

```
%% Problem 4
clc; close all; clearvars; set(groot,"defaultTextInterpreter","latex");
rng(0,"v4");

% Simulation variables
dt = 0.1; % Process time step, s
dtMeas = 0.1;
tNextMeas = dtMeas;
meas_available = 0;
tMax = 20; % Max time, s
makeplot = 1;
Res = 3; L = 1; C = 0.5;

% Kalman Filter variables
var_init_uncertainty = [0; 0];
var_meas_noise = 1;
var_process_noise = [0; 1];

n = length(var_init_uncertainty);
m = length(var_meas_noise);

R = diag(var_meas_noise); % mxm Measurement noise matrix
P = diag(var_init_uncertainty); % nxn Covariance matrix
Q = diag(var_process_noise); % nxn

A = dt*[0, 1/C; -1/L, -Res/L];
PHI = eye(n) + A; % State transition matrix
H = [1 0]; % Measurement matrix

x0_true = [0; 0];
x0_est = [0; 0];

% For storing process
saveVars = {"T", "X_true", "X_est", "Z_true", "Z_est", "P_est", "P_plot", "K_plot", "P_lim", "K_lim", "L_lim", "info", "makeplot"};

T = 0:dt:tMax;
t_length = length(T);
X_true = nan(n,t_length); % True state vectors (n x steps)
X_est = nan(n,t_length); % Estimate state vectors (n x steps)
Z_true = nan(m,t_length); % True measurement vectors (m x steps)
Z_est = nan(m,t_length); % Estimate measurement vectors (m x steps)
P_est = nan(n,n,t_length); % Estimate variance vectors (n x n steps)
P_plot = nan(n,t_length);
K_plot = nan(n,t_length);
xtrue = x0_true;
xest = x0_est;
[P_lim,K_lim,L_lim,info] = dare(A,H',Q,R,[],[]);
for i = 1:length(T)
    t = T(i);
```

```

if t>=tNextMeas
    tNextMeas = t+dtMeas;
    meas_available=1;
else
    meas_available=0;
end

if meas_available == 1
    X_true(:,i) = xtrue;
    X_est(:,i) = xest;
    Z_true(:,i) = H*X_true(:,i) + sqrt(R)*normrnd(0,1,m,1);
    Z_est(:,i) = H*X_est(:,i);
    P_est(:,:,i) = P;
    P_plot(:,i) = diag(P_est(:,:,i));

    % Gain Matrix
    K = P_est(:,:,i)*H'/(H*P_est(:,:,i)*H' + R);

    % States updated with measurement information
    X_est(:,i) = X_est(:,i) + K*(Z_true(:,i) - Z_est(:,i));

    % Covariance matrix updated with measurement information
    I = eye(n);
    P_est(:,:,i) = (I - K*H)*P_est(:,:,i);
    K_plot(:,i) = K;
    P_plot(:,i) = diag(P_est(:,:,i));
else
    X_true(:,i) = xtrue;
    X_est(:,i) = xest;
    Z_true(:,i) = nan(m,1);
    Z_est(:,i) = nan(m,1);
    P_est(:,:,i) = P;
    K_plot(:,i) = nan(n,1);
    P_plot(:,i) = diag(P_est(:,:,i));
end

t = t+dt;

xtrue = PHI*X_true(:,i) + Q*normrnd(0,1,n,1);

% STATE propagation
xest= PHI*X_est(:,i);
P = PHI*P_est(:,:,i)*PHI' + Q;

end

clearvars ("-except",saveVars{:})

if makeplot

```

```

%% Estimate plots
figure()
est_plot = tiledlayout(2,1);
title(est_plot, "Circuit State Estimation");
xlabel(est_plot, "Time(s)");

nexttile
plot(T(1:2:end),X_true(1,1:2:end), "k",T(1:2:end),X_est(1,1:2:end), "r");
ylabel("Capacitor Voltage")
legend("$V_c$", "$\hat{V}_c$", "Location", "north", "interpreter", "latex")

nexttile
plot(T(1:2:end),X_true(2,1:2:end), "k",T(1:2:end),X_est(2,1:2:end), "r");
ylabel("Current");
legend("$I$", "$\hat{I}$", "Location", "north", "interpreter", "latex")

%% Error plots
figure()
err_plot = tiledlayout(2,1);
title(err_plot, "Circuit Error");
xlabel(err_plot, "Time(s)");

% nexttile
% plot(T(1:2:end),X_est(1,1:2:end)-X_true(1,1:2:end), "r", ...
%      T(1:2:end),sqrt(P_plot(1,1:2:end)), "b", T(1:2:end), -sqrt(P_plot(1,1:2:end)), ↵
"b");
% ylabel("Capacitor Voltage")

nexttile
plot(T(1:2:end),X_est(2,1:2:end)-X_true(2,1:2:end), "r", ...
     T(1:2:end),sqrt(P_plot(2,1:2:end)), "b", T(1:2:end), -sqrt(P_plot(2,1:2:end)), ↵
"b");
ylabel("Current Post")

nexttile
plot(T(2:2:end),X_est(2,2:2:end)-X_true(2,2:2:end), "r", ...
     T(2:2:end),sqrt(P_plot(2,2:2:end)), "b", T(2:2:end), -sqrt(P_plot(2,2:2:end)), ↵
"b");
ylabel("Current Prio")
end

```