```
%%%
%%% Kalman Filter design example for FDI
%%%

clear all, close all, clc

A = [1.1269   -0.4940    0.1129
     1.0000         0         0
          0    1.0000         0];

B = [-0.3832       % The first input is the control signal
      0.5919
      0.5191];

C = [1 0 0];

Tc = 1;

D = 0; % The model has 1 input and 1 output!!!

Plant = ss(A,[B B],C,[D 0],Tc,'inputname',{'u' 'w'},'outputname','y');

Q = 1; R = 1; %%% Assuming that Q = R = 1;

rank(ctrb(A,B))
rank(obsv(A,C))

[kalmf,L,P] = kalman(Plant,Q,R); % Design first procedure

kalmf = kalmf(1,:);
```

```
Akf = kalmf.a
Bkf = kalmf.b
Ckf = kalmf.c
Dkf = kalmf.d


%%% Design second procedure (simpler, when the filter is implemented in Simulink!)
%%%
%%% Check the controllability of (A,B) and the observability of (A,C), which must be equal ↙
to the dimension of A
%%% rank(ctrb(A,B))
%%% rank(obsv(A,C))
%%% [Pkf,Ekf,Kkft] = dare(A',C',B*Q*B',R); % dual CARE for discrete time systems!
%%% Kkf = Kkft';
%%% Kalman filter matrices: apart from the Kalman gain, it is an output observer!
%%% Akf = A - Kkf * C
%%% Bkf = [B Kkf]
%%% Ckf = C
%%% Dkf = zeros(1,2) % 1 output (row of Dkf) and 2 inputs (columns of Dkf)
%%%


%%%
%%% Comparison with the output dynamic observer
%%%

Ko = place(A',C',[0.2 0.21 0.22])';
Ao = A - Ko*C;
Bo = [B Ko];
Co = C;
Do = zeros(1,2);
```

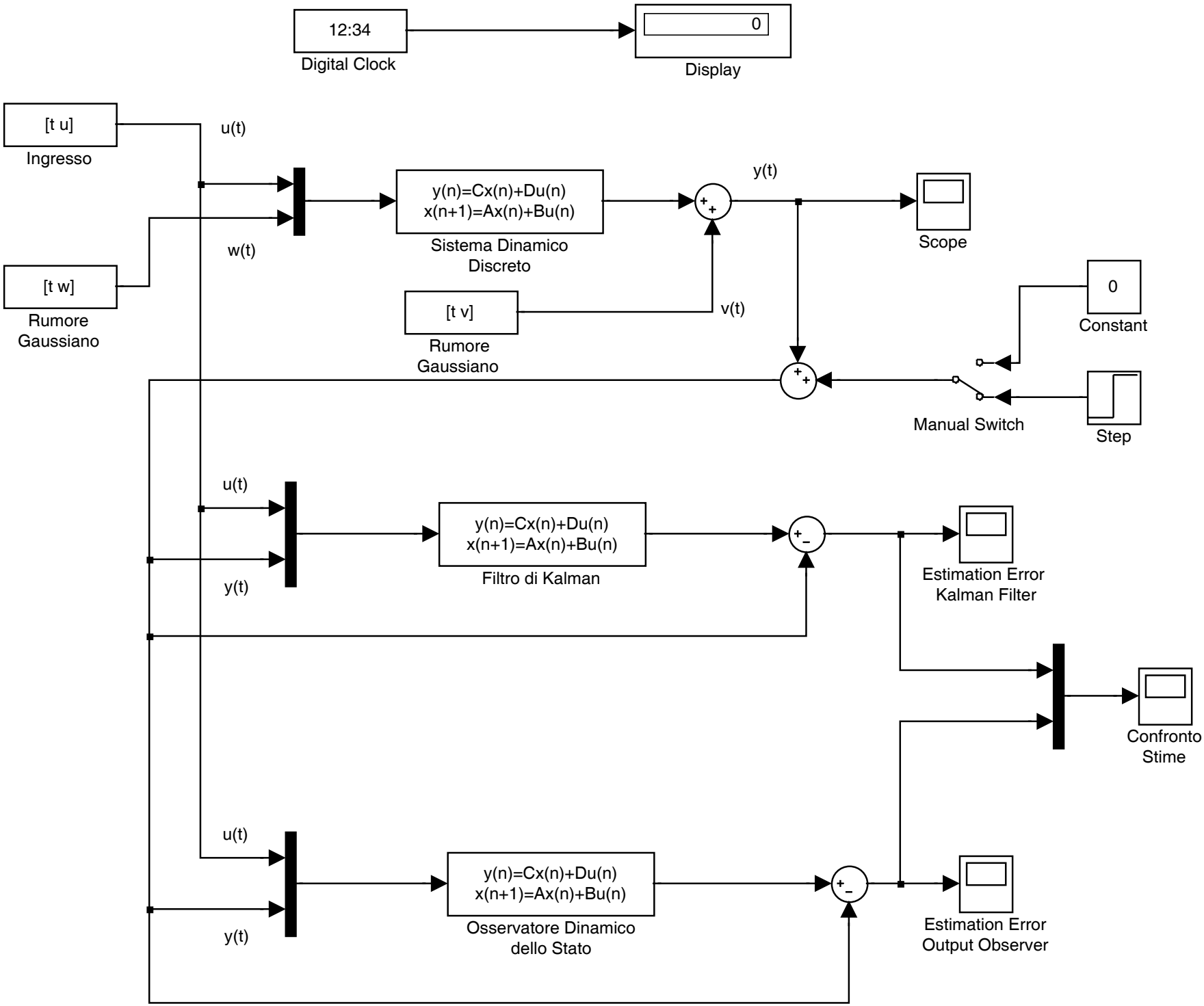
```
%%% Kalman filter design for Matlab simulations only!

a     = A;
b     = [B B 0*B];
c     = [C;C];
d     = [0 0 0;0 0 1];
Plant = ss(a,b,c,d,-1,'inputname',{'u' 'w' 'v'},'outputname',{'y' 'yv'});

sysp = parallel(Plant,kalmf,1,1,[],[]);

SimModel = feedback(sysp,1,4,2,1);

SimModel = SimModel([1 3],[1 2 3]);

t = [0:5000]';
u = sin(t/5);
n = length(t);

randn('seed',0);
w = randn(n,1);
w = sqrt(Q)*(w-mean(w))/std(w);

randn('seed',1);
v = randn(n,1);
v = sqrt(R)*(v-mean(v))/std(v);

[out,x] = lsim(SimModel,[w,v,u]);

y  = out(:,1);   % true response
ye = out(:,2);   % filtered response
```

```
yv = y + v;      % measured response

resh = yv - ye;

N = length(resh);

fault_size = 0.2;

fy = zeros(N,1); fy(round(N/2)+1:N,1) = fault_size*std(yv)*ones(size(fy(round(N/2)+1:N)));

resf = yv + fy - ye; % Faulty residual for FDI

figure, plot([0:N-1]',resh,'-g',[0:N-1]',resf,'--r')
        xlabel('Samples.'), ylabel('r(t)'), title('(-) fault-free and (--) faulty residuals↙
')

moving_window = 0; %1 or 0, boolean option


if(moving_window),

window_length = 500;

zetah = zeros(N-window_length+1,1);
zetaf = zeros(N-window_length+1,1);
meanh = zeros(N-window_length+1,1);
meanf = zeros(N-window_length+1,1);
stdh = zeros(N-window_length+1,1);
stdf = zeros(N-window_length+1,1);

for indx = 1:N-window_length+1,
```

```
    zetah(indx) = whiterestest(resh(indx:window_length+indx-1));
    zetaf(indx) = whiterestest(resf(indx:window_length+indx-1));

    meanh(indx) = mean(resh(indx:window_length+indx-1));
    meanf(indx) = mean(resf(indx:window_length+indx-1));

    stdh(indx) = std(resh(indx:window_length+indx-1));
    stdf(indx) = std(resf(indx:window_length+indx-1));

end

else

window_length = 500;

zetah = zeros(N-window_length+1,1);
zetaf = zeros(N-window_length+1,1);
meanh = zeros(N-window_length+1,1);
meanf = zeros(N-window_length+1,1);
stdh = zeros(N-window_length+1,1);
stdf = zeros(N-window_length+1,1);

for indx = 1:N-window_length+1,

    zetah(indx) = whiterestest(resh(1:window_length+indx-1));
    zetaf(indx) = whiterestest(resf(1:window_length+indx-1));

    meanh(indx) = mean(resh(1:window_length+indx-1));
    meanf(indx) = mean(resf(1:window_length+indx-1));
```

```
    stdh(indx) = std(resh(1:window_length+indx-1));
    stdf(indx) = std(resf(1:window_length+indx-1));

end % for

end %if

tzeta = [window_length:N]';

figure, plot(tzeta,meanh,'-g',tzeta,1.1*max(meanh)*ones(size(tzeta)),'-.b',tzeta,1.1*min(me ↙
anh)*ones(size(tzeta)),'-.m')
        xlabel('Samples. Fault-free case'), ylabel('r_m(t)'), title('Mean value of r(t)')

figure, plot(tzeta,meanh,'-g',tzeta,1.1*max(meanh)*ones(size(tzeta)),'-.b',tzeta,1.1*min(me ↙
anh)*ones(size(tzeta)),'-.m',...
        tzeta,meanf,'--r')
        xlabel('Samples. Faulty case'), ylabel('r_m(t)'), title('Mean value of r(t)')

figure, plot(tzeta,stdh,'-g',tzeta,1.001*max(stdh)*ones(size(tzeta)),'-.b',tzeta,0.999*min( ↙
stdh)*ones(size(tzeta)),'-.m')
        xlabel('Samples. Fault-free case'), ylabel('\sigma_r(t)'), title('Standard value of ↙
 r(t)')

figure, plot(tzeta,stdh,'-g',tzeta,1.001*max(stdh)*ones(size(tzeta)),'-.b',tzeta,0.999*min( ↙
stdh)*ones(size(tzeta)),'-.m',...
        tzeta,stdf,'--r')
        xlabel('Samples. Faulty case'), ylabel('\sigma_r(t)'), title('Standard value of r(t ↙
)')

figure, plot(tzeta,zetah,'-g',tzeta,20.1*ones(size(tzeta)),'-.b')
        xlabel('Samples. Fault-free case'), ylabel('\zeta_N(M)'), title('Residual whiteness ↙
```

```
: \chi^2_N(M) test')

figure, plot(tzeta,zetah,'-g',tzeta,20.1*ones(size(tzeta)),'-.b',tzeta,zetaf,'--r')
        xlabel('Samples. Faulty case'), ylabel('\zeta_N(M)'), title('Residual whiteness: \c ↙
hi^2_N(M) test')


return
```


12:34
Digital Clock
0
Display
[t u]
Ingresso
u(t)
w(t)
[t w]
Rumore
Gaussiano
y(n)=Cx(n)+Du(n)
x(n+1)=Ax(n)+Bu(n)
Sistema Dinamico
Discreto
y(t)
Scope
[t v]
Rumore
Gaussiano
v(t)
0
Constant
Manual Switch
Step
u(t)
y(t)
y(n)=Cx(n)+Du(n)
x(n+1)=Ax(n)+Bu(n)
Filtro di Kalman
Estimation Error
Kalman Filter
Confronto
Stime
u(t)
y(t)
y(n)=Cx(n)+Du(n)
x(n+1)=Ax(n)+Bu(n)
Osservatore Dinamico
dello Stato
Estimation Error
Output Observer

```
function zeta = whiterestest(res)

Mgdl = 8;
N = length(res);
Rr = zeros(1,Mgdl+1);

for indx=1:Mgdl+1,
    Rr(indx) = ( (res(1:N-Mgdl))' * res(indx:N+indx-1-Mgdl) )/(N-Mgdl);
end;

zeta = (N-Mgdl) *( Rr(2:Mgdl+1)*( Rr(2:Mgdl+1)' ) )/(Rr(1)*Rr(1));

return
```