given by `X=linspace(0,1,18)` and `Y=sin(7*X)`.**

   One possible way to do this is

```
X = linspace(0,1,18);
Y = sin(7*X);
XL=linspace(0,1,500);
for N = 1:5
    V = vander(X);
    A = V(:,18-N:18);
    a = A'*A\A'*Y';
    YL(N,:) = polyval(a,XL);
end
plot(X, Y, 'ko', XL, YL(1,:), 'k', XL, YL(2,:), 'b', XL, YL(3,:), ...
     'r', XL, YL(4,:), 'g', XL, YL(5,:), 'm');
legend('data', 'N = 1', 'N = 2', 'N = 3', 'N = 4', 'N = 5', 'location', 'SW');
```



7. **Consider the points $(1,0)$, $(2,0)$, $(3,0)$ and $(4,10)$ and consider doing a polynomial fit with polynomials of order zero, that is, constant functions $p(x) = a_1$. In the least squares fit problem we are finding the value of a_1 that minimizes the usual Euclidean norm $\|[0-a_1, 0-a_1, 0-a_1, 10-a_1\|$. Solve this. What happens when we replace the Euclidean norm by the 1-norm in this problem? Find the value of a_1 that minimizes the 1-norm $\|[0-a_1, 0-a_1, 0-a_1, 10-a_1\|_1$. This is the so-called least sums problem. Is it more or less sensitive to outliers in the data?**

   To minimize the usual Euclidean norm, we can minimize the square of the norm instead. To do this we must find the value of a_1 that makes $\|[0-a_1, 0-a_1, 0-a_1, 10-a_1\|^2 = 3a_1^2 + (10-a_1)^2$ the smallest. This is a first year calculus problem that has solution $a_1 = 10/4$. So the outlier point is pulling the solution up from $a_1 = 0$ (which is clearly what the solution would be without the outlier point) to $a_1 = 10/4 = 2.5$

   To minimize the 1-norm we must find the value of a_1 that makes

   $$\|[0-a_1, 0-a_1, 0-a_1, 10-a_1\|_1 = 3|a_1| + |10-a_1|$$

   the smallest. This can't be done by setting the derivative to zero, because the function is not differentiable. However, this function is piecewise linear so we can figure out where it is smallest.

   When $a_1 \leq 0$ then $|a_1| = -a_1$ and $|10-a_1| = 10-a_1$. So, in this interval, $3|a_1| + |10-a_1| = -4a_1 + 10$ which is a decreasing straight line. This is smallest at the right endpoint $a_1 = 0$.

   In the interval $0 \leq a_1 \leq 10$ we have $|a_1| = a_1$ and $|10-a_1| = 10-a_1$. So in this interval $3|a_1| + |10-a_1| = 2a_1 + 10$. This is an increasing straight line which is smallest at the left endpoint $a_1 = 0$.

   Similarly, when $a_1 \geq 10$ we find $3|a_1| + |10-a_1| = 4a_1 - 10$. This is an increasing straight line.

   Thus, the function to be minimized is decreasing to the left of $a_1 = 0$ and increasing to the right of $a_1 = 0$. It's minimum must occur at $a_1 = 0$

   So we see that the least sums problem completely ignores the outlier point in this example. You can't get less sensitive than that!

8. **Show in some random examples that if MATLAB/Octave is asked to solve an overdetermined system (that is, more equations than unknowns) using `A\b` then the least squares solution is returned.**

Let's define A and b so that there are 10 equations and 5 unknowns. The corresponding system of equations will almost never have a solution.

```
A = rand(10,5);
b = rand(10,1);
```

Nevertheless MATLAB/Octave gives an answer when we ask it to solve the corresponding system

```
A\b
ans =

   0.44395
  -0.73732
   0.79427
   0.70093
   0.12219
```

Let's compare to the solution of the least squares equation $A^TA\mathbf{x} = A^T\mathbf{b}$.

```
A'*A\A'*b
ans =

   0.44395
  -0.73732
   0.79427
   0.70093
   0.12219
```

The two "solutions" are the same.

9. **Verify that the matrix $P = A(A^TA)^{-1}A^T$ that projects onto the range of A when A^TA is invertible satisfies $P^2 = P$ and $P^T = P$.**

   $P^2 = A(A^TA)^{-1}A^TA(A^TA)^{-1}A^T = A(A^TA)^{-1}A^T = P$ because one factor of $(A^TA)^{-1}$ cancels with the factor of A^TA in the middle.

   To show that $P^T = P$ we can use that $(ABC)^T = C^TB^TA^T$. Thus $P^T = (A(A^TA)^{-1}A^T)^T = (A^T)^T((A^TA)^{-1})^TA^T = A((A^TA)^{-1})^TA^T$. so we are done if we can show that $((A^TA)^{-1})^T = (A^TA)^{-1}$. This follows from the fact that the inverse of the transpose is the transpose of the inverse. To see that, note that $(A^{-1})^TA^T = (AA^{-1})^T = I^T = I$

10. **Using MATLAB/Octave, find an orthonormal set of vectors $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$ with the same span as $\begin{bmatrix}1\\1\\2\\0\\0\\0\end{bmatrix}\begin{bmatrix}1\\0\\1\\0\\1\\0\end{bmatrix}$ and $\begin{bmatrix}0\\0\\1\\1\\1\\0\end{bmatrix}$. Provide the commands that you used.**

```
a1=[1 1 2 0 0 0]';
a2=[1 0 1 0 1 0]';
a3=[0 0 1 1 1 0]';
A=[a1 a2 a3];
[Q R] = qr(A,0)
```

```
Q =

  -0.40825   0.40825   0.51640
  -0.40825  -0.40825  -0.00000
  -0.81650   0.00000  -0.25820
  -0.00000   0.00000  -0.77460
  -0.00000   0.81650  -0.25820
  -0.00000   0.00000   0.00000

R =

  -2.44949  -1.22474  -0.81650
   0.00000   1.22474   0.81650
   0.00000   0.00000  -1.29099
```

The three orthonormal vectors are the columns of Q.

11. **Do the following computational experiment. First start with a random symmetric 10×10 matrix A (for example `B=rand(10,10); A=B'*B;` will produce such a matrix) and compute its QR factorization. Call the factors Q_1 and R_1. Now multiply Q_1 and R_1 in the "wrong" order to obtain $A_2 = R_1 Q_1$ and compute the QR factorization of the resulting matrix A_2. Repeat this step to obtain a sequence of matrices Q_k, R_k and A_k. Do these sequences converge? If so can you identify the limit? (Hint: `eig(C)` computes the eigenvalues of `C`).**

We can automate the iterative procedure. Here I'll do it ten times, check the value of R and compare to the eigenvalues of A

```
A=rand(10,10);
A=A'*A;
[Q R] = qr(A);
for k=[1:10]
  [Q R] = qr(R*Q,0);
end
R

R =

 Columns 1 through 7:

  -28.26041   -0.00000   -0.00000    0.00000   -0.00000   -0.00000   -0.00000
    0.00000   -2.20459   -0.48470   -0.00121    0.00004   -0.00000    0.00000
    0.00000    0.00000   -2.23187    0.00364   -0.00022    0.00000   -0.00000
    0.00000    0.00000    0.00000   -1.15933    0.05953   -0.00037    0.00003
    0.00000    0.00000    0.00000    0.00000   -0.76320   -0.00532    0.00035
    0.00000    0.00000    0.00000    0.00000    0.00000   -0.45709    0.01440
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000   -0.29274
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000

 Columns 8 through 10:
```

```
   -0.00000   -0.00000   -0.00000
   -0.00000    0.00000   -0.00000
    0.00000   -0.00000    0.00000
   -0.00000   -0.00000    0.00000
   -0.00000    0.00000   -0.00000
    0.00000   -0.00000    0.00000
    0.00001   -0.00000    0.00000
   -0.11023    0.00000   -0.00000
    0.00000   -0.04297    0.00005
    0.00000    0.00000    0.01267

eig(A)

ans =

    0.012671
    0.042971
    0.110228
    0.292491
    0.457455
    0.761467
    1.162016
    1.988698
    2.474165
   28.260409
```

Okay, lets do it 100 more times.

```
for k=[1:100]
  [Q R] = qr(R*Q);
end
R

R =

 Columns 1 through 7:

  -28.26041   -0.00000   -0.00000   -0.00000    0.00000   -0.00000   -0.00000
    0.00000   -2.47417   -0.00000    0.00000   -0.00000    0.00000   -0.00000
    0.00000    0.00000   -1.98870   -0.00000    0.00000   -0.00000   -0.00000
    0.00000    0.00000    0.00000   -1.16202    0.00000   -0.00000   -0.00000
    0.00000    0.00000    0.00000    0.00000   -0.76147    0.00000   -0.00000
    0.00000    0.00000    0.00000    0.00000    0.00000   -0.45745    0.00000
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000   -0.29249
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000

 Columns 8 through 10:

   -0.00000   -0.00000   -0.00000
    0.00000    0.00000   -0.00000
```

```
  -0.00000   -0.00000    0.00000
   0.00000   -0.00000    0.00000
  -0.00000   -0.00000   -0.00000
   0.00000    0.00000    0.00000
   0.00000   -0.00000    0.00000
  -0.11023   -0.00000    0.00000
   0.00000   -0.04297    0.00000
   0.00000    0.00000    0.01267
```

We see that the matrix R is a diagonal matrix with ± 1 times the eigenvalues on the diagonal. If the MATLAB/Octave `qr` function returned the factorization we carried out in class (where the diagonal entries of R are always positive), this calculation would return the eigenvalues on the diagonal. In this situation, Q has ± 1 on the diagonal (as you can check), and the sign will indicate whether we need to flip the entry in R to get the eigenvalue. A more coherent way of saying this is that QR will be diagonal with the eigenvalues on the diagonal. Let's check

```
Q*R

ans =

 Columns 1 through 7:

   28.26041    0.00000    0.00000    0.00000   -0.00000    0.00000    0.00000
    0.00000    2.47417    0.00000   -0.00000    0.00000   -0.00000    0.00000
    0.00000    0.00000    1.98870    0.00000   -0.00000    0.00000    0.00000
   -0.00000    0.00000   -0.00000    1.16202   -0.00000    0.00000    0.00000
    0.00000   -0.00000    0.00000   -0.00000    0.76147   -0.00000    0.00000
   -0.00000    0.00000   -0.00000    0.00000    0.00000    0.45745   -0.00000
    0.00000   -0.00000    0.00000   -0.00000   -0.00000   -0.00000    0.29249
    0.00000    0.00000   -0.00000    0.00000    0.00000   -0.00000   -0.00000
   -0.00000   -0.00000    0.00000    0.00000   -0.00000    0.00000    0.00000
    0.00000    0.00000   -0.00000   -0.00000    0.00000   -0.00000   -0.00000

 Columns 8 through 10:

    0.00000    0.00000    0.00000
   -0.00000   -0.00000    0.00000
    0.00000    0.00000   -0.00000
   -0.00000    0.00000   -0.00000
    0.00000    0.00000    0.00000
   -0.00000   -0.00000   -0.00000
   -0.00000    0.00000   -0.00000
    0.11023    0.00000   -0.00000
   -0.00000    0.04297   -0.00000
    0.00000   -0.00000    0.01267
```

12. **Verify that the set of orthogonal matrices form a group (as defined in the notes).**

    We need to verify (1) that I is an orthogonal matrix, (2) that if U is orthogonal then so is U^{-1} and (3) that if U_1 and U_2 are orthogonal, then so is $U_1 U_2$.

    (1) $I^T I = I^2 = I$ so I is orthogonal. (It also clearly preserves the length of vectors.)

(2) $(U^{-1})^T U^{-1} = (U^T)^{-1} U^{-1} = (UU^T)^{-1} = I^{-1} = I$

(3) $(U_1 U_2)^T U_1 U_2 = U_2^T U_1^T U_1 U_2 = U_2^T U_2 = I$.

13. **Suppose that A is a 3×3 matrix whose first two columns are linearly independent, but where the third column is a linear combination of the first two. Show that $A = QR$ where Q is a 3×2 matrix with orthonormal columns and R is a 2×3 matrix with $R_{2,1} = 0$.**

Let $A = [\mathbf{a}_1|\mathbf{a}_2|\mathbf{a}_3]$. The usual QR decomposition for $[\mathbf{a}_1|\mathbf{a}_2]$ gives

$$[\mathbf{a}_1|\mathbf{a}_2] = [\mathbf{q}_1|\mathbf{q}_2] \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \end{bmatrix}$$

On the other hand $\mathbf{a}_3$ is a linear combination of $\mathbf{q}_1$ and $\mathbf{q}_2$ because these orthonormal vectors are obtained by performing the Gram-Schmidt procedure to $\mathbf{a}_1$ and $\mathbf{a}_2$ and must therefore have the same span. Thus

$$\mathbf{a}_3 = [\mathbf{q}_1|\mathbf{q}_2] \begin{bmatrix} R_{1,3} \\ R_{2,3} \end{bmatrix}$$

for some coefficients $R_{1,3}$ and $R_{2,3}$. Combining these equations gives

$$[\mathbf{a}_1|\mathbf{a}_2|\mathbf{a}_3] = [\mathbf{q}_1|\mathbf{q}_2] \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & R_{2,3} \end{bmatrix}$$