FRTN15 Predictive Control—Home Work 1
Signals and Systems

In this homework exercise we recapitulate theory for discrete time signals and systems in assignments 1-3. Recursive Least square estimation (RLS) is treated in assignment 4. The exercise also gives the opportunity to practice Matlab/Simulink.

E-mail your detailed and motivated solutions in pdf-format to jerker@control.lth.se. Attach any Matlab code or Simulink models you might have used.

1.

a. The observer canonical state space form of the continuous time system is given by

$$\begin{aligned}\frac{dx}{dt} &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u \quad (= Ax + Bu) \\ y &= \begin{pmatrix} 1 & 0 \end{pmatrix} x \quad (= Cx).\end{aligned}$$

Hence, with sampling period h one obtains

$$\begin{aligned}x(kh + h) &= \Phi x(kh) + \Gamma u(kh) \\ y(kh) &= \begin{pmatrix} 1 & 0 \end{pmatrix} x(kh)\end{aligned}$$

where

$$\begin{aligned}\Phi &= e^{Ah} = I + Ah + A^2h^2/2 + \cdots = \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & h \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}.\end{aligned}$$

and

$$\Gamma = \int_0^h \begin{pmatrix} 1+s \\ 1 \end{pmatrix} ds = \begin{pmatrix} h + \frac{h^2}{2} \\ h \end{pmatrix}.$$

The transfer function representation is given by

$$\begin{aligned}H_{y,u}(z) &= C(zI - \Phi)^{-1}\Gamma = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} z-1 & -h \\ 0 & z-1 \end{pmatrix}^{-1} \begin{pmatrix} h + \frac{h^2}{2} \\ h \end{pmatrix} = \\ &= \frac{(z-1)(h + h^2/2) + h^2}{(z-1)^2}\end{aligned}$$

b.
```
s=tf('s');
Gc=(s+1)/s^2;
h=1;
Gd=c2d(G,h)
[Phi,Gamma,C,D]=tf2ss(Gd.num{1},Gd.den{1})

Transfer function:
```

```
0.105 z - 0.095
---------------
 z^2 - 2 z + 1
```

If you are uncertain what a command does, type `help` followed by the command name in the Matlab prompt.

c. The sampled model behaves like its continuous time counterpart up to a higher frequencies when sampled more rapidly (h smaller). However, decreasing the sampling period leads to higher hardware demands and introduces numerical sensitivity when implemented on finite word length machines.

2. Simple algebra yields

$$H_1H_2(U + H_3Y) = Y \Rightarrow Y = HU = \frac{H_1H_2}{1 - H_1H_2H_3}U,$$

i.e.

$$H(z) = \frac{z+2}{z^2+z+1}.$$

To simulate the step response you can use this code

```
h = 1;
z = tf('z',h);
H1 = z+2;
H2 = 1/(z^2+2*z+1);
H3=z/(z+2);
clsys = H1*H2/(1-H1*H2*H3);  % positive feedback!
[sys] = minreal(clsys)   % compute minimial realization (nicer)
[y,t] = step(sys,20)
plot(t,y)
```

3.

a. A discrete time LTI is stable iff the eigenvalues of Φ are strictly inside the unit circle. The system is hence not asymptotically stable. ($1 \in \mathrm{Sp}(\Phi)$.)

b. Let the linear state feedback be described by $u = -Kx$, i.e.

$$x(kh+h) = (\Phi - \Gamma K)x.$$

Inserting numerical values of K, Γ and solving

$$\mathrm{Sp}(\Phi - \Gamma K) = \{0, 0\}$$

for K yields

$$K = \begin{pmatrix} \frac{1}{h^2} & \frac{1.5}{h} \end{pmatrix} = \begin{pmatrix} 100 & 15 \end{pmatrix}$$

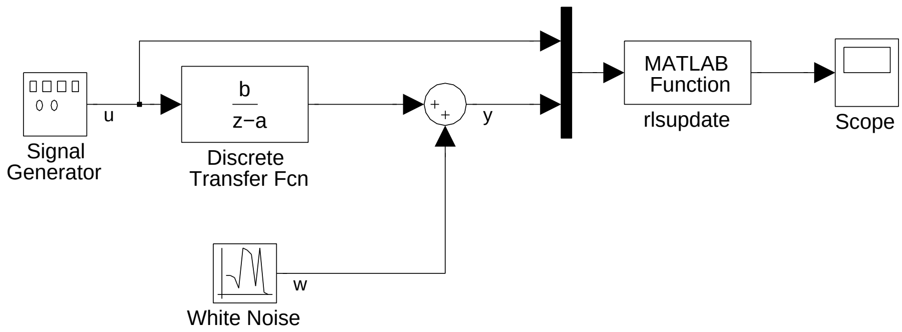
In Matlab the `acker` command can be used to achieve this:

```
Phi=[1 h;0 1];
Gamma=[h^2/2 h]';
K=acker(Phi,Gamma,[0 0])
```

c. Any initial state $x(0)$ will be controlled to the origin in 2 time steps. This looks good, might require very large control signals, especially if the sample period is small. Think of moving a heavy robot arm to a wanted position in $h = 1$ second compared to in $h = 0.001$ second. Dead-beat control can be realistic in the former case and unrealistic in the latter.

4.

a. This can be done in many ways. One possibility is to use the simulink model depicted below: The following settings have been applied:



- 'Configuration Parameters' from the 'Simulation menu': 'Stop time' under 'Simulation time' changed to '300', 'Solver' under 'Solver options' changed to 'Discrete (no continuous states)'
- 'Signal generator': 'Wave form' changed to 'square', 'Frequency' changed to '`1/uP`'
- 'MATLAB Function': 'Matlab function' changed to '`rlsupdate`'

System parameters were defined as follows:

```
h = 1.0;     % sample period [s]
a = 0.9;
b = 0.1;
uA = 1.0;    % amplitude of input (square)
uP = 75*2;   % period of input (square) [s]
```

The RLS update function was defined:

```
function thetaHat = rlsupdate(uy)
persistent P theta phi
if isempty(P)
    P = eye(2);
    theta = [0 0]';
    phi = [0 0]';
end
u = uy(1);
y = uy(2);
P = P-(P*phi*phi'*P)/(1+phi'*P*phi);
e = y - phi'*theta;
theta = theta + P*phi*e;
```

```
thetaHat = theta;
phi = [y u]';
return
```

To estimate the parameters correctly, we need a signal which excites the system dynamics enough. For instance if $u = 0$ we will not get any information about b, and if $y = 0$ we get no information about a either.

b. Augment it with exponential forgetting. The more rapidly varying parameter one wants to track, the smaller λ one should use.