H-infinity Kalman Estimation for Standard Systems with Unknown Inputs

Mr. A. Shivakumar (1), P.G student, MREC(A)
Mr. M. Praveenkumar (2), Associate Professor, MREC(A)
gr8Shivakumar@gmail.com (1)
mp.praveenkumar@gmail.com (2)

Abstract—This note considers filtering and prediction for rectangular descriptor systems with unknown inputs that affect both the system and the output. An optimal descriptor Kalman estimator (HDKE), which can simultaneously solve the filtering and prediction problems for rectangular descriptor systems with unknown inputs, is developed based on the maximum likelihood descriptor Kalman filtering method. The HDKE serves as a unified solution to solve Kalman filtering for descriptor systems and standard systems with or without unknown inputs. To reduce the computational complexity problem, some efficient alternatives to the developed HDKE are further proposed. The relationship between the HDKE and the existing literature results is also addressed. An illustrative example is compared to show the usefulness of the proposed results.

Keywords—H-infinity, Kalman estimation, vehicle on road simulation.

I. INTRODUCTION

Unknown input filtering (UIF) serves as a useful technique to solve many practical state estimation problems that often arise in systems subject to disturbances, modeling errors, system uncertainties, and reduced-order filtering therein). A general approach to solve for the state estimation of standard systems with unknown inputs that have arbitrary statistics is to apply unknown input decoupled state estimation; this yields unbiased minimum-variance filters (UMVF). Recently, the global optimality issues of the UMVF have been explored and the optimal solutions are established [2], [3]. Note that the UMVF can also be applied to solve UIF.

II. PROBLEM FORMULATION

Consider a general class of descriptor systems with unknown inputs as follows: problems for descriptor systems through a suitable system reformulation [4].

\[ E_{k+1} x_{k+1} = A_k x_k + G_k d_k + w_k \]  
\[ y_k = C_k x_k + H_k d_k + v_k \]

where \( x_k \in \mathbb{R}^n \) is the descriptor vector, \( d_k \in \mathbb{R}^r \) is the unknown input, and \( y_k \in \mathbb{R}^m \) is the measured output. Here, the row numbers of matrices \( E_{k+1} \) and \( A_k \) are equivalent (\( = \) \( m \)) and may not equal \( n \). Signals \( w_k \) and \( v_k \) are uncorrelated white noise sequences of zero-mean, with covariance matrices \( Q_k \geq 0 \) and \( R_k \geq 0 \), respectively. The initial state \( x_0 \) is with unbiased mean \( \hat{x}_0 \) and covariance matrix \( P_0 \) and is independent of \( w_k \) and \( v_k \).

The problem of interest in this note is for estimating the combination

\[ z_k = L_k x_k + D_k d_k + n_k \]

III. M-File Program

Considering example of vehicle travelling along a road:

function Hinfinity(g, duration, dt, SteadyState)
\% function Hinfinity(g, duration, dt, SteadyState)
\% % H-infinity filter simulation for a vehicle.
\% % This code also simulates the Kalman filter.
\% % INPUTS
\% % g = gamma
\% % if g is too large the program will terminate with
\% % set g = 0.01.
\% % duration = length of simulation (seconds). I used
\% % dt = step size (seconds). I used dt = 0.1.
\% % SteadyState = flag indicating use of steady state
\% % 1 = steady state, 0 = time-varying
\% if ~exist(‘g’, ’var’)
\% g = 0.01;
\% end
\% if ~exist(‘duration’, ’var’)
\% duration = 60;
\% end
\% if ~exist(‘dt’, ’var’)
\% dt = 0.1;
\% end
\% if ~exist(‘SteadyState’, ’var’)
\% SteadyState = 0;
\% end
\% measnoise = 2; % nominal velocity measurement
\% accelnoise = 0.2; % nominal acceleration noise
\% feet/sec^2

\% 2320-8007

ISSN: 2320-8007 3573
a = [1 dt; 0 1]; % transition matrix
b = [dt^2/2; dt]; % input matrix
c = [0 1]; % measurement matrix
x = [0; 0]; % initial state vector
y = c * x; % initial measurement

% Initialize Kalman filter variables
xhat = x; % initial Kalman filter state estimate
Sz = measnoise^2; % measurement error covariance
Sw = accelnoise^2 * [dt^4/4 dt^3/2; dt^3/2 dt^2]; % process noise cov
P = Sw; % initial Kalman filter estimation covariance

% Initialize H-infinity filter variables
xhatinf = x; % initial H-infinity filter state estimate
Pinf = 0.01*eye(2); % measurement error covariance
W = [0.0003 0.0050; 0.0050 0.1000]/1000;
V = 0.01;
Q = [0.01 0; 0 0.01];

% Initialize arrays for later plotting.
pos = [x(1)]; % true position array
vel = [x(2)]; % true velocity array
poshat = [xhat(1)]; % estimated position array
(Kalman filter)
velhat = [xhat(2)]; % estimated velocity array
(Kalman filter)
poshatinf = [xhatinf(1)]; % estimated position array
(H-infinity)
velhatinf = [xhatinf(2)]; % estimated velocity array
(H-infinity)
HinfGains = [0; 0]; % H-infinity filter gains
KalmanGains = [0; 0]; % Kalman filter gains
for t = 0 : dt: duration-dt
    % Use a constant commanded acceleration of 1
    % foot/sec^2.
    u = 1;
    % Figure out the H-infinity estimate.
    if (SteadyState == 1)
        % Use steady-state H-infinity gains
        K = [0.11; 0.09];
    else
        L = inv(eye(2) - g * Q * Pinf + c' * inv(V) * c * Pinf);
        K = a * Pinf * L * c' * inv(V);
        Pinf = (Pinf + Pinf') / 2;
        % Force Pinf to be symmetric.
    end
    % Update the Kalman filter state estimate.
    xhat = xhat + K * Inn;
    % Form the Innovation vector.
    Inn = y - c * xhat;
    if (SteadyState == 1)
        K = [0.1; 0.01];
    else
        % Compute the covariance of the Innovation.
        s = c * P * c' + Sz;
        % Form the Kalman Gain matrix.
        K = a * P * c' * inv(s);
        % Compute the covariance of the estimation error.
        P = a * P * a' - a * P * c' * inv(s) * c * P * a' + Sw;
        % Force P to be symmetric.
        P = (P + P') / 2;
    end
    % Plot the results
    close all; % Close all open figures
    t = 0 : dt : duration; % Create a time array
    % Plot the position estimation error
    figure;
    plot(t,pos-poshat,'r', t,pos-poshatinf,'b--'); % Plot the position error
    set(gca,'FontSize',12); set(gcf,'Color','White');
    grid;
    xlabel('Time (sec)');
    ylabel('Position Error (feet)');
    title('Position Error');
    legend('Kalman filter', 'H_{\infty} filter');
    % Plot the velocity estimation error
    figure;
    plot(t,vel-velhat,'r', t,vel-velhatinf,'b--');
    set(gca,'FontSize',12); set(gcf,'Color','White');
    grid;
    xlabel('Time (sec)');
    ylabel('Velocity Error (feet)');
    title('Velocity Estimation Error');
legend('Kalman filter', 'H_{\infty} filter');
% Plot the Kalman filter gain matrix
figure;
plot(t,KalmanGains(1,:),'r', t,KalmanGains(2,:),'b--');
set(gca,'FontSize',12); set(gcf,'Color','White');
grid;
xlabel('Time (sec)');
title('Kalman Gains');
% Plot the H-infinity filter gain matrix
figure;
plot(t,HinfGains(1,:),'r', t,HinfGains(2,:),'b--');
set(gca,'FontSize',12); set(gcf,'Color','White');
grid;
xlabel('Time (sec)');
title('H-Infinity Gains');

Fig 1, Comparision of position error versus time for Kalman and H_{\infty} filters

Fig 2, Comparision of velocity error versus time for Kalman and H_{\infty} filters

IV. CONCLUSION

The HDKE serves as a unified solution, besides the one developed using data fitting approach, to solve the addressed unknown input filtering problem for descriptor systems. Some compact forms of the HDKE, based on a novel matrix transformation and the recently proposed gain-covariance matrix concept, are developed as the extensions of the existing results in [1], [5], [2] for the general case that matrices, and exist in the system model. Finally, simulation results verify the usefulness of the proposed results.
V. REFERENCES


