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;

% Exam 2
Qk = 10;
Rk = 0.025;
```

KALMAN FILTER

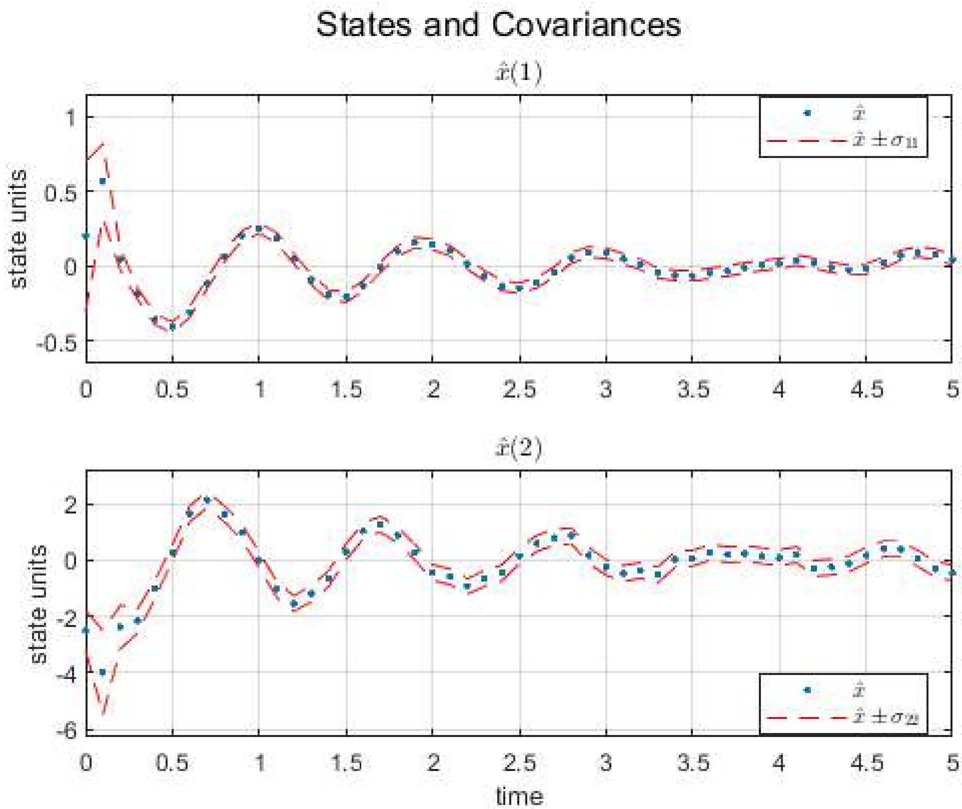
```
Gk = Gammak;

[xhat_arr, P11_arr, P22_arr, P_cell, xbar_arr, Pbar_cell, nu_arr, S_arr] = kf( ...
    xhat0, P0, zhist, Fk, Gk, Qk, Hk, Rk );

for i = 1:length(nu_arr)
    e_v(:,i) = nu_arr(i)' * S_arr(i) * nu_arr(i);
end
```

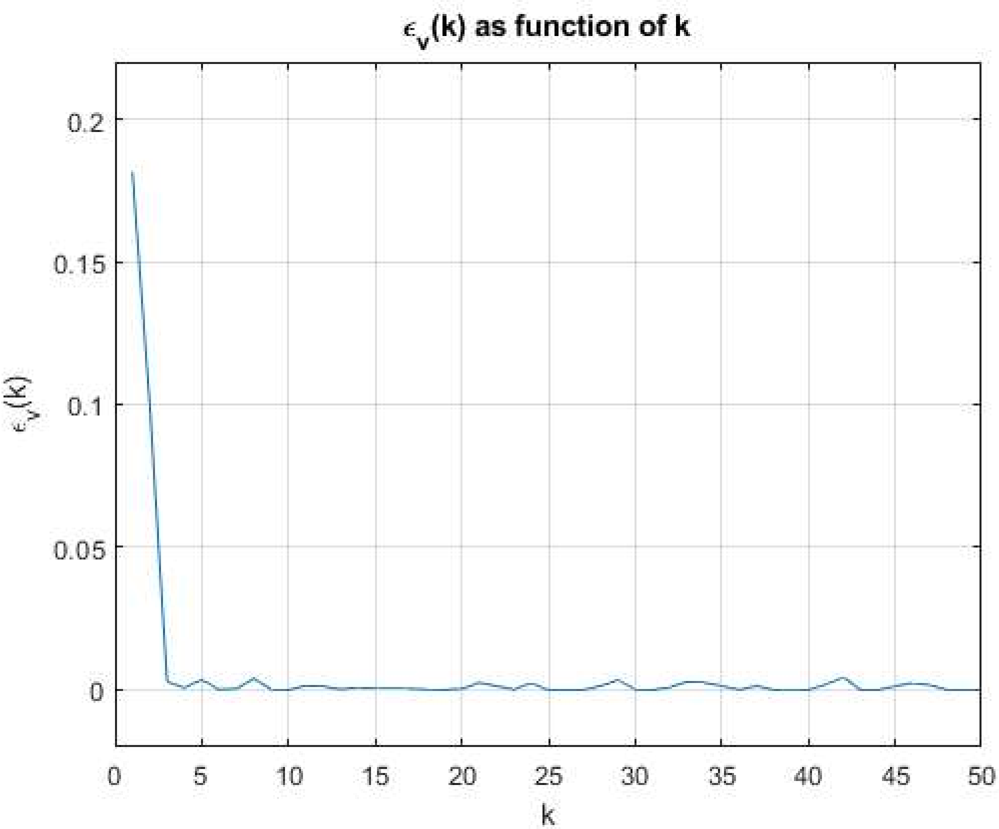
results

```
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( P11_arr ), 'r--');
        plot( thist0, xhat_arr(:,1) - sqrt( P11_arr ), 'r--');
        title('$\hat{x}$(1)', 'interpreter', 'latex');
        legend('$\hat{x}$', '$ \hat{x} \pm \sigma_{11}$', 'interpreter', 'latex', 'location', 'best');
        ylabel('state units');
        bigger_ylim
    subplot(2,1,2)
        plot( thist0, xhat_arr(:,2), '.' ); hold on; grid on;
        plot( thist0, xhat_arr(:,2) + sqrt( P22_arr ), 'r--');
        plot( thist0, xhat_arr(:,2) - sqrt( P22_arr ), 'r--');
        title('$\hat{x}$(2)', 'interpreter', 'latex');
        legend('$\hat{x}$', '$ \hat{x} \pm \sigma_{22}$', 'interpreter', 'latex', 'location', 'best');
        ylabel('state units');
        bigger_ylim
    xlabel('time');
    sgtitle(ftitle);

ftitle = 'e_v(k)';
figure('name', ftitle);
    plot(e_v);
    bigger_ylim
    ylabel('\epsilon_v(k)')
    xlabel('k');
    title('\epsilon_v(k) as function of k')

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

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

```
xhat(50) =
        0.0403586781964148        -0.455082764849298

P(50) =
       0.00117513862430904      0.000693974312791607
       0.00069397431279162        0.0777933474981554
```





subfunctions KALMAN FILTER

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

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

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

% 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 priori covar est

    % update
    nu = zhist(k+1) - Hk * xbar;              % innovation
    S  = Hk * Pbar * Hk' + Rk;                % innovation covariance
    W  = Pbar * Hk' * inv(S);                 % Kalman gain
    xhat = xbar + W * nu;                     % 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_cell = {Pbar_cell; 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)];
    nu_arr    = [nu_arr; nu];
    S_arr     = [S_arr; S];

end

end
```

Published with MATLAB® R2020a