Homework 4
Numerical Matrix Analysis
Math 543
Stephen Giang

Problem 1:

$$\text{Let } A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow Q = \begin{bmatrix} 0.1231 & 0.9045 & -0.1111 \\ 0.4924 & 0.3015 & -0.4444 \\ 0.8616 & -0.3015 & -0.8889 \end{bmatrix}, R = \begin{bmatrix} 8.1240 & 9.6011 & 11.0782 \\ 0 & 0.9045 & 1.8091 \\ 0 & 0 & 0.0000 \end{bmatrix}$$

Problem 9.1 (a): Run the six-line MATLAB program of Experiment 1 to produce a plot of approximate Legendre polynomials.

Problem 9.1 (b): For k = 0,1,2,3, plot the difference on the 257-point grid between these approximations and the exact polynomials (7.11). How big are the errors, and how are they distributed?

Solution 9.1 (b): The errors when k = 0, and k = 1, are 0. The errors for the other k values are in between ± 0.015. The errors get larger as the degree of each polynomial gets bigger, or for greater k values.

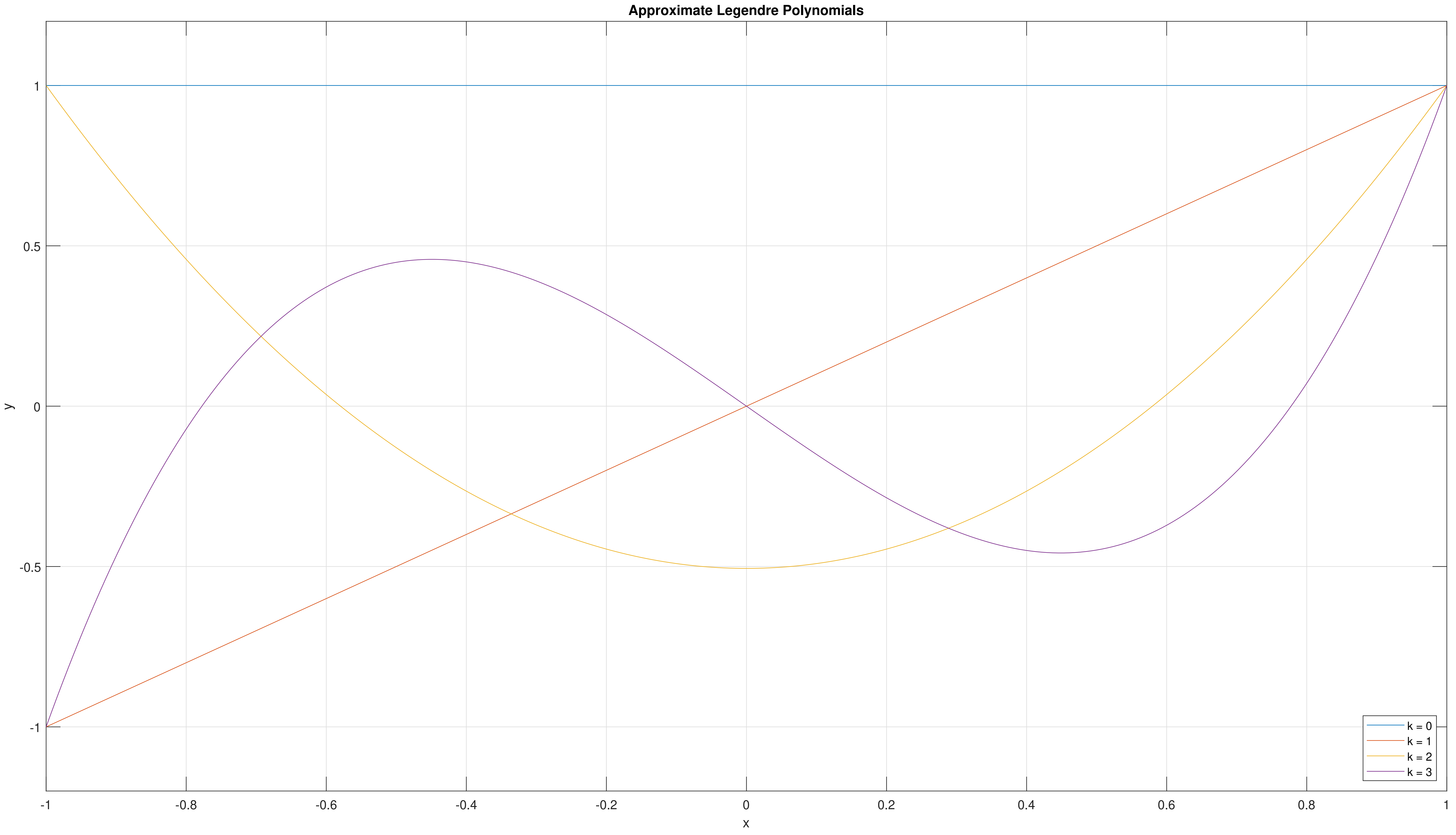
Problem 9.2: In Experiment 2, the singular values of A match the diagonal elements of a QR factor R approximately. Consider now a very different example. Suppose Q = I and A = R, the m x m matrix (a Toeplitz matrix) with 1 on the main diagonal, 2 on the first superdiagonal, and 0 everywhere else

Solution 9.2 (a): What are the eigenvalues, determinant, and rank of A?

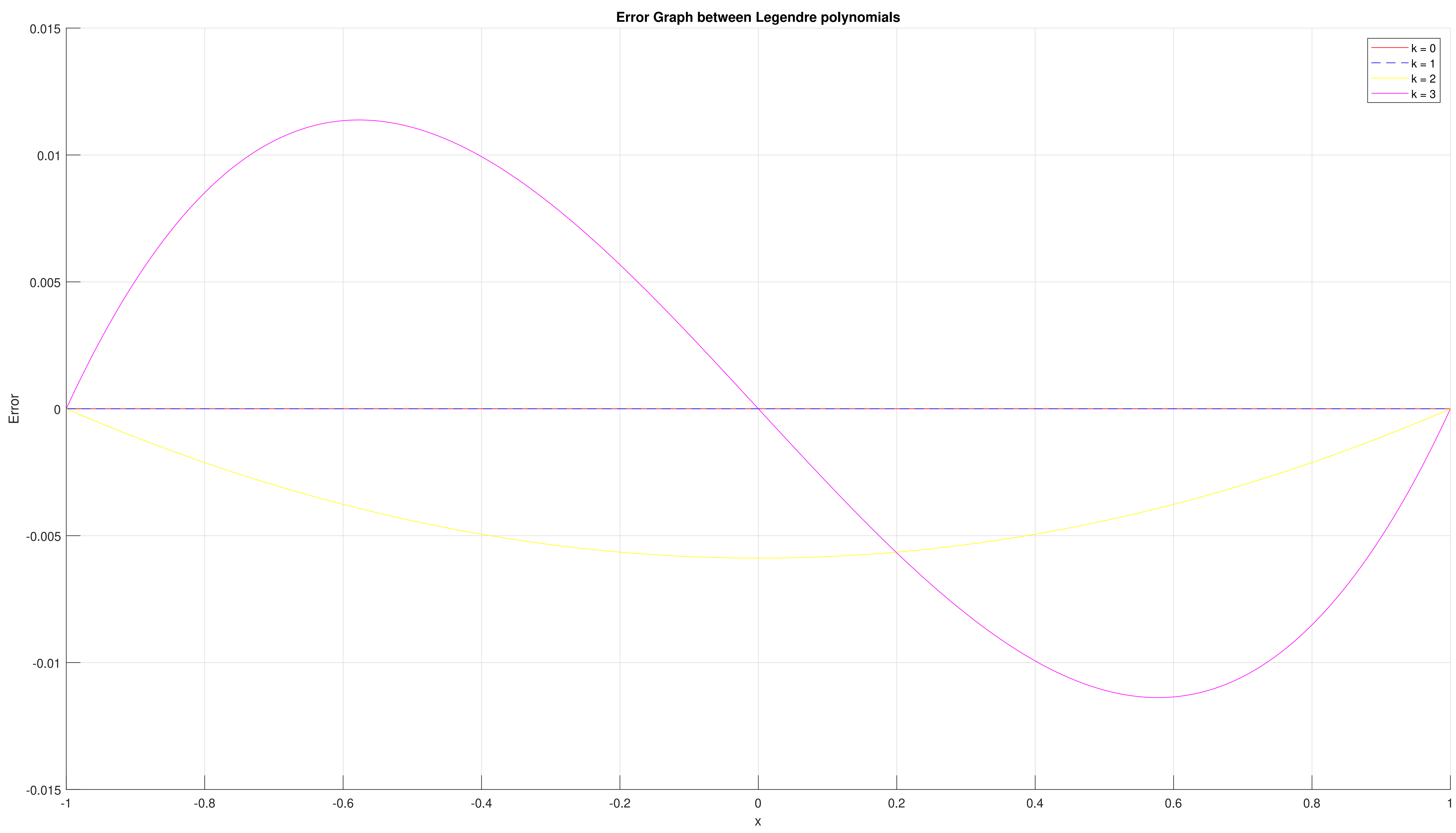
$$\begin{aligned} &\text{All } eig(A) = 1 \\ &det(A) = 1 \\ &rank(A) = m \end{aligned}$$

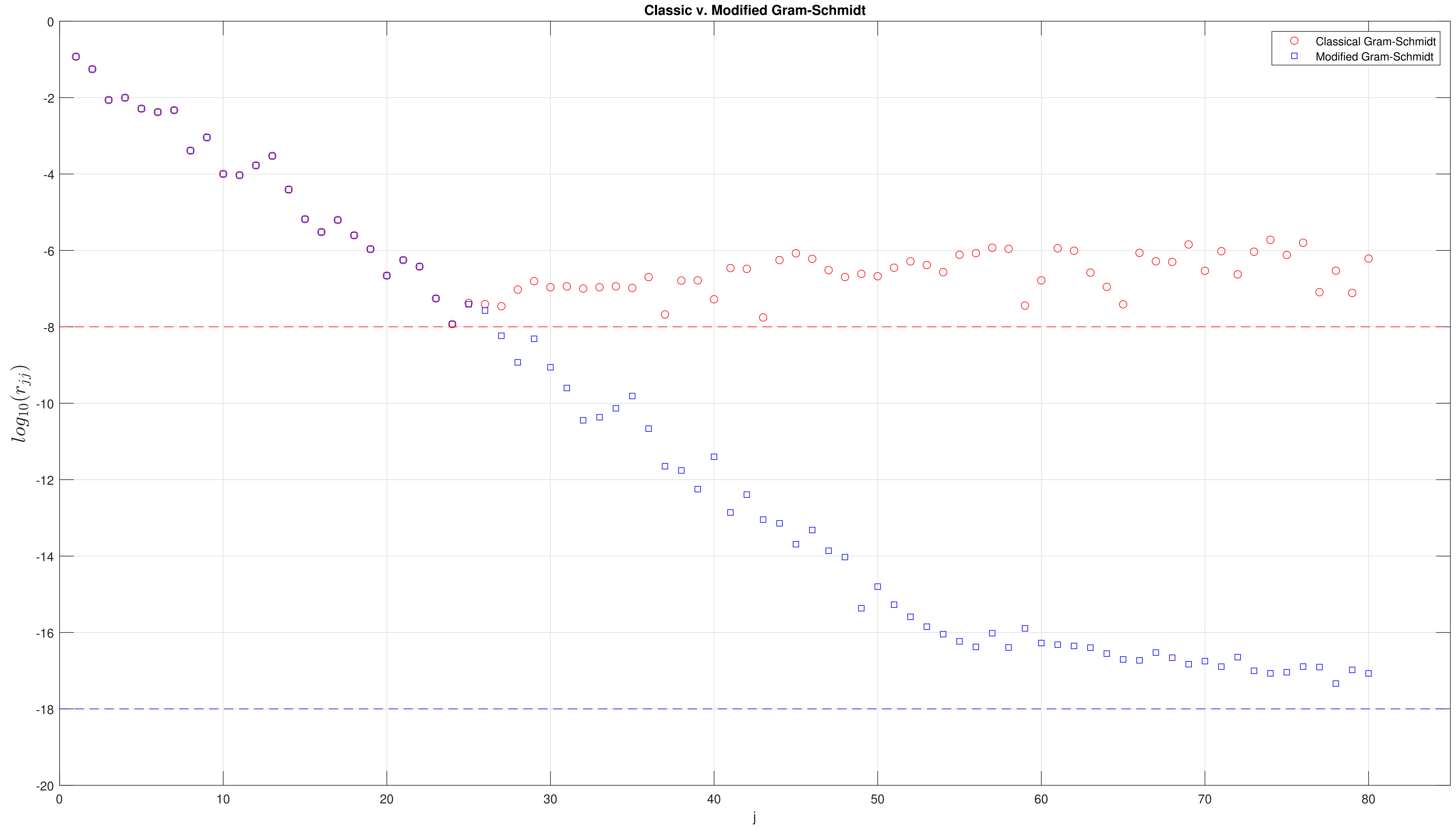
Solution 9.2 (b): What is A^{-1}?

$A^{-1} = m \times m$ matrix with diagonal entries being 1, and its superdiagonal entries being -2



Approximate Legendre Polynomials
1
0.5
0
-0.5
-1
y
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
x
k = 0
k = 1
k = 2
k = 3


Error Graph between Legendre polynomials
k = 0
k = 1
k = 2
k = 3
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
Error
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
x


Classic v. Modified Gram-Schmidt
Classical Gram-Schmidt
Modified Gram-Schmidt
$log_{10}(r_{jj})$
j
0
-2
-4
-6
-8
-10
-12
-14
-16
-18
-20
0
10
20
30
40
50
60
70
80

```
function [Q, R] = qr_mgs(A)

[m, n] = size(A);
Q = zeros([m n]);
R = zeros([n n]);
v = zeros([m n]);

for i = 1 : n
    v(:,i) = A(:, i);
end

for i = 1 : n
    R(i,i) = norm(v(:,i));
    Q(:,i) = v(:,i) / R(i,i);

    for j = (i + 1) : n
        R(i,j) = Q(:,i)' * v(:,j);
        v(:,j) = v(:,j) - R(i,j) * Q(:,i);
    end
end
```