# Opt Reg 7

## Problem 1

(1) $P = Q + A^T P (I + BR^{-1} B^T P)^{-1} A$

(2) $A^T P A - P - A^T P B (R + B^T P B)^{-1} B^T P A + Q = 0$

Using

$(S + UTV)^{-1} = S^{-1} - S^{-1} U (T^{-1} + V S^{-1} U)^{-1} V S^{-1}$

with $B = U$, $R^{-1} = T$, $B^T P = V$

on the expression

$(I + BR^{-1} B^T P)^{-1}$

we get

$$(I + BR^{-1} B^T P)^{-1} = I - I B ((R^{-1})^{-1} + B^T P B)^{-1} B^T P$$

$$= I - B (R + B^T P B)^{-1} B^T P$$

and inset the result into (1)

$$P = Q + A^T P (I - B (R + B^T P B)^{-1} B^T P) A$$

$$= Q + A^T P A - A^T P B (R + B^T P B) B^T P A$$

$$\Rightarrow A^T P A - P - A^T P B (R + B^T P B) B^T P A + Q = 0 = (2)$$

# Problem 2

## a)

By using dlqr on the system, we get $K = [1.0373, 1.6498]$ and the poles $E = [0.8675 + 0.0531i, 0.8675 - 0.0531i]$

## b)

The result of using the estimated state as feedback is shown in Figure 1. The estimated states are plotted as dashed lines. Notice that the estimators converge quickly and is therefore suitable to be used as feedback for the controller. The response is good, thus I chose not to tune the controller. The code is as shown below

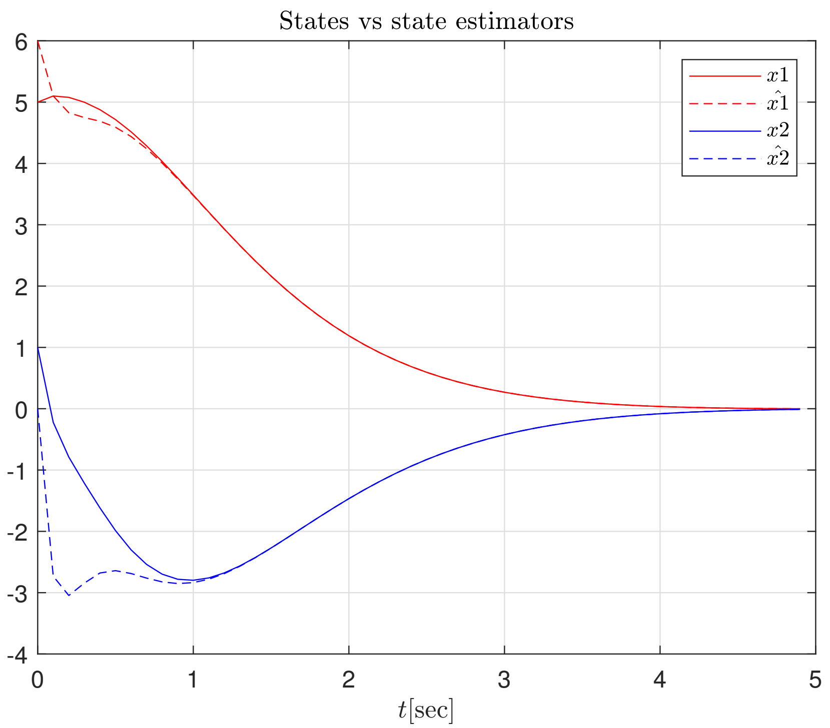


Figure 1: States and state estimators

```
%% Constants and system matrices
k1 = 1; k2 = 1; k3 = 1; T = 0.1;
A = [0  1;
     -k2 -k1];
B = [0  k3]';
```

```
C = [1  0];

x0 = [5  1]'; x0_hat = [6  0]';

%Discretized

Ad = eye(2) + A*T;
Bd = B*T;

%% Solve problem
Q = [4  0;
     0  4];
R = 1;
[K, P, E] = dlqr(Ad, Bd, Q, R);
K_f = place(Ad', C', [0.5 + 0.03j, 0.5 - 0.03j]')';

%% Simulate
N = 50; %timesteps
nx = 2; %state dimension
nu = 1; %input dimension
t = 0:T:(N-1)*T;

%Dynamics
x_tnext = @(x_t, x_t_hat) Ad*x_t - Bd*K*x_t_hat;
x_tnext_hat = @(x_t_hat, x_t) Ad*x_t_hat - Bd*K*x_t_hat + K_f*(C*x_t - C*x_t_hat);

x = zeros(N*nx, 1);
x_hat = zeros(N*nx, 1);
u = zeros(N*nu, 1);

%Start conditions
x(1:nx) = x0;
x_hat(1:nx) = x0_hat;

for i = nx+1:nx:N*nx
    x(i:i+nx-1) = x_tnext(x(i-nx:i-1), x_hat(i-nx:i-1));
    x_hat(i:i+nx-1) = x_tnext_hat(x_hat(i-nx:i-1), x(i-nx:i-1));
end
%Extract solution
x1 = x(1:nx:N*nx);
x2 = x(2:nx:N*nx);
```

```
x1_hat = x_hat(1:nx:N*nx);
x2_hat = x_hat(2:nx:N*nx);

%% Plots
figure(1)
plot(t, x1, 'r', 'DisplayName', '$x1$')
hold on; grid on;
plot(t, x1_hat, '--r', 'DisplayName', '$\hat{x1}$')
plot(t, x2, 'b', 'DisplayName', '$x2$')
plot(t, x2_hat, '--b', 'DisplayName', '$\hat{x2}$')
legend('Interpreter', 'Latex')
title('States vs state estimators', 'Interpreter', 'Latex')
xlabel('$t$[sec]', 'Interpreter', 'Latex')
```

## c)

$$\xi_{t+1} = \begin{bmatrix} x_{t+1} \\ \tilde{x}_{t+1} \end{bmatrix} = \begin{bmatrix} A - BK & BK \\ 0 & A - K_f C \end{bmatrix} \xi_t = \underbrace{\begin{bmatrix} 1 & 0.1 & 0 & 0 \\ -0.2037 & 0.7350 & 0.1037 & 0.1650 \\ 0 & 0 & 0.1 & 0.1 \\ 0 & 0 & -1.6090 & 0.9 \end{bmatrix}}_{\Phi} \xi_t \tag{1}$$

# Problem 3

## a)

With an MPC based on the estimator to provide the input for the system, the response becomes as shown in Figure 2.

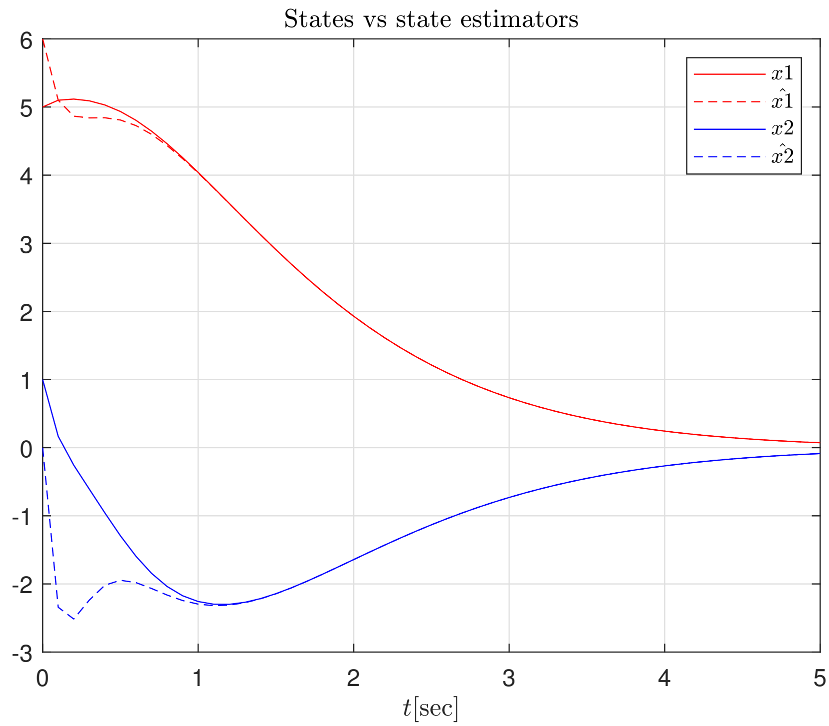


Figure 2: Response with estimator based MPC providing optimal input

## b)

When using the state feedback directly without estimation, the response becomes as shown in Figure 3

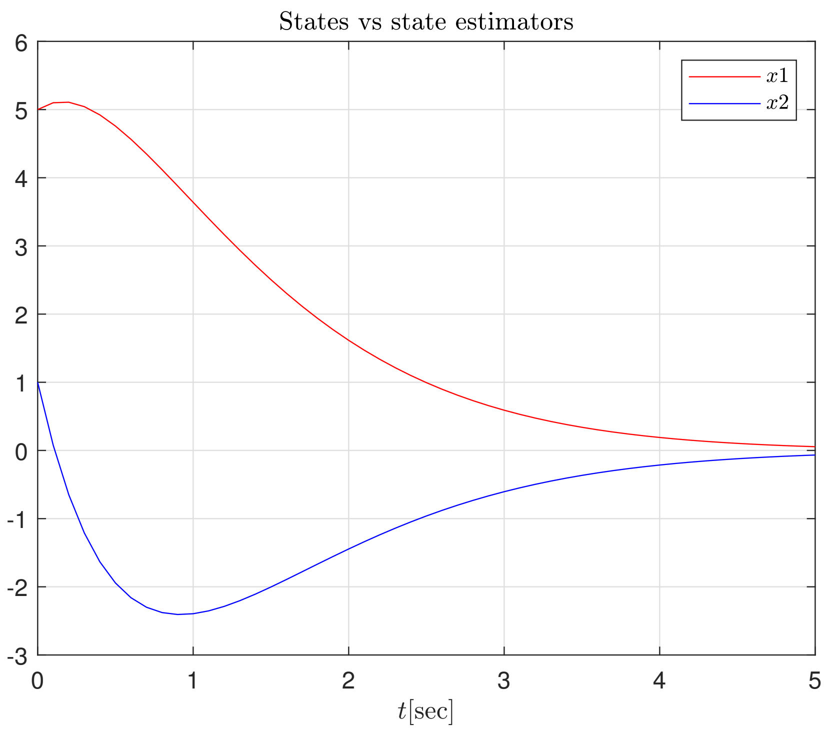


Figure 3: Response with direct state feedback MPC providing input

# Problem 4

## a)

Using dlqr from problem 2, we get

$$P = \begin{bmatrix} 27.5170 & 7.2713 \\ 7.2713 & 10.2339 \end{bmatrix} \tag{2}$$

## b)

With $2P$ inserted instead of the last $Q$ in the G-matrix, the response becomes as shown in Figure 4 With a MPC in closed loop we only use the first input for every iteration, so the

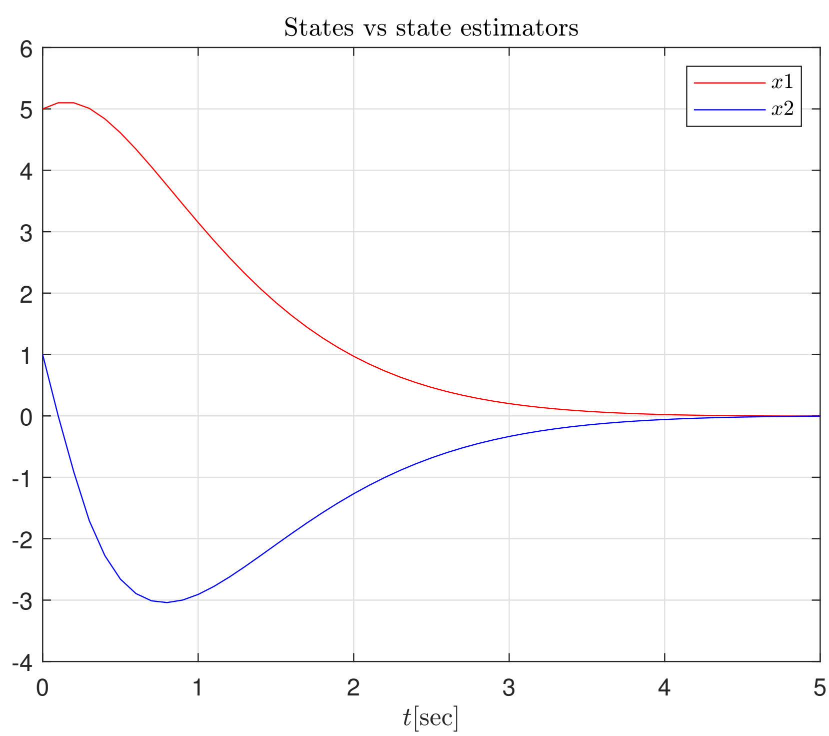


Figure 4: Modified objective function response

length of the horizon, N, has a very small effect on the response. With a finite-horizon LQ controller, however, the effect of a too small N would have a huge effect on the controller, since P is far from stationary.