Chapter 11
Analysis of Sampled-Data Systems

11.1 Discrete-Time Systems

11.1.1 z-Transforms

In sampled-data systems, the signals are handled and stored in digital form. In these systems continuous-time signals are sampled periodically at sampling instants and their processing takes place in the discrete-time domain. Thus, the signals available in continuous-time control systems should be sampled and converted to digital form. Denote the sampling time by T_s. Then the digital form of a sampled continuous-time signal $y(t)$ can be represented by a train of impulses as follows:

$$\begin{aligned} y_d[k] &= y(t = kT_s) \\ &= y(0)\delta(t) + y(T_s)\delta(t - T_s) + y(2T_s)\delta(t - 2T_s) + y(3T_s)\delta(t - 3T_s) + \cdots . \end{aligned}$$

The LAPLACE transform of $y(t = kT_s) = y_d[k]$ is

$$Y_d(s) = y(0) + y(T_s)e^{-sT_s} + y(2T_s)e^{-2sT_s} + y(3T_s)e^{-3sT_s} + \cdots .$$

Introduce the notations $z = e^{sT_s}$ and $z^{-1} = e^{-sT_s}$, where $z = e^{sT_s}$ is the operator of the *z-transformation*. Then z^{-1} can be interpreted as a backward shift operator. Using the introduced notation, we have

$$Y_d(z) = y(0) + y(T_s)z^{-1} + y(2T_s)z^{-2} + y(3T_s)z^{-3} + \cdots = \sum_{k=0}^{\infty} y(kT_s)z^{-k}.$$

L. Keviczky et al., *Control Engineering: MATLAB Exercises*,
Advanced Textbooks in Control and Signal Processing,
https://doi.org/10.1007/978-981-10-8321-1_11

Example 11.1 Suppose given the *z-transform* of a signal as follows:

$$Y_\mathrm{d}(z) = \frac{2z^2 - z}{z^2 - z + 0.24}; \quad T_\mathrm{s} = 0.5.$$

Use MATLAB™ to determine the first 5 samples of the inverse *z-transform* of this signal.

(a) *Numerical calculation*

```
num=[2, -1, 0]
den=[1, -1, 0.24]
yd=dimpulse(num,den,5)
plot(yd,'*')
plot(1:5,yd,'*')
```

The data for this example can also be given in symbolic form using the *LTI sys* structure. Similarly to the way s was defined, a z variable can be defined by $z = e^{sT_\mathrm{s}}$. However, first the sampling time T_s should be specified:

```
Ts=0.5
z=tf('z',Ts)
Y=(2*z^2-z)/(z*z-z+0.24)
```

One advantage of using the *LTI sys* structure is that formally identical commands can be applied both for continuous-time and discrete-time systems. In the case of zero sampling time sys.Ts MATLAB™ considers the system as a continuous-time system, otherwise as a discrete-time system. The help command describes the exact usage of the impulse command:

```
help impulse
```

IMPULSE(SYS, TFINAL) simulates the impulse response of the system in the time range from $t = 0$ till the final time $t = \mathrm{TFINAL} = 2$. $\mathrm{TFINAL} = 2$ is the time required by the conditions of $T_s = 0.5$ and number of the samples (5). The impulse command essentially divides the numerator by the denominator and this is the way it provides the samples of the impulse response:

```
impulse(Y,2);
yd=impulse(Y,2);
plot(Ts*(0:4),yd,'*');
```

(b) *Partial fraction expansion*

Partial fractional form allows recognizing the components in the discrete time-domain. Then these components can simply be added up.

E.g. if an exponential component looks like

$$y(t) = e^{-at}; \quad Y_\mathrm{d}[nT_\mathrm{s}] = e^{-anT_\mathrm{s}} \xrightarrow{z} \frac{z}{z - e^{-aT_\mathrm{s}}},$$

then the partial fraction form contains components as follows:

$$Y_\mathrm{d}(z) = \frac{num}{den} = z\frac{num1}{den} = z\left(k_\mathrm{o} + \frac{r_1}{z - p_1} + \frac{r_2}{z - p_2}\right)$$

Then using the MATLAB™ command residue,

```
num1=[2, -1]
den=[1, -1, 0.24]
[r,p,k0]=residue(num1,den)
    r =
        1
        1
    p =
       0.6000
       0.4000
    k0 =
        []
Ts=0.5
```

$$Y_\mathrm{d}(z) = z\left(\frac{r_1}{z - p_1} + \frac{r_2}{z - p_2}\right) = z\left(\frac{1}{z - 0.6} + \frac{1}{z - 0.4}\right) = \frac{z}{z - 0.6} + \frac{z}{z - 0.4}$$

$$y_\mathrm{d}[k] = \{y(kT_\mathrm{s})\} = \left\{e^{akT_\mathrm{s}} + e^{bkT_\mathrm{s}}\right\}, \text{ where } e^{aT_\mathrm{s}} = 0.6; e^{bT_\mathrm{s}} = 0.4$$

```
a=log(0.6)/Ts
b=log(0.4)/Ts
```

(In MATLAB™ the command log means the calculation of the mathematical ln function.)

$$a = \frac{\ln(0.6)}{T_\mathrm{s}} = -1.02, \ b = \frac{\ln(0.4)}{T_\mathrm{s}} = -1.83$$

$$y(t) = e^{-1.02t} + e^{-1.83t}, \ \mathrm{t} \geq 0.$$

```
t=[0:Ts:2]'
yd=exp(a*t)+exp(b*t)
plot(t,yd,'*');
```

11.1.2 Discrete-Time Impulse Response and Pulse Transfer Function

The discrete-time (pulse) transfer function describes the relation between the sampled output of a continuous-time process and the input of a holding unit driving the process (see Fig. 11.1). In practice, typically zero order holding (ZOH) units realized by analog-to-digital (A/D) converters are applied.

Example 11.2 Find the discrete-time pulse transfer function of the continuous process given by

$$P(s) = \frac{1}{(1+5s)(1+10s)}; \quad T_s = 2.5$$

supposing zero-order holding.

```
s=zpk('s')
Ps=1/((1+5*s)*(1+10*s))
```

Alternatively, the process can also be displayed in pole-zero form:

$$P(s) = \frac{0.02}{(s+0.2)(s+0.1)}$$

```
[zerof,polef,kf]=zpkdata(Ps,'v')
```

The process poles are: −0.2 and −0.1. Note that the process has no zeros.

MATLAB™ offers the c2d (read as continuous-to-discrete) command to derive the discrete-time pulse transfer function: sysd = c2d(sysc, ts, method), where ts stands for the sampling time and method defines the type of the holding unit. The *default* method is 'zoh', i.e. zero-order holding, so it can be omitted.

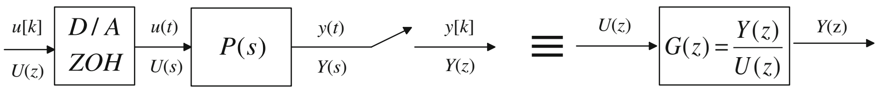


Fig. 11.1 Sampling and holding

```
Ts=2.5;
Gz=c2d(Ps,Ts,'zoh')
```

or alternatively:

```
Gz=c2d(Ps,Ts)
```

$$G(z) = 0.048929 \frac{(z + 0.7788)}{(z - 0.7788)(z - 0.6065)}$$

Type

```
[zerod,poled,kd]= zpkdata(Gz,'v')
```

to get the discrete-time zeros and poles. The discrete-time poles are 0.7788 and 0.6065, while the discrete-time zero is −0.7788. Note that the discrete-time poles can be obtained from the continuous-time poles using the relation $z = e^{sT_s}$:

```
exp(-0.2*2.5)
    0.6065
```

Similarly

```
exp(polef*Ts)
```

can be used. Note that no similar simple relation exists between the continuous-time and discrete-time zeros.

Also note, that while the continuous-time process in the example has no zero, the discrete form still has one. This is because the sampling procedure results in subsidiary zeros. Processes of lag type exhibit several zeros, namely the number of discrete-time zeros will be less by one than the number of discrete-time poles.

11.1.3 Initial Value and Final Value Theorems

Once its z-transform is given, the initial value of a discrete-time signal can be calculated by:

$$\lim_{k \to 0} y(kT_s) = \lim_{z \to \infty} Y(z),$$

while the final value is obtained by

$$\lim_{k \to \infty} y(kT_s) = \lim_{z \to 1} \left(1 - z^{-1}\right) Y(z).$$

Apply a discrete-time unit step to drive the process of the previous example. The z-transform of the discrete-time unit step is $z/(z-1)$. Find the initial and final values of the sampled output.

$$Y(z) = \frac{z}{z-1} G(z)$$

The initial value of the sampled output can be obtained by $\lim_{z\to\infty} Y(z)$, which happens to be zero in this case (the degree of the denominator is higher than the degree of the numerator). The final value of the sampled output is obtained by substituting $z = 1$ into $G(z)$. This relation is in full harmony with finding the steady-state value of a step response produced by a continuous-time transfer function. MATLAB™ offers the dcgain command to find the dc gain both for continuous-time and discrete-time linear systems. For discrete-time systems ddcgain(numd, dend) can also be used. As far as the dcgain command is concerned, the sampling time will decide whether the system is of continuous-time or discrete-time.

```
Ps.Ts
Gz.Ts
A=dcgain(Ps)
Ad=dcgain(Gz)
```

As expected, both gains—A (CT) and Ad (DT)—will turn out to be 1.

11.1.4 Stability of Sampled-Data Systems

Discrete-time systems are stable if their poles (roots of the characteristic equation) are within the unit circle in the complex plane.

Example 11.3 Check the stability of the discrete-time system given by the following pulse transfer function:

$$G(z) = \frac{z^2 - 0.3z - 0.1}{z^3 + 3z^2 + 2.5z + 1}, T_s = 1$$

Define the discrete-time system for MATLAB™:

```
z=tf('z')
Gz=(z*z-0.3*z-0.1)/(z^3+3*z^2+2.5*z+1)
```

Consider the zeros and poles:

```
[zerod,poled,kd]=zpkdata(Gz,'v')
```

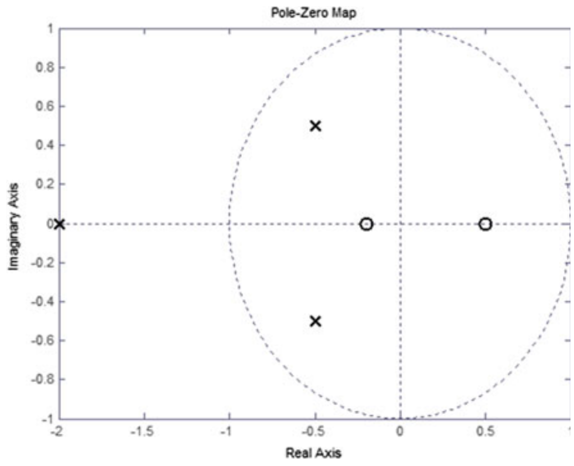


Fig. 11.2 Pole-zero configuration of the pulse transfer function

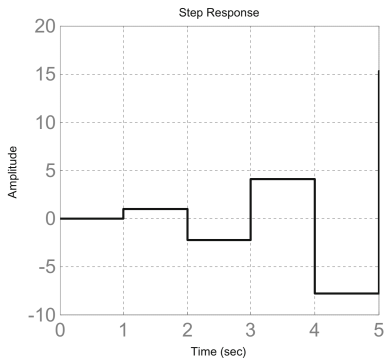


Fig. 11.3 The step response shows instability

The magnitude of the poles:

```
abs(poled)
    ans =
                2.0000
                0.7071
                0.7071
```

Since one pole is outside the unit circle, the system is unstable. Alternatively, this can be shown in graphical form as well (Fig. 11.2).

```
pzmap(Gz)
```

The zgrid command draws the unit circle together with the lines belonging to identical damping values and natural frequencies.

Stability can be checked in the time-domain by displaying the unit step response (see Fig. 11.3).

```
step(Gz),grid
```

Here the output is not bounded, so the system is unstable. Note that though MATLAB™ displays the samples together with zero-order holding, the impulse response of a discrete-time system consists of samples only.

11.2 Analysis of Closed-Loop Sampled-Data Systems

Closed-loop discrete-time systems may exhibit unexpected behaviour, as the system is essentially in open-loop between two samples. Also, the holding unit is an additional factor to modify the closed-loop behaviour. In the case of fast sampling, the continuous-time and discrete-time behaviours are not too far from each other. The following example illustrates the fundamental operation of a closed-loop sampled-data system.

Example 11.4 Assume that the continuous-time process to be controlled is an integrator given by the transfer function $1/s$. Unit negative feedback is applied (see Fig. 11.4) with sampling time T_s. A proportional controller with gain K provides the control signal. To convert the discrete-time control input train to a continuous-time signal, a zero-order holding unit is employed.

Analyze the dynamic behaviour of the closed-loop system and check the stability of the closed-loop system. Derive the difference equation of the closed-loop system and calculate the first 5 samples of the process output. Assume a unit step reference signal. Select $K = 1$ and repeat the analysis for various sampling times like $T_s = 0.5$, 1, 1.5 and 3. Calculate the impulse response transfer function for the closed-loop system and find an analytical form of the sampled output.

Figure 11.5 illustrates the nature of the error signal e and the output signal y.

The difference equation between the sampled output and the error signal is:

$$\begin{aligned} y[kT_s] &= y[(k-1)T_s] + KT_s e[(k-1)T_s] \\ &= y[(k-1)T_s] + KT_s\{r[(k-1)T_s] - y[(k-1)T_s]\} \end{aligned}$$

Here it has been taken into account that the error signal e is the difference between the reference signal r and the sampled output y. Rearranging this equation results in

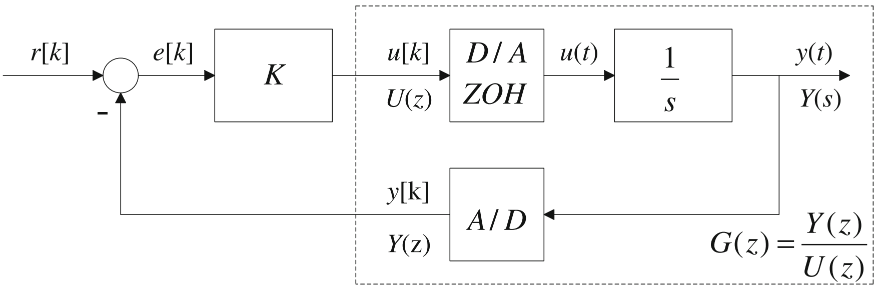


Fig. 11.4 Sampled control system

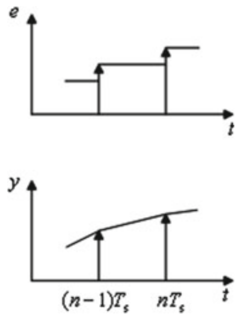


Fig. 11.5 Illustrating the error and the output signals

$$y[kT_s] = (1 - KT_s)y[(k-1)T_s] + KT_s r[(k-1)T_s]$$

or, further

$$\frac{y[kT_s] - y[(k-1)T_s]}{T_s} + Ky[(k-1)T_s] = Kr[(k-1)T_s],$$

which is the discretized version of the following differential equation as $T_s \to 0$:

$$\frac{\mathrm{d}y(t)}{\mathrm{d}t} + Ky(t) = Kr(t).$$

The continuous-time system is structurally stable, however, the discrete-time system exhibits various natures as the sampling time changes. The sequence of the output samples can be evaluated as follows:

For $T_s = 0.5$:	0; 0.5; 0.75; 0.875; 0.9375;	Stable, asymptotically converges to 1.
For $T_s = 1$:	0; 1; 1; 1; 1; 1;	Finite settling time.
For $T_s = 1.5$:	0; 1.5; 0.75; 1.125; 0.9375;	Stable, but oscillates.
For $T_s = 3$:	0; 3; −3; 9; −15;	Unstable.

Note that (unlike the continuous-time version of this problem) the discrete-time version will not be structurally stable.

The pulse transfer function of the closed-loop system is

$$T(z) = \frac{Y(z)}{R(z)} = \frac{\frac{KT_s}{z-1}}{1 + \frac{KT_s}{z-1}} = \frac{KT_s}{z - (1 - KT_s)}.$$

The poles of the closed-loop system remain within the unit circle if

$$|z_1| = |1 - KT_s| < 1.$$

The above condition can also be written as

$$0 < KT_s < 2$$

Introducing a new variable by $b = 1 - KT_s = e^{-aT_s}$, we have

$$Y(z) = \frac{z}{z-1}\frac{KT_s}{z-b}.$$

The partial fraction expansion is

$$Y(z) = KT_s z\left(\frac{\alpha}{z-1} + \frac{\beta}{z-b}\right) = \frac{z}{z-1} - \frac{z}{z-b} \quad \text{where} \quad \alpha = \frac{1}{1-b} \quad \text{and} \quad \beta = \frac{1}{b-1}.$$

Finally, the inverse z-transformation gives

$$y[kT_s] = 1[kT_s] - e^{-akT_s} = 1[kT_s] - \left(e^{-aT_s}\right)^k = 1[kT_s] - b^k.$$

Just to check this relation, assume $K = 1$ and $T_s = 0.5$:

$$b = 1 - K\,T_s = 0.5.$$

$$y[0] = 0; \quad y[T_s] = 1 - 0.5^1 = 0.5; \quad y[2T_s] = 1 - 0.5^2 = 0.75; \quad y[3T_s] = 1 - 0.5^3 = 0.875.$$

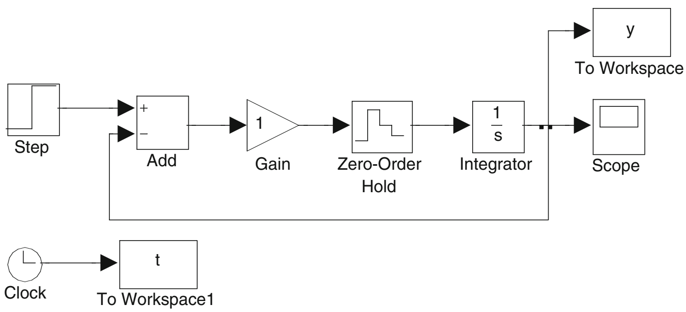


Fig. 11.6 SIMULINK™ diagram of the system

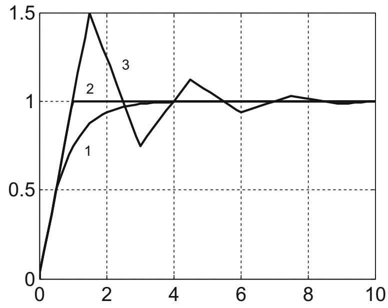


Fig. 11.7 The output signal for different sampling times

Alternatively, a SIMULINK™ program by Fig. 11.6 can be set up to illustrate the above calculations. Figure 11.7 shows the output for $T_s = 0.5$ (curve 1), $T_s = 1$ (curve 2) and $T_s = 1.5$ (curve 3).

11.3 State Space Equation of Sampled-Data Systems

The concept of the state-space model is valid for sampled-data systems, as well. The number of states describing the system dynamics is equal to the order of the difference equation representing the system. As learned earlier for continuous-time systems, several input-output equivalent state models can be derived for discrete-time systems, too.

11.3.1 Discretization of the Continuous-Time State Equation

Start with the solution of the continuous-time state-equation:

$$\boldsymbol{x}(t) = e^{\boldsymbol{A}(t-t_0)}\boldsymbol{x}(0) + \int_{t_o}^{t} e^{\boldsymbol{A}(t-\tau)}\boldsymbol{b}u(\tau)d\tau$$

Use a zero order holding unit. Enforce the integration between two consequtive sampling instants: $t_o = kT_s$ and $t = (k+1)T_s$. Within this region the input signal is constant: $u(\tau) = \text{constant} = u(kT_s)$. Assume that the $\boldsymbol{A}$ matrix is invertible. Then

$$\mathbf{x}[(k+1)T_s] = e^{AT_s}\boldsymbol{x}(kT_s) + \boldsymbol{A}^{-1}\left(e^{AT_s} - \boldsymbol{I}\right)\boldsymbol{b}u(kT_s).$$

Using the notation $u(kT_s) = u[k]$ and $\boldsymbol{x}[(k+1)T_s] = \boldsymbol{x}[k+1]$, the sampled version of the state equation takes the following form:

$$\begin{aligned}\boldsymbol{x}[k+1] &= \boldsymbol{F}\boldsymbol{x}[k] + \boldsymbol{g}u[k] \\ y[k] &= c^{\mathrm{T}}\boldsymbol{x}[k] + du[k]\end{aligned}$$

where

$$\boldsymbol{F} = e^{AT_s} \quad \text{and} \quad \boldsymbol{g} = \boldsymbol{A}^{-1}\left(e^{AT_s} - \boldsymbol{I}\right)\boldsymbol{b}.$$

Note that there is no change in the parameters $\mathbf{c}^{\mathrm{T}}$ and d as the system is transformed to discrete form.

Example 11.5 Derive the state equation for the system introduced in Example 11.2. Define the transfer function of the continuous-time process, and for sampling employ sampling time $T_s = 2.5$.

$$P(s) = \frac{1}{(1+5s)(1+10s)}$$

The related MATLAB™ commands are:

```
s=zpk('s')
Ps=1/((1+5*s)*(1+10*s))
Sysc=ss(Ps)
[A,b,c,d]=ssdata(Sysc)
Ts=2.5
Sysd=c2d(Sysc,Ts,'zoh')
[F,g,c1,d1]=ssdata(Sysd)
%check
F1=expm(A*Ts)
g1=inv(A)*(expm(A*Ts)-eye(2))*b
c2=c
d2=d
% display the continuous-time and
% the discrete-time step responses
step(Sysc,Sysd)
```

Summing up the results, the parameter matrices of the continuous-time system are:

```
A = -0.1000   1.0000
     0       -0.2000
b =  0
     0.1250
c =  0.1600   0
d =  0
```

while the parameter matrices of the discretized system are:

```
F =  0.7788   1.7227
     0        0.6065
g =  0.3058
     0.2459
c1 = 0.1600    0
d1 = 0
```

Application of the analytical relationships leads to the same result.

11.3.2 Derivation of the Discrete State Equation from the Pulse Transfer Function

Several state representations can be derived from the pulse transfer function.

Example 11.6 Consider the system introduced in Example 11.5. The related MATLAB™ program is:

```
% starting from a zero-pole-gain form
s=zpk('s')
Ps=1/((1+5*s)*(1+10*s))
Ts=2.5
z=zpk('z',Ts)
Pz=c2d(Ps,Ts)
Sysz=ss(Pz)
[F,g,c,d]=ssdata(Sysz)
% starting from a polinom/polinom form
s=tf('s')
Ps=1/((1+5*s)*(1+10*s))
Ts=2.5
Pz1=c2d(Ps,Ts)
Sysz1=ss(Pz1)
[F1,g1,c1,d1]=ssdata(Sysz1)
% canonical form
Sysz2=canon(Sysz1,'modal')
```

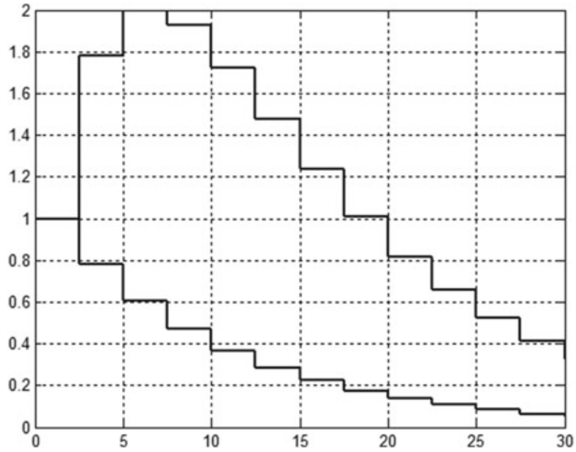
```
[F2,g2,c2,d2]=ssdata(Sysz2)
figure(1)
step(Sysz,Sysz1,Sysz2)
```

Summarizing the results:

```
      0.048929 (z+0.7788)
Pz= ---------------------
     (z-0.7788) (z-0.6065)

F =  0.6065   1.1770
     0         0.7788
g =  0
     0.2500
c =   0.2304   0.1957
d =   0

      0.04893 z + 0.03811
Pz1= ----------------------
      z^2 - 1.385 z + 0.4724

F1 = 1.3853  -0.4724
      1.0000     0
g1 = 0.2500
      0
c1 = 0.1957   0.1524
d1 = 0
Parameter matrices of the canonical form:
F2 = 0.7788   0
     0      0.6065
g2 = 2.6862
    -2.2815
c2 = 0.1647   0.1725
d2 = 0
```

The parameter matrices belonging to different interpretations are not identical, though the pulse transfer function behind these interpretations is unique. The second interpretation is the so called controllability form.

Fig. 11.8 The time course of the state variables in the sampled system



Investigate the system response to various initial conditions. Set $\boldsymbol{x}[0] = [1 \quad 1]^T$ and consider the first state-space representation.

The MATLAB™ program is

```
s=zpk('s')
Ps=1/((1+5*s)*(1+10*s))
Ts=2.5
z=zpk('z',Ts)
Pz=c2d(Ps,Ts)
Sysz=ss(Pz)
[F,g,c,d]=ssdata(Sysz)
x0=[1;1]
tfinal=30;
figure(2)
[y,t,x]=initial(Sysz,x0,tfinal)
stairs(t,x),grid
```

The state variables versus the time are shown in Fig. 11.8.