HW6 Problem 2

Spencer Freeman

AOE 5784

11/21/2024

Results:

HW6-P2

Filter Problem a

xhat(50):
1.0e+05 *

-2.1063
-0.0529
-0.0007

phat(50):
8.6366 -0.7391 -6.4165
-0.7391 1.2292 -0.5956
-6.4165 -0.5956 8.6089

xhat_SRIF(50):
1.0e+05 *

-2.1063
-0.0529
-0.0007

phat_SRIF(50):
8.6366 -0.7391 -6.4165
-0.7391 1.2292 -0.5956
-6.4165 -0.5956 8.6089

covariance error metric (50):
1.0e-09 *

0.0280 0.2491 0.0196
0.2716 0.0757 0.0251
0.0209 0.0249 0.0120

Filter Problem b

xhat(50):
1.0e+04 *

-4.0118

```
   0.0254
   0.0022


phat(50):
   6.5794  -0.6665  -5.2465
  -0.6665   0.7258  -0.7852
  -5.2465  -0.7852   6.8169


xhat_SRIF(50):
  1.0e+04 *

  -4.0118
   0.0254
   0.0022


phat_SRIF(50):
   6.5794  -0.6665  -5.2465
  -0.6665   0.7258  -0.7852
  -5.2465  -0.7852   6.8169


covariance error metric (50):
  1.0e-06 *

   0.1397   0.1292   0.1423
   0.1292   0.1412   0.1515
   0.1423   0.1515   0.1444
```

Code:

```
%% Implement a Kalman filter for the example problem that was presented in class
% Spencer Freeman, 11/21/2024
% AOE 5784, Estimation and Filtering
%
% This script solves number 2 of problem set 6
% ------------------------------------------------------------------------
clear;clc;close all

disp('HW6-P2')


%%

kf_example03a % bring in data

[ts, xhats_a, phats_a, ~] = ...
   filter(Fk, Gammak, Hk, Qk, Rk, xhat0, P0, thist, zhist);

[~, xhats_s_a, phats_s_a, ~] = ...
   SRIF(Fk, Gammak, Hk, Qk, Rk, xhat0, P0, thist, zhist);

cov_error_metric_a = abs(phats_a - phats_s_a) ./ (abs(phats_s_a) + eps^6);

kf_example03b % bring in data

[~, xhats_b, phats_b, ~] = ...
   filter(Fk, Gammak, Hk, Qk, Rk, xhat0, P0, thist, zhist);

[~, xhats_s_b, phats_s_b, ~] = ...
   SRIF(Fk, Gammak, Hk, Qk, Rk, xhat0, P0, thist, zhist);

cov_error_metric_b = abs(phats_b - phats_s_b) ./ (abs(phats_s_b) + eps^6);


%% plotting
close all

plot_filter(ts, xhats_a, phats_a, 'KF-a')
plot_filter(ts, xhats_s_a, phats_s_a, 'SRIF-a')
plot_filter(ts, xhats_b, phats_b, 'KF-b')
plot_filter(ts, xhats_s_b, phats_s_b, 'SRIF-b')


%%
ind = 50;

disp(" ")
disp('Filter Problem a')
disp(" ")
disp("xhat(50): ")
disp((xhats_a(:, ind)))
disp(" ")
disp("phat(50): ")
disp((reshape(phats_a(:, ind), size(P0))))
disp(" ")
disp("xhat_SRIF(50): ")
disp((xhats_s_a(:, ind)))
```

```
disp(" ")
disp("phat_SRIF(50): ")
disp((reshape(phats_s_a(:, ind), size(P0))))
disp(" ")
disp("covariance error metric (50): ")
disp((reshape(cov_error_metric_a(:, ind), size(P0))))
disp(" ")


disp("Filter Problem b")
disp(" ")
disp("xhat(50): ")
disp((xhats_b(:, ind)))
disp(" ")
disp("phat(50): ")
disp(reshape(phats_b(:, ind), size(P0)))
disp(" ")
disp("xhat_SRIF(50): ")
disp((xhats_s_b(:, ind)))
disp(" ")
disp("phat_SRIF(50): ")
disp((reshape(phats_s_b(:, ind), size(P0))))
disp(" ")
disp("covariance error metric (50): ")
disp((reshape(cov_error_metric_b(:, ind), size(P0))))
disp(" ")

function plot_filter(ts, xhats, phats, name)

h = figure;
h.WindowStyle = 'Docked';

subplot(3, 1, 1)
plot(ts, xhats(1, :), 'r*'); hold on
plot(ts, xhats(1, :) + sqrt(phats(1, :)) .* [1; -1], 'bo')
grid on
legend('Estimate', '+-1\sigma')
title(name)
ylabel('xhat_1')

subplot(3, 1, 2)
plot(ts, xhats(2, :), 'r*'); hold on
plot(ts, xhats(2, :) + sqrt(phats(5, :)) .* [1; -1], 'bo')
grid on
ylabel('xhat_2')

subplot(3, 1, 3)
plot(ts, xhats(3, :), 'r*'); hold on
plot(ts, xhats(3, :) + sqrt(phats(9, :)) .* [1; -1], 'bo')
grid on
ylabel('xhat_2')

xlabel('Time (s)')

end


function [ts, xhats, phats, evs] = ...
   SRIF(Fk, Gammak, Hk, Qk, Rk, xhat0, P0, thist, zhist)

n = length(thist) + 1;
```

```
nx = length(xhat0);
nv = size(Qk, 1);

t = 0; % s
ev = 0;

Finv = inv(Fk);
info = inv(P0); % information matrix
Rxx = chol(info);
Rvv = inv(chol(Qk))';
Ra = chol(Rk);
Rainv = inv(Ra);
Ha = Rainv' * Hk;
zahist = Rainv' * zhist;
zx = Rxx * xhat0;

G = 0;
u = 0;

xhat = xhat0; % initial state estimate
phat = P0; % initial state covariance

ts = nan(1, n);
xhats = nan(nx, n);
phats = nan(nx * nx, n);
evs = nan(1, n);
for i = 1:(n - 1)

  ts(i) = t;
  xhats(:, i) = xhat;
  phats(:, i) = phat(:); % unwrap to column vector
  evs(i) = ev;

  t = thist(i); % s

  % propagation
  [q, r] = qr([ ...
    Rvv, zeros(nv, nx); ...
    -Rxx * Finv * Gammak, Rxx * Finv]);

  Ta = q';
  Rvvbar = r(1:nv, 1:nv);
  Rvxbar = r(1:nv, nv + 1:end);
  Rxxbar = r(nv + 1:end, nv + 1:end);

  temp = Ta * [zeros(nv, 1); zx];
  zxbar = temp(nv + 1:end, :);

  % measurement update
  [q, r] = qr([Rxxbar; Ha]);

  Tb = q';
  Rxx = r(1:nx, :);

  temp = Tb * [zxbar; zahist(i)];
  zx = temp(1:nx, :);

  % convert back to true states
  Rxxinv = inv(Rxx);
  xhat = Rxxinv * zx;
  phat = Rxxinv * Rxxinv';
```

```
end

% record the final filter outputs
ts(n) = t;
xhats(:, n) = xhat;
phats(:, n) = phat(:); % unwrap to column vector
evs(n) = ev;

end % function

function [ts, xhats, phats, evs] = ...
  filter(Fk, Gammak, Hk, Qk, Rk, xhat0, P0, thist, zhist)

n = length(thist) + 1;
nx = length(xhat0);

t = 0; % s
xhat = xhat0; % initial state estimate
phat = P0; % initial state covariance
ev = 0;

ts = nan(1, n);
xhats = nan(nx, n);
phats = nan(nx * nx, n);
evs = nan(1, n);
for i = 1:(n - 1)

  ts(i) = t;
  xhats(:, i) = xhat;
  phats(:, i) = phat(:); % unwrap to column vector
  evs(i) = ev;

  t = thist(i); % s
  xbar = Fk * xhat; % propagate state estimate
  pbar = Fk * phat * Fk' + Gammak * Qk * Gammak'; % propagate state covariance

  zbar = Hk * xbar; % expected measurement
  z = zhist(i); % actual measurement
  v = z - zbar; % filter innovation

  S = Hk * pbar * Hk' + Rk; % expected measurement covariance
  Sinv = inv(S);
  W = pbar * Hk' * Sinv; % filter gain

  ev = v' * Sinv * v; % estimation error statistic

  xhat = xbar + W * v; % updated state estimate
  phat = pbar - W * S * W'; % updates state covariance

end

% record the final filter outputs
ts(n) = t;
xhats(:, n) = xhat;
phats(:, n) = phat(:); % unwrap to column vector
evs(n) = ev;

end % function
```