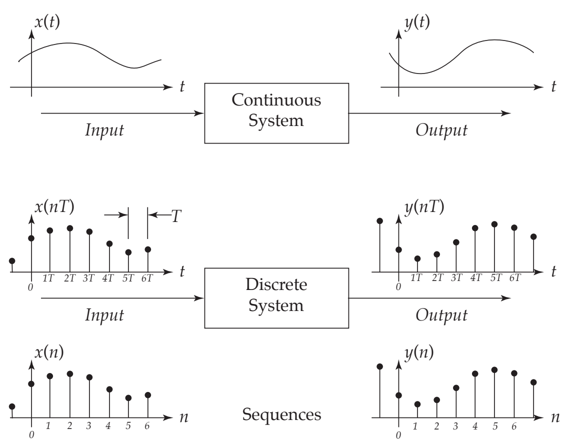
ME 477 Embedded Computing
Notes on Discrete-Time Dynamic Systems

J. L. Garbini

Suppose that we wish to implement an embedded computer system that behaves analogously to a continuous linear single-input-single-output dynamic system. The input and output for the continuous system are continuous functions of time. The corresponding input and output for the embedded system are data, sampled with period T, that form two discrete-time sequences as shown.



The continuous system can be described by a linear, constant-coefficient differential equation:

$$\frac{d^n y}{dt^n} + c_{n-1}\frac{d^{n-1}y}{dt^{n-1}} + \ldots + c_1\frac{dy}{dt} + c_0 y = \\ = d_m\frac{d^m x}{dt^m} + d_{m-1}\frac{d^{m-1}x}{dt^{m-1}} + \ldots + d_1\frac{dx}{dt} + d_0 x$$

where c_k and d_k are constants. The equivalent transfer function is

$$T(s) = \frac{Y(s)}{X(s)} = \frac{d_m s^m + d_{m-1}s^{m-1} + \ldots + d_1 s^1 + d_0}{s^n + c_{n-1}s^{n-1} + \ldots + c_1 s^1 + c_0}$$

The corresponding discrete system is described by a *difference* equation that operates on the sequence of input values $x(n)$ to produce the output sequence $y(n)$.

The difference equation has the form

$$a_0 y(n) + a_1 y(n-1) + \ldots + a_N y(n-N) = \\ = b_0 x(n) + b_1 x(n-1) + \ldots + b_M x(n-M) \quad (1)$$

for $n = 0, 1, 2, \ldots$, and where $x(n)$ is a sequence of periodically digitized values of the analog input signal, $y(n)$ is a sequence of values that determine the output signal, and $a_k, k = 0, 1, \ldots N$ and $b_k, k = 0, 1, \ldots M$ are constants.

This equation can also be written in summation form:

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k) \quad (2)$$

or, solving (2) for the current output sample $y(n)$,

$$y(n) = \frac{1}{a_0}\left[\sum_{k=0}^{M} b_k x(n-k) - \sum_{k=1}^{N} a_k y(n-k)\right] \quad (3)$$

Notice that the most recent output value $y(n)$ depends on previous values of y and on the previous and current values of the input x.

The values of the constants in the difference equation can be determined from the constants in the differential equation and from the sample period T. To see the relationship between the differential and difference equations consider the following.

The z-transform – In the analysis of continuous systems, we use the Laplace transform, defined by

$$L\{f(t)\} = F(s) = \int_0^\infty f(t)e^{-st}dt$$

which leads directly to the familiar property that the Laplace transform of the derivative of a function $f(t)$ (with zero initial conditions) is s times the transform of the function $F(s)$:

$$L\left\{\frac{df(t)}{dt}\right\} = sF(s), \quad (4)$$

which enables us to find easily the transfer function of a linear *continuous* system, given its *differential* equation.

For discrete systems a very similar procedure is available. The z-transform of a sequence is defined by

$$Z\{f(n)\} = F(z) = \sum_{n=0}^{\infty} f(n)z^{-n}$$

where $f(n)$ is the sampled version of $f(t)$, as shown above. This leads directly to a property analogous to (4) for discrete systems: The z-transform of a function delayed by one sample period is z^{-1} times the transform of the function $F(z)$:

$$Z\{f(n-1)\} = z^{-1}F(z), \quad (5)$$

We can easily find the transfer function of a *discrete* system given its *difference* equation. For example, the z-transform of the second order difference equation

$$y(n) + a_1 y(n-1) + a_2 y(n-2) = \\ = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) \quad (6)$$

is determined by successively applying (5) to arrive at

$$\left(1 + a_1 z^{-1} + a_2 z^{-2}\right) Y(z) = \left(b_0 + b_1 z^{-1} + b_2 z^{-2}\right) X(z) \quad (7)$$

Rearranging, the discrete transfer function is

$$T(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \quad (8)$$

Notice that the transfer function (8) and the difference equation (6), can be derived from each other by inspection. Notice also that the transfer function of a discrete system is the ratio of two polynomials in z, just as the transfer function of a continuous system is the ratio of two polynomials in s.

There are several ways to derive an approximate *discrete* model from a corresponding *continuous* model. We will use a popular technique called Tustin's method that approximates a continuous function of time by straight lines connecting the sampled points (trapezoidal integration.)

The *discrete* transfer function is found using Tustin's method by making the following substitution in the *continuous* transfer function

$$s = \frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right) \tag{9}$$

and rewriting the transfer function in the form of equation (8). Here, T is the sample period.

Example – Consider a continuous first order system described by the transfer function:

$$T(s) = \frac{Y(s)}{X(s)} = \frac{1}{\tau s + 1}, \text{ where } \tau \text{ is the time constant.}$$

We want to find the corresponding discrete-time transfer function and difference equation. Substituting equation (9) into $T(s)$ we have:

$$T(z) = \frac{Y(z)}{X(z)} = \frac{\alpha + \alpha z^{-1}}{1-(1-2\alpha)z^{-1}}, \tag{10}$$

where α is a constant:

$$\alpha = \frac{T}{2\tau + T}$$

from which the difference equation can be inferred (see equations (6), (7), and (8) above):

$$y(n) = (1-2\alpha)y(n-1) + \alpha x(n) + \alpha x(n-1) \tag{11}$$

Notice again that the current value of the output $y(n)$ depends on the previous output, $y(n-1)$, and on the *current* and previous inputs, $x(n)$ and $x(n-1)$.

Notice also that the coefficients depend on the time constant τ in the original continuous system and on the sample period T.

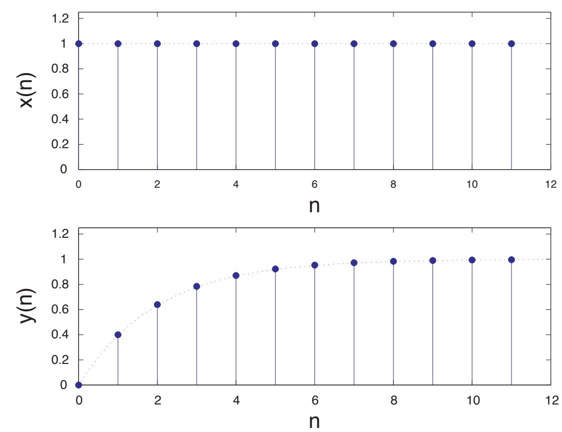
During each sample period, the value of the current value of the input $x(n)$ is measured and the current value of the output $y(n)$ is computed. Suppose that the time constant $\tau = 2$, the sample period $T = 1$, and that the input is a unit step ($x(n) = 1$ for all n), and the initial condition $y(0) = 0$. Then from equation (11):

$$y(n) = 0.6y(n-1) + 0.4$$

and we can compute the output sequence:

$$\begin{aligned} y(0) &= 0 \\ y(1) &= 0.6 \times 0 + 0.4 = 0.4 \\ y(2) &= 0.6 \times 0.4 + 0.4 = 0.64 \\ y(3) &= 0.6 \times 0.64 + 0.4 = 0.784 \\ y(4) &= 0.6 \times 0.784 + 0.4 = 0.870 \quad \text{etc.} \end{aligned}$$

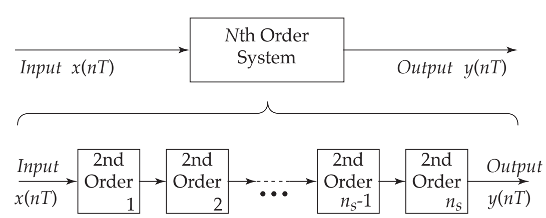
Here are plots of the input and output sequences.



The dotted line is the exact solution $y(t/T)$ of the original continuous *differential* equation. As you can see, Tustin's method is very close to the exact solution at the sample points.

The Biquad Cascade – Although we could implement equation (3) as shown, the sensitivity of the output to the coefficients leads to numerical inaccuracies as the order of the system N becomes large. We will solve this problem by breaking the Nth order system it into a series of n_s 2nd order systems.

The technique is called a *Biquad Cascade*:



Notice that the output of each 2nd order section (biquad)[1] is the input to the subsequent section. Each biquad implements the same 2nd order difference equation, but with different coefficients, inputs, and outputs.

[1] "biquad" is short for "biquadratic." The biquad transfer function has 2nd order polynomials in both numerator and denominator.

For example, the current output $y_i(n)$ from the ith section would be:

$$y_i(n) = [\ b_{0_i}x_i(n) + +b_{1_i}x_i(n-1) + b_{2_i}x_i(n-2) + -a_{1_i}y_i(n-1) - a_{2_i}y_i(n-2)\ ]/a_{0_i} \quad (12)$$

Of course, a first or second order transfer function would require only one biquad. Depending on the value of N, some of the coefficients of at least one biquad may be zero. We will implement a function to handle any value of N.

There are a variety of algorithms for breaking a transfer function into the 2nd order sections. MATLAB contains a built-in function "`tf2sos`" (transfer function to second order sections) for this purpose.

Discrete-Time Controllers - For reference, here are the Tustin equivalents for some common continuous-time controllers:

Phase Lead/Lag continuous transfer function	discrete transfer function
$T(s) = \frac{Y(s)}{X(s)} = k\frac{s+z}{s+p}$	$T(z) = \frac{Y(z)}{X(z)} = k\frac{b_0+b_1z^{-1}}{a_0+a_1z^{-1}}$
differential equation	difference equation
$\frac{dy}{dt} + py = k\left(\frac{dx}{dt} + zx\right)$	$y(n) = -\frac{a_1}{a_0}y(n-1) + +\frac{b_0}{a_0}x(n) + \frac{b_1}{a_0}x(n-1)$ $a_0 = 1, b_0 = k\frac{zT+2}{pT+2}$ $a_1 = \frac{pT-2}{pT+2}, b_1 = k\frac{zT-2}{pT+2}$

PI continuous transfer function	discrete transfer function
$T(s) = \frac{Y(s)}{X(s)} = K_p + \frac{K_i}{s}$	$T(z) = \frac{Y(z)}{X(z)} = \frac{b_0+b_1z^{-1}}{a_0+a_1z^{-1}}$
differential equation	difference equation
$y(t) = K_px(t) + +K_i\int_0^t x(t)dt$	$y(n) = -\frac{a_1}{a_0}y(n-1) + +\frac{b_0}{a_0}x(n) + \frac{b_1}{a_0}x(n-1)$ $a_0 = 1, b_0 = K_p + \frac{1}{2}K_iT$ $a_1 = -1, b_1 = -K_p + \frac{1}{2}K_iT$

PID continuous transfer function	discrete transfer function
$T(s) = \frac{Y(s)}{X(s)} = K_p + \frac{K_i}{s} + K_ds$	$T(z) = \frac{Y(z)}{X(z)} = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{a_0+a_1z^{-1}+a_2z^{-2}}$
differential equation	difference equation
$y(t) = K_px(t) + +K_i\int_0^t x(t)dt + +K_d\frac{dx}{dt}$	$y(n) = -\frac{a_1}{a_0}y(n-1) - \frac{a_2}{a_0}y(n-2) + +\frac{b_0}{a_0}x(n) + \frac{b_1}{a_0}x(n-1) + +\frac{b_2}{a_0}x(n-2)$ $a_0 = 1, b_0 = \frac{2K_pT+K_iT^2+4K_d}{2T}$ $a_1 = 0, b_1 = \frac{2K_iT^2-8K_d}{2T}$ $a_2 = -1, b_2 = \frac{-2K_pT+K_iT^2+4K_d}{2T}$

P.S. There are many more uses for z-transforms. For reference, see *Digital Control of Dynamic Systems*, by Franklin, Powell, and Workman. p. 189. 1997

By the way, the MATLAB Control Toolbox contains a function "c2d" that computes the Tustin equivalent discrete system, `SYSD`, from the continuous system, `SYS`:

```
SYSD = c2d(SYS, T, 'tustin')
```