# MA342 - Numerical Analysis
# Homework # 5 (Least Squares, Eigenvalues)

Dylan Allen Student

March 30, 2017

---

1. Find Chapter 7 exercise #17 on page 179 of the textbook. Follow all of the instructions for part (a), but in part (b) we are going to do something a bit different. There are 10,000 test numbers that are not in the training set and I want to try two different ways of determining which digit is in the test image. The textbook authors give the method of finding the minimum norm distance from the training images and the test image. They are suggesting that the mean training image that is closest in $\mathbb{R}^{784}$ is likely the correct numeral. I suggest one other test: project the test image vector $d$ onto the mean training images T(i,:). That is, the projection of $d$ onto T(i,:) is: $\frac{d \cdot T(i,:)}{d \cdot d}$, and determine the largest projection. I'm suggesting that the mean training image that gives the largest projection will likely be the correct numeral.

   Write MATLAB code that tests all 10,000 test images and gathers statistics on how often each method correctly identifies the test image. Report your answers by stating the proportion of correct identifications for each of the 10 numerals.

Soln: a Produce an image of the digits

b The textbook author suggests to find the correct digit in the training set we should compute the $norm(T(I,:) - d$. The smallest value should be the correct digit. When we set d to be the first

row for digits 0 through 9 and compute $norm(T(I,:) - d$. To do this we loop through every test digit and took the norm of the difference of T and d. We then calculate how many times the min value is in the correct row as the digit and take the proportion, the results can be seen in Table 1.
When using the second approach, $\frac{d \cdot T(i,:)}{d \cdot d}$, we get similiar proportions as we did for the first method. This summary can be seen in Table 1 The digit that was the most wrong was digit 5 with .6446 correctly identified. When comparing the two methods it seems that they are fairly close for identifying the test image; however, method two may be a bite better.

| Digits | Proportion correct Method 1 | Proportion correct Method 2 |
|---|---|---|
| 0 | 0.8959 | 0.9224 |
| 1 | 0.9621 | 0.9392 |
| 2 | 0.7568 | 0.7655 |
| 3 | 0.8059 | 0.8436 |
| 4 | 0.8238 | 0.8075 |
| 5 | 0.6861 | 0.6446 |
| 6 | 0.8633 | 0.8831 |
| 7 | 0.8327 | 0.8268 |
| 8 | 0.7372 | 0.7628 |
| 9 | 0.7166 | 0.7998 |

Table 1: Proportion that method one and two correctly identified the test image

2. We are going to find the polynomial of degree 4 that best fits the function $y = \cos(4t) + 0.1\epsilon(t)$ at 50 equally spaced points $t$ between 0 and 1. Here we are using $\epsilon(t)$ as a function that outputs normally distrubuted random white noise.

Soln: To begin a counter was initialized for both the LU and QR difference. A for loop was then started which looped through 10000 iterations. The t vector, y vector and A matrix were all generated to begin solving for the coefficients. Both methods, LU and QR, were used to determine the coefficients. Once the 5 coefficients for each method were determined the error term was computed. We then checked to see which method ended up having a smaller error. After running through the loop we see that LU outperformed QR 4990 times while QR outperformed LU 5010 times. This suggests that the two methods are nearly identical when determining the coefficients for a least squares regression.

3. On pages 312 and 313 of the textbook the author describes the deflation technique for finding the eigenvector associated with the second largest eigenvalue. Write a MATLAB function MyPower2 to find the second largest eigenvalue and eigenvector pair by putting the author's exposition into practice. Test your code on a symmetric matrix A and compare against MATLAB's eig command.

Soln: To begin, a random n by n symmetric matrix was generated. An x vector and a v vector were initialized as required for the power method. The power method is then ran and the greatest eigenvalue and its corresponding eigenvector are computed. To find the second eigenvalue eigenvector pair we use the deflation technique as described in the book. This method was then compared to the matlab "eig" command. We recorded one output from the matlab command: using the eig command we see that, $ans = [0.0341, 0.1565, 0.4383, 4.3619]$, where each value represents an eigenvalue. Using MyPower2 we see that $lambda2 = .4383$, where $lambda2$ is the second largest eigenvalue.

# A Code for 5.4.7

```
n = 4;
A = rand(n,n);
A = A'*A;
x = rand(n,1);
v = x/norm(x)
for k = 2:100
    vtemp = A*v;
    lambda1 = dot(vtemp,v);
    v = vtemp/norm(vtemp);
end

newx = x - dot(x,v)*v;
v2 = newx/norm(newx)
for k = 2:1000
    v2temp = A*v2;
        if mod(k,4) == 0
             v2temp = v2temp - dot(v2temp,v)*v;
        end

    lambda2 = dot(v2temp,v2);
    v2 = v2temp/norm(v2temp);

end

[vec, val] = eig(A);
DominantEV = max(max(val))
thev = [v, vec(:,1)]
thev2 = [v2,vec(:,2)]
```

# B Code for 5.3.7

```
counterLU = 0;
counterQR = 0;
for n = 1:10000
    numpts = 50;
    t = linspace(0,1,numpts);
    ys = cos(4*t) + 0.1*randn(size(t));
    order = 4;
    A = ones(length(t),order + 1);
    for n = 1:order+1
        A(:,n) = t.^(n-1);
    end
    ATA = A' * A;
    b = A' * ys';
    [L,U] = MyLU(ATA);
    [y] = Lsolve(L,b);
    [x1] = Usolve(U,y) ;
    [Q,R] = myQR(A);
```

```
    %R*x = Q'b
    newb = Q'*ys';
    [x2] = Usolve(R,newb) ;
    approxLU = x1(1) + x1(2).*t + x1(3)*t.^2 +x1(4)*t.^3 + x1(5)*t.^4;
    approxQR = x2(1) + x2(2).*t + x2(3)*t.^2 +x2(4)*t.^3 + x2(5)*t.^4;
    f = cos(4*t);
    differenceLU = sum((approxLU - f).^2);
    differenceQR = sum((approxQR - f).^2);

    if differenceLU < differenceQR
        counterLU = counterLU + 1;
    else
       counterQR = counterQR + 1;
    end


end
 counterLU
 counterQR
```