

In addition to *dbode* and *dnyquist*, MATLAB (CST) provides the command *dnichols* for drawing the digital Nichols plot. You may refer to the User's Guide for Control Systems Toolbox [1] for details about this as well as other digital frequency response commands.

8.6 Performance of Single-Input, Single-Output Digital Systems

Performance of a digital system can be studied by obtaining the time response to specified inputs. In a manner similar to the inverse Laplace transform for analog systems, we can use the definition of the z -transform to find the *inverse z -transform* of the output, $Y(z)$, denoted by $z^{-1}\{Y(z)\}$, to find the discrete time response of a single-input, single-output digital system with pulse transfer function, $G(z)$, and input, $U(z)$. From Eq. (8.6) it is clear that the inverse z -transform, $z^{-1}\{Y(z)\}$, yields the discrete time output, $y(kT)$. The method of finding the inverse z -transform of a function, $F(z)$, is quite similar to that followed in finding the inverse Laplace transform, and involves expressing $F(z)$ as a *partial fraction* expansion in z (there is an alternative method of using *power series expansion* in z , but we will not consider it due to its complexity). However, examining the z -transforms given in Table 8.1, it is clear that z is usually present as a numerator factor in most z -transforms. Hence, to find the inverse z -transform of $F(z)$, it will be useful to express $F(z)$ as the following expansion:

$$F(z) = K_1 z/(z - z_1) + K_2 z/(z - z_2) + \cdots + K_n z/(z - z_n) \quad (8.47)$$

where $z_1, z_2, \dots, z_n$ are the poles of $F(z)$ and $K_1, K_2, \dots, K_n$ are the residues of the following partial fraction expansion:

$$F(z) = K_1/(z - z_1) + K_2/(z - z_2) + \cdots + K_n/(z - z_n) \quad (8.48)$$

(Note that, for simplicity, we have assumed that all the poles of $F(z)$ are distinct. If some poles are repeated, we can use the more general partial fraction expansion introduced in Chapter 2.) Taking the inverse z -transform of each term on the right-hand side of Eq. (8.47), we can write

$$f(kT) = K_1 z_1^k + K_2 z_2^k + \cdots + K_n z_n^k \quad (8.49)$$

We can apply the above strategy to find the discrete time output, $y(kT)$, of a digital system, $G(z)$, with input $U(z)$ and zero initial conditions, by finding the inverse z -transform of $Y(z) = G(z)U(z)$.

Example 8.12

Let us find the discrete time response of $G(z) = z/(z^2 - 0.7z + 0.3)$ to a unit step input, $u(kT) = u_s(kT)$. The z -transform of $u_s(kT)$ is $z/(z - 1)$. Hence, $Y(z) = G(z)U(z) = z^2/[(z - 1)(z^2 - 0.7z + 0.3)]$. We begin by finding the partial fraction expansion of $Y(z)/z$ as follows by using the MATLAB command *residue*:

```
>>num = [1 0];den = conv([1 -1],[1 -0.7 0.3]); [K,z,c] = residue(num,
den)<enter>
```

```
K =
    1.6667
   -0.8333+0.0989i
   -0.8333-0.0989i
z =
    1.0000
    0.3500+0.4213i
    0.3500-0.4213i
c =
    []
```

Hence, the partial fraction expansion of $Y(z)/z$ is the following:

$$\begin{aligned} Y(z)/z &= z/[(z-1)(z^2-0.7z+0.3)] \\ &= (5/3)/(z-1) + (-0.8333+0.0989i)/[z-(0.3500+0.4213i)] \\ &\quad + (-0.8333-0.0989i)/[z-(0.3500-0.4213i)] \end{aligned} \quad (8.50)$$

or

$$\begin{aligned} Y(z) &= (5/3)z/(z-1) + (-0.8333+0.0989i)z/[z-(0.3500+0.4213i)] \\ &\quad + (-0.8333-0.0989i)z/[z-(0.3500-0.4213i)] \end{aligned} \quad (8.51)$$

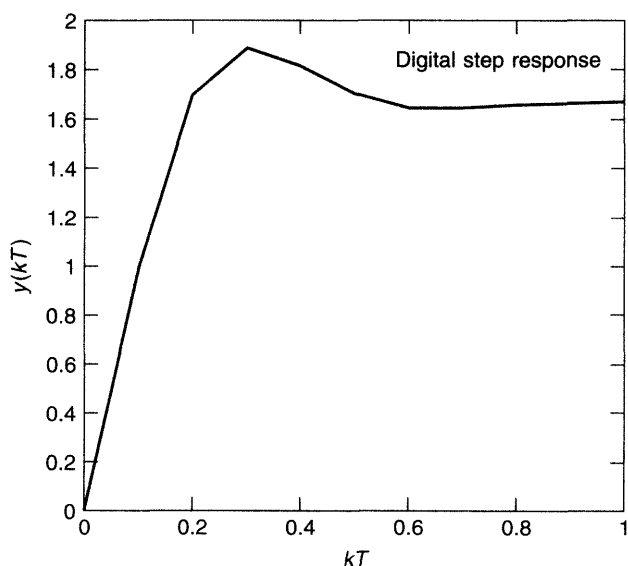


Figure 8.7 Step response of the digital system of Example 8.11 with sampling interval, $T = 0.1$ s

Then, taking the inverse z -transform of each term on the right-hand side of Eq. (8.51), we get the discrete time output at the k th sampling instant as follows:

$$f(kT) = 5/3 + (-0.8333 + 0.0989i)(0.3500 + 0.4213i)^k \\ + (-0.8333 - 0.0989i)(0.3500 - 0.4213i)^k \quad (8.52)$$

The step response, $f(kT)$, calculated by Eq. (8.52) is plotted in Figure 8.7 for $T = 0.1$ seconds using the following MATLAB command:

```
>>for k=1:11;f(k)=K(1)+K(2)*z(2)^(k-1)+K(3)*z(3)^(k-1);t(k)=0.1*(k-1);end;plot(t,f) <enter>
```

Note that the step response has a maximum overshoot of 13% with a steady state value of about 1.67. The peak time and the settling time are defined for the digital response as occurring after so many sampling intervals. The peak time occurs for $k = 3$, while the settling time occurs for $k = 5$. With $T = 0.1$ second, the peak time is 0.3 second, while the settling time is 0.5 second.

MATLAB M-files can be written to compute step and impulse responses of digital systems using the inverse z -transform of partial fraction expansion (such as *impresp.m* and *stepresp.m* for impulse and step responses, respectively, of analog systems in Chapter 2).

The steady state value of the response, $Y(z)$, can be directly obtained using the final-value theorem given by Eq. (8.16), as follows:

$$y(\infty) = \lim_{z \rightarrow 1} (1 - z^{-1})Y(z) = \lim_{z \rightarrow 1} (1 - z^{-1})G(z)U(z) \quad (8.53)$$

(Of course, the existence of the steady-state limit in Eq. (8.53) translates into the requirement that all the poles of $G(z)U(z)$ should lie inside the unit circle.) The steady state value of the response can thus be found using Eq. (8.53), without having to compute and plot the response. Let us use Eq. (8.53) to find the steady state value of the *step response* of a digital system with pulse transfer function, $G(z)$, as follows:

$$y(\infty) = \lim_{z \rightarrow 1} (1 - z^{-1})Y(z) = \lim_{z \rightarrow 1} (1 - z^{-1})G(z)U(z) \\ = \lim_{z \rightarrow 1} (1 - z^{-1})[z/(z - 1)]G(z) = \lim_{z \rightarrow 1} G(z) \quad (8.54)$$

Thus, the steady state value of the step response of $G(z)$ is given by $\lim_{z \rightarrow 1} G(z)$ called the *digital DC gain* of $G(z)$. However, instead of having to calculate the limit in Eq. (8.54) by hand, we can use the MATLAB (CST) command *ddcgain* to find the digital DC gain as follows:

```
>>ddcgain(num,den) <enter>
```

where *num* and *den* are the numerator and denominator polynomials of the pulse transfer function, $G(z)$, in decreasing powers of z . Let us find the steady state value of the step response in Example 8.12 using *ddcgain* as follows:

```
>>num = [1 0];den = [1 -0.7 0.3];ddcgain(num,den) <enter>
```

```
ans =  
1.6667
```