EE551 Assignment

1. Extended Kalman Filter

MATLAB code:

```
clear
N = 100;
x = [1;1];
xi = x;
P = 1000*eye(2);
q1 = 0.15*randn(N,1);
q2 = 0.15*randn(N,1);
r = 0.2*randn(N,1);
Q = [var(q1) 0; 0 var(q2)];
R = var(r);

%Actual states calculation
for i=1:N
    xa = [x(1)/(1+x(2)^2)+q1(i); x(1)*x(2)/(1+x(2)^2)+q2(i)];
    y(i) = xa(1)+r(i);
    xa1(i,:) = xa(1);
    xa2(i,:) = xa(2);
    x = xa;
end

%Estimated states calculation
for i=1:N
    A = [1/(1+xi(2)^2) -2*xi(1)*xi(2)/(1+xi(2)^2)^2;
         xi(2)/(1+xi(2)^2) xi(1)*(1-xi(2)^2)/(1+xi(2)^2)^2];
    x = [xi(1)/(1+xi(2)^2); xi(1)*xi(2)/(1+xi(2)^2)];
    P = A*P*A'+Q;
    C = [1 0];
    K = P*C'*pinv(C*P*C'+R);
    x = x+K*(y(i)-x(1));
    P = (eye(2)-K*C)*P;
    xk1(i,:) = x(1);
    xk2(i,:) = x(2);
    yk(i) = C*x;
    if sum((xa1(i)-xk1(i))^2+(xa2(i)-xk2(i))^2)/sum(xa1(i)^2+xa2(i)^2)<0.01
        i1 = i;
        break;
    end
    xi = x;
end

t1 = 1:i1;
subplot(2,1,1)
plot(t1,xa1(1:i1),'m',t1,xk1,'b');
xlabel('k');
ylabel('x1');
legend({'Actual State','Estimated State'})
subplot(2,1,2)
plot(t1,xa2(1:i1),'m',t1,xk2,'b');
xlabel('k');
ylabel('x2');
legend({'Actual State','Estimated State'})
```

Outputs:

Different initial conditions:

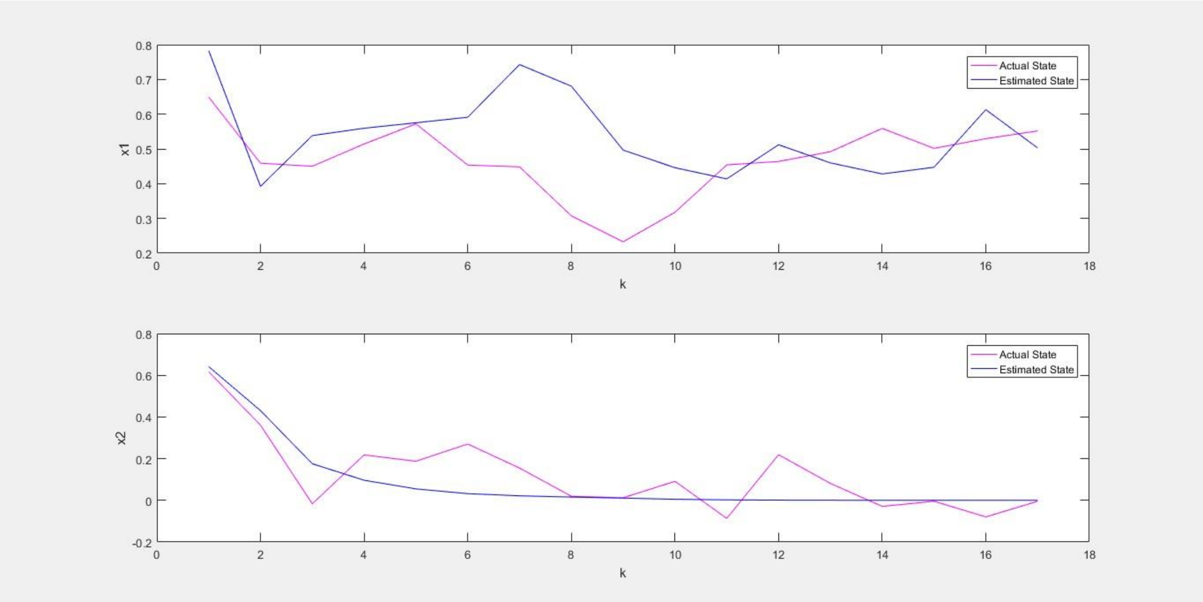

Figure 1: Initial state = [1 1]

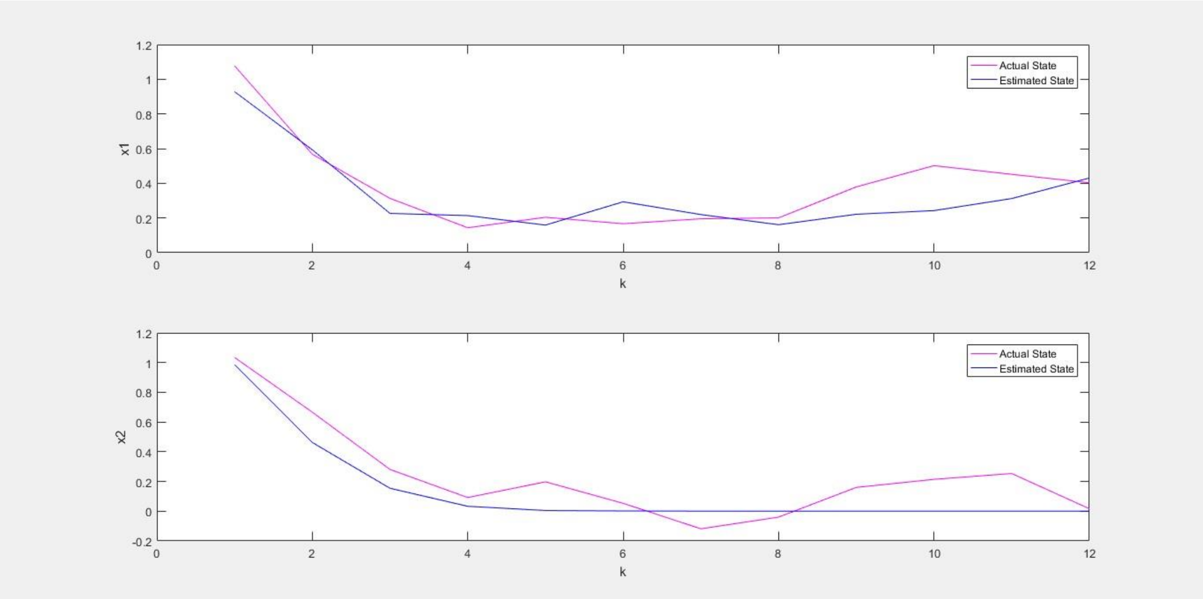

Figure 2: Initial state = [2 1]

Different values of Q and R:

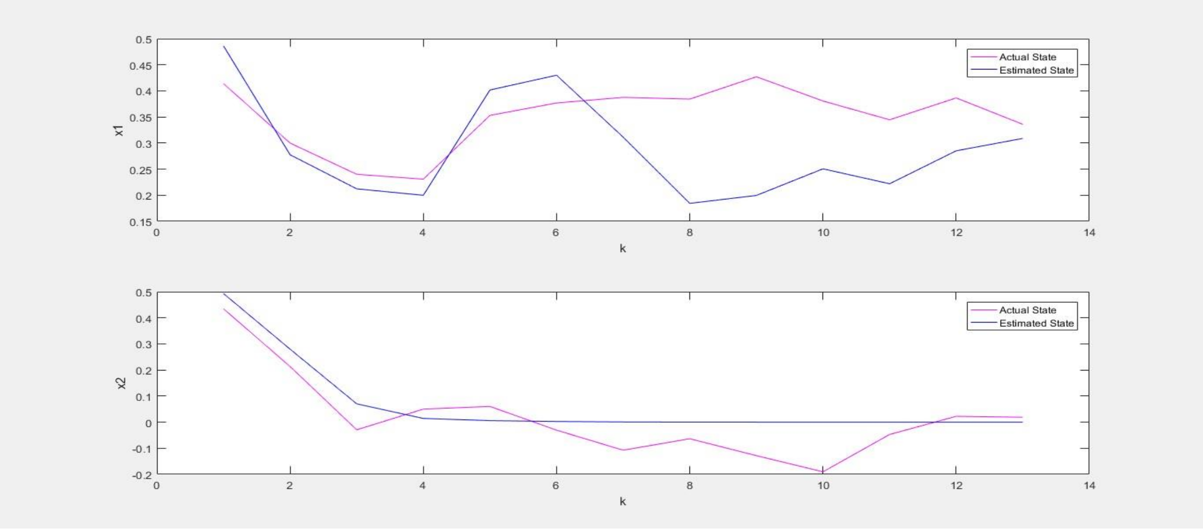

Figure 3: Q = 0.1, R = 0.2

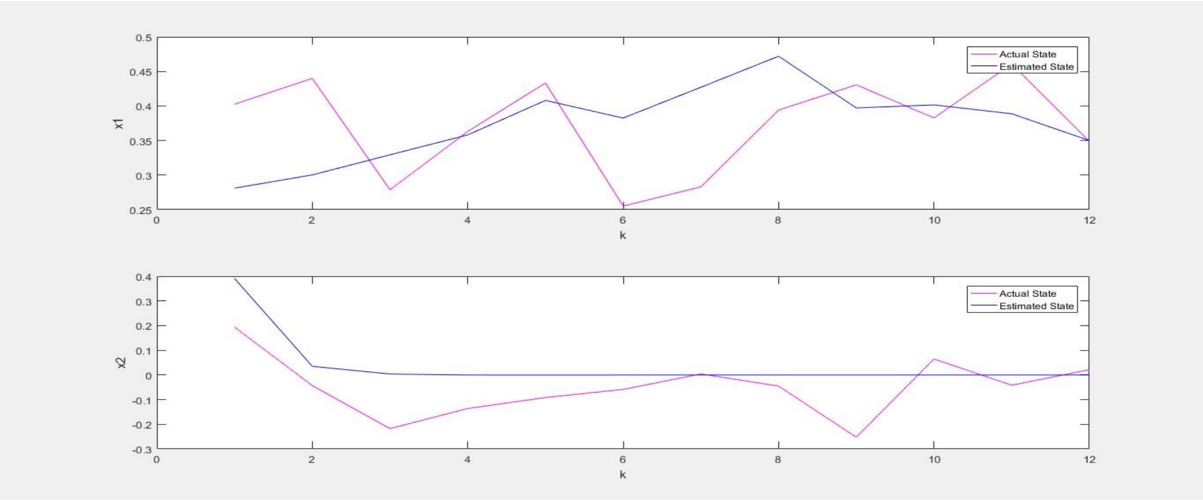

Figure 4: Q = 0.15, R = 0.2

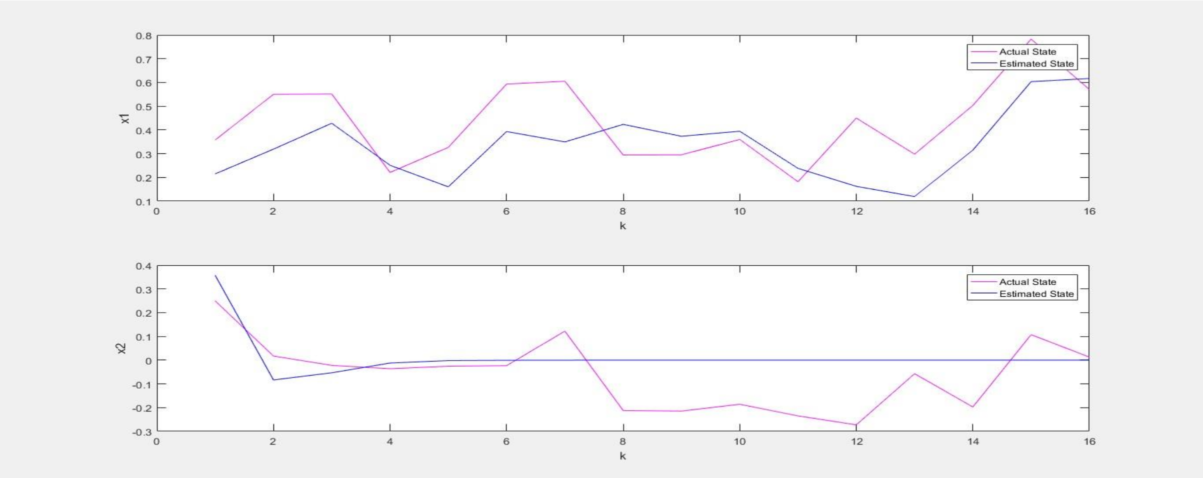

Figure 5: Q = 0.2, R = 0.2

2. NARX model using FROLS algorithm

MATLAB code:

```
clear all
N = 200;
u = -1+2*rand(N,1);
y = zeros(N,1);
e = 0.1*randn(N,1);
%ny = 2;
%nu = 2;
%ne = 0;
l = 3;
del = 0.05;
y(1:2) = e(1:2);

for k = 3:N
    y(k) = -0.605*y(k-1)-0.163*y(k-2)^2+0.588*u(k-1)-0.24*u(k-2)+e(k);
end

z = 1;
for f = 0:l
    for h = 0:l
        for i = 0:l
            for j = 0:l
                if f+h+i+j>l
                    continue;
                end
                for k = 3:N
                    t(k-2) = (y(k-1)^j)*(y(k-2)^i)*(u(k-1)^h)*(u(k-2)^f);
                end
                %fprintf('z=%d y1=%d y2=%d u1=%d u2=%d \n',z,j,i,h,f);
                Z(z,:) = [j i h f];
                D(:,z) = t; %Dictionary of candidate model terms
                z = z+1;
            end
        end
    end
end

M = z-1;
y1 = y(3:N);
sig = y1'*y1;

%Step-1: s=1
for j = 1:M
    q = D(:,j);
    g1(j) = (y1'*q)/(q'*q);
    ERR(j) = g1(j)^2*(q'*q)/sig;
end
[c, b(1)] = max(ERR);
Al(:,1) = D(:,b(1)); %Matrix consisting of alpha
Q(:,1) = D(:,b(1)); %Matrix consisting of q
g(1) = g1(b(1)); %Vector consisting of model parameters, g
A(1,1) = 1;
err(1) = ERR(b(1)); %Vector consisting of ERR values
ers = err(1);
```

```
%Step-2: s>=2
for i = 2:M
    k = 1;
    x = zeros(M-i+1,1);
    g1 = zeros(1,M-i+1);
    D0 = zeros(N-2,M-i+1);
    Q1 = zeros(N-2,M-i+1);
    ERR = zeros(M-i+1,1);

    for j = 1:M
        if find(b==j)
            continue;
        else
            x(k) = j;
            k = k+1;
        end
    end

    k = 1;
    for j = 1:size(x)
        D0(:,k) = D(:,x(j));
        s = zeros(N-2,1);
        p = D(:,x(j));
        for r = 1:i-1
            q1 = Q(:,r);
            s = s+((p'*q1)/(q1'*q1)*q1);
        end
        q = p-s;
        g1(j) = (y1'*q)/(q'*q);
        ERR(j) = g1(j)^2*(q'*q)/sig;
        Q1(:,j) = q;
        k = k+1;
    end
    [c, d] = max(ERR);
    b(i) = x(d);
    Al(:,i) = D0(:,d); %Matrix consisting of alpha
    Q(:,i) = Q1(:,d); %Matrix consisting of q
    g(i) = g1(d); %Vector consisting of model parameters, g
    for r = 1:i-1
        A(r,i) = (Q(:,r)'*Al(:,i))/(Q(:,r)'*Q(:,r));
    end
    A(i,i) = 1;
    err(i) = ERR(d); %Vector consisting of ERR values
    ers = ers+err(i); %Sum of ERR values
    if (1-ers)<=del
        break;
    end
end

fprintf('Model terms:\n');
for k = 1:size(g,2)
    fprintf('y(k-1)^%d*y(k-2)^%d*u(k-1)^%d*u(k-2)^%d\n',Z(b(k),:));
end
fprintf('\n\nModel parameters:\n');
for k = 1:size(g,2)
    fprintf('%.4f\n',g(k));
end
```

Outputs:

Index	Model terms	Model Parameters
1	y(k-1)	-0.6966
2	u(k-1)	0.5941
3	u(k-2)	-0.2499
4	y^2(k-2)	-0.1433

Table 1: SD of noise = 0.1

Index	Model terms	Model Parameters
1	y(k-1)	-0.6838
2	u(k-1)	0.6543
3	u(k-2)	-0.2545
4	y^2(k-2)	-0.1927
5	u(k-1)*u(k-2)	-0.1216
6	y^2(k-2)*u(k-2)	-0.0980
7	u^2(k-1)	0.0908
8	u^2(k-1)*u(k-2)	-0.1362
9	y^3(k-1)	0.0581
10	y(k-1)*u(k-2)	-0.1219
11	y^2(k-1)	0.0636
12	u(k-1)*u^2(k-2)	0.1963
13	u^2(k-2)	0.0983
14	y(k-1)*y(k-2)	-0.1143
15	y^3(k-2)	-0.1565
16	y(k-2)	0.1085
17	y(k-1)*y(k-2)*u(k-2)	-0.1379
18	y(k-2)*u^2(k-1)	-0.1667
19	y(k-1)*u^2(k-1)	0.0742
20	y(k-1)*u^2(k-2)	0.0924
21	y^2(k-1)*u(k-2)	-0.0598
22	y(k-1)*y^2(k-2)	0.0634
23	y(k-2)*u(k-2)	-0.0793
24	y(k-2)*u(k-1)	0.1196
25	y(k-1)*u(k-1)	-0.0766
26	y^2(k-2)*u(k-1)	0.1747
27	y^2(k-1)*y(k-2)	0.1571
28	u^3(k-2)	0.0800
29	y(k-1)*y(k-2)*u(k-1)	-0.2668
30	y(k-1)* u(k-1)*u(k-2)	0.0161
31	y(k-2)*u^2(k-2)	-0.0389
32	u^3(k-1)	-0.1725
33	1	-1.1359
34	y(k-2)*u(k-1)*u(k-2)	0.0150
35	y^2(k-1)*u(k-1)	-0.0012

Table 2: SD of noise = 0.2