Contents

problem set-up

```
% Implement a Kalman filter for a stochastic linear time invariant (SLTI)
% system in the standard form used in class (with Γ(k) =/= I).

% The problem matrices and the measurement data, z(k) for k = 1, ..., 50,
% can be loaded into your Matlab workspace by running the Matlab script
% kf_example02a.m.

% Hand in plots of the two elements of xˆ(k) vs. time and of the predicted
% standard deviations of xˆ(k) vs. time, i.e., of sqrt([P(k)_11]) and
% sqrt([P(k)_22]).

% Plot each element of xˆ(k) and its corresponding standard deviation
% together on the same graph.

% Use symbols on the plot at each of the 51 points and do not connect the
% symbols by lines (type “help plot” in order to learn how to do this).
% Also, hand in numerical values for the terminal values of xˆ(50) and P(50).

clear; clc; close all

kf_example02a;
```

KALMAN FILTER

```
Gk = Gammak;

[xhat_arr, Pxx_arr, Pzz_arr, P_cell] = kf( ...
    xhat0, P0, zhist, Fk, Gk, Qk, Hk, Rk );
```

results

```
thist0 = [ 0; thist ];

% plot
ftitle = 'States and Covariances';
figure('name', ftitle);
    subplot(2,1,1)
        plot( thist0, xhat_arr(:,1), '.' ); hold on; grid on;
        plot( thist0, xhat_arr(:,1) + sqrt( Pxx_arr ), 'r--');
        plot( thist0, xhat_arr(:,1) - sqrt( Pxx_arr ), 'r--');
        title('$\hat{x}$(1)', 'interpreter', 'latex');
        legend('$\hat{x}$', '$ \hat{x} \pm \sigma_{xx}$', 'interpreter', 'latex', 'location', 'best');
        ylabel('state units');
    subplot(2,1,2)
        plot( thist0, xhat_arr(:,2), '.' ); hold on; grid on;
        plot( thist0, xhat_arr(:,2) + sqrt( Pzz_arr ), 'r--');
```

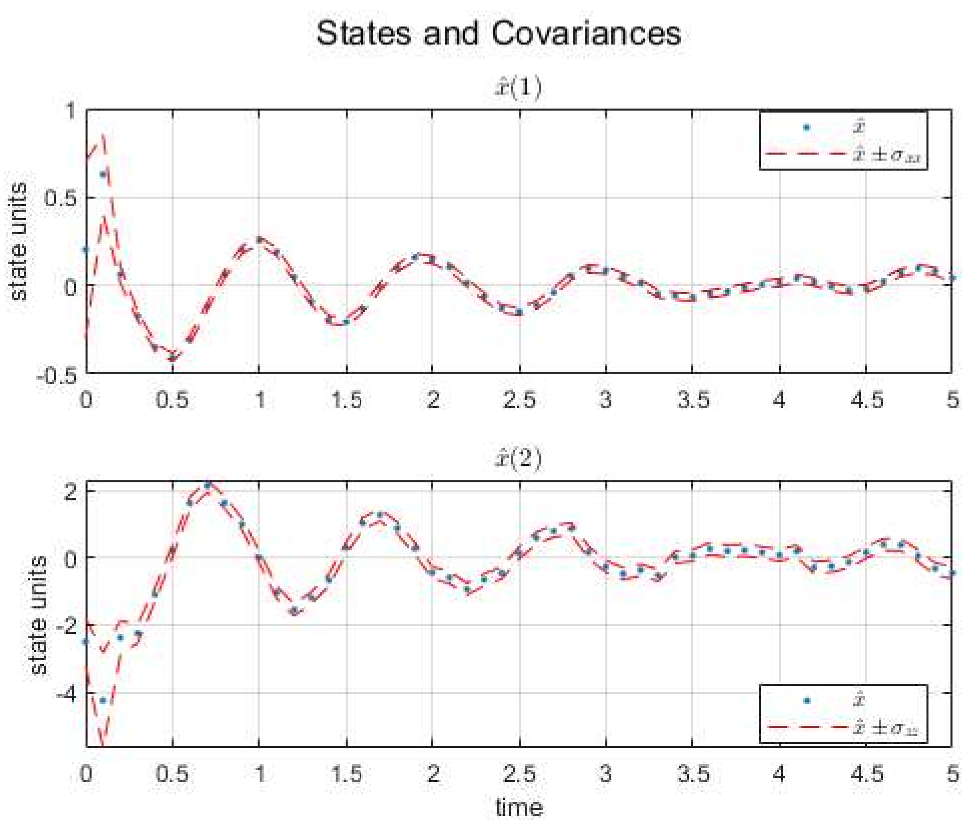
```
        plot( thist0, xhat_arr(:,2) - sqrt( Pzz_arr ), 'r--');
        title('$\hat{x}$(2)', 'interpreter', 'latex');
        legend('$\hat{x}$', '$ \hat{x} \pm \sigma_{zz}$', 'interpreter', 'latex', 'location', 'best');
        ylabel('state units');
    xlabel('time');
    sgtitle(ftitle);

% print final values
disp('xhat(50) =')
disp(xhat_arr(end,:))

disp('P(50) =')
disp(P_cell{end})
```

```
xhat(50) =
     0.040359     -0.45508

P(50) =
   0.00047006   0.00027759
   0.00027759     0.031117
```



subfunctions KALMAN FILTER

```
function [xhat_arr, Pxx_arr, Pzz_arr, P_cell] = kf( ...
    xhat0, P0, zhist, Fk, Gk, Qk, Hk, Rk )

% initialize for k = 0
xhat = xhat0;
P    = P0;
```

```
% Initialize saved output arrays
xbar_arr = [];
Pbar_arr = [];
xhat_arr = [xhat'];
P_cell   = {P};
Pxx_arr  = [P(1,1)];
Pzz_arr  = [P(2,2)];

% Propagate and filter through all measurements
for k = 0 : length(zhist)-1

    % propagate state and covar
    xbar = Fk * xhat;                         % a priori state est
    Pbar = Fk * P * Fk' + Gk * Qk * Gk';      % a posteriori covar est

    % update
    v = zhist(k+1) - Hk * xbar;               % innovation
    S = Hk * Pbar * Hk' + Rk;                 % innovation covariance
    W = Pbar * Hk' * inv(S);                  % Kalman gain
    xhat = xbar + W * v;                      % a posteriori state est
    P = Pbar - W * S * W';                    % a posteriori covar est

    % next step
    k = k + 1;

    % save states and covariances
    xbar_arr = [xbar_arr; xbar'];
    Pbar_arr = [Pbar_arr; Pbar];
    xhat_arr = [xhat_arr; xhat'];
    P_cell   = {P_cell; P};
    Pxx_arr  = [Pxx_arr; P(1,1)];
    Pzz_arr  = [Pzz_arr; P(2,2)];

end

end
```

Published with MATLAB® R2020b