EE16B - Fall'16 - Lecture 11B Notes[1]

Murat Arcak

10 November 2016

[1]

Sampling Continued

Suppose we sample the function $f : \mathbb{R} \to \mathbb{R}$ at evenly spaced points

$$x_i = \Delta i, \quad i = 1, 2, 3, \ldots$$

Then, sinc interpolation between the samples $y_i = f(x_i)$ gives:

$$\hat{f}(x) = \sum_i y_i \phi(x - \Delta i) \tag{1}$$

where

$$\phi(x) = \text{sinc}(x/\Delta),$$

which is band-limited by π/Δ.

Sampling Theorem: If $f(x)$ is band-limited by frequency

$$\omega_{\max} < \frac{\pi}{\Delta} \tag{2}$$

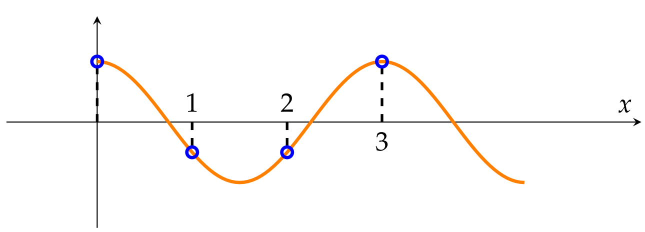
then the sinc interpolation (1) recovers $f(x)$, that is $\hat{f}(x) = f(x)$.

This means that the sampling frequency, $\omega_s = 2\pi/\Delta$, must exceed $2\omega_{\max}$ which is known as the Nyquist frequency.

Example 1: Suppose we sample the function

$$f(x) = \cos\left(\frac{2\pi}{3}x\right)$$

with period $\Delta = 1$. This means that we take 3 samples in each period of the cosine function, as shown in the figure below. Since $\omega_{\max} = \frac{2\pi}{3}$ and $\Delta = 1$, the criterion (2) holds and we conclude that the sinc interpolation (1) exactly recovers $f(x)$.



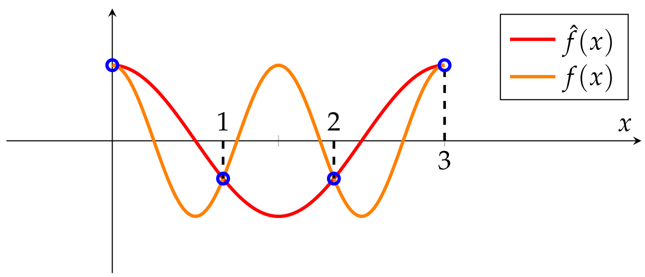
Example 2: Suppose now the function being sampled is

$$f(x) = \cos\left(\frac{4\pi}{3}x\right). \tag{3}$$

With $\omega_{\max} = \frac{4\pi}{3}$ and $\Delta = 1$, the criterion (2) fails. To see that the result of the sinc interpolation $\hat{f}(x)$ is now different from $f(x)$, note that this time we take 3 samples every *two* periods of the cosine function, as shown below. These samples are identical to the 3 samples collected in one period of the function in Example 1 above. Therefore, sinc interpolation gives the same result it did in Example 1:

$$\hat{f}(x) = \cos\left(\frac{2\pi}{3}x\right)$$

which does not match (3).



Aliasing and Phase Reversal

In Example 2 the low frequency component $2\pi/3$ appeared in $\hat{f}(x)$ from the actual frequency $4\pi/3$ of $f(x)$ that exceeded the critical value $\pi/\Delta = \pi$. The emergence of phantom lower frequency components as a result of under-sampling is known as "aliasing."

To generalize Example 2 suppose we sample the function

$$f(x) = \cos(\omega x + \phi) \tag{4}$$

with period $\Delta = 1$ and obtain

$$y_i = \cos(\omega i + \phi).$$

Using the identity $\cos(2\pi i - \theta) = \cos(\theta)$ which holds for any integer i, and substituting $\theta = \omega i + \phi$. we get

$$y_i = \cos(2\pi i - \omega i - \phi) = \cos((2\pi - \omega)i - \phi)$$

which suggests that the samples of the function

$$\cos((2\pi - \omega)x - \phi) \tag{5}$$

are identical to those of (4).

If $\omega \in (\pi, 2\pi]$ in (4) then sinc interpolation gives the function in (5) whose frequency is $2\pi - \omega \in [0, \pi)$. This function changes more slowly than (4) and the sign of the phase ϕ is reversed. These effects are visible in movies where a rotating wheel appears to rotate more slowly and in the opposite direction when its speed exceeds half of the sampling rate (18-24 frames/second).

Example 3: Suppose we sample the function

$$f(x) = \sin(1.9\pi x)$$

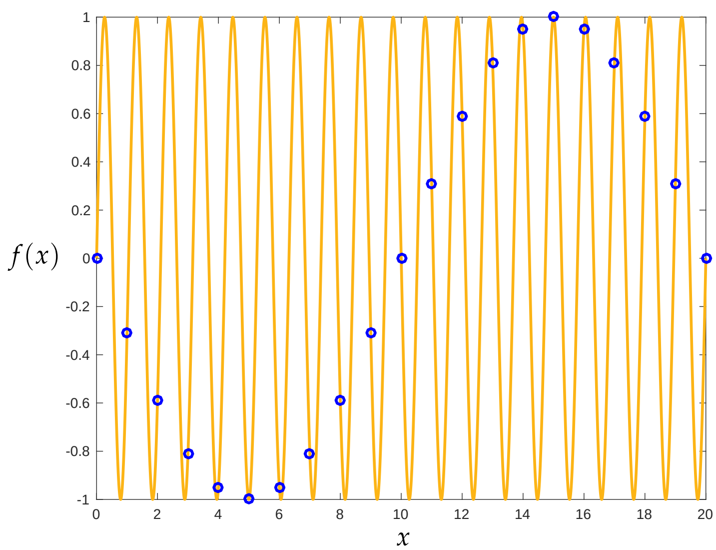
with $\Delta = 1$ as shown in the figure below. This function is of the form (4) with $\omega = 1.9\pi$ and $\phi = -\pi/2$ because

$$\sin(1.9\pi x) = \cos(1.9\pi x - \pi/2).$$

Thus, from (5), the sinc interpolation gives

$$\hat{f}(x) = \cos(0.1\pi x + \pi/2) = -\sin(0.1\pi x)$$

as evident from the samples in the figure. Note that the negative sign of $-\sin(0.1\pi x)$ is a result of the phase reversal discussed above.



Discrete-Time Control of Continuous-Time Systems

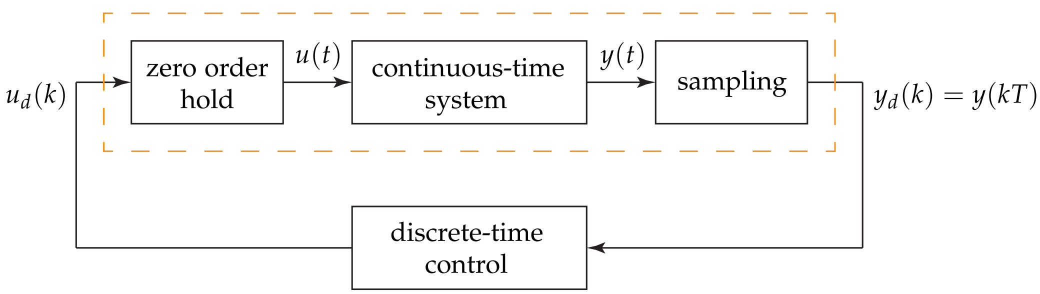
In a typical application the control algorithm for a continuous-time physical system is executed in discrete-time. This means that the measured variables of the system must be sampled before being processed by the control algorithm. Conversely, the discrete-time control input generated by the algorithm must be interpolated into a continuous-time function, typically with a zero order hold, before being applied back to the continuous-time system.

This scheme is depicted in the block diagram below. We let T denote the sampling period and represent the samples of the output $y(t)$ at $t = kT$, $k = 0, 1, 2, \ldots$ by

$$y_d(k) = y(kT)$$

where the subscript "d" stands for discrete-time. The control sequence generated in discrete time is denoted $u_d(k)$ and is interpolated by the zero order hold block to the continuous-time input

$$u(t) = u_d(k) \quad t \in [kT, (k+1)T). \tag{6}$$



To design a discrete-time control algorithm we need a discrete-time model for the continuous-time system, combined with the zero order hold and sampling blocks. This combination is depicted with the dashed box in the figure above, with input $u_d(k)$ and output $y_d(k)$.

Suppose the continuous-time system model is

$$\begin{aligned} \frac{d}{dt}\vec{x}(t) &= A\vec{x}(t) + Bu(t) \\ y(t) &= C\vec{x}(t) \end{aligned} \tag{7}$$

and we wish to obtain a discrete-time model

$$\begin{aligned} \vec{x}_d(k+1) &= A_d\vec{x}(k) + B_d u_d(k) \\ y_d(k) &= C_d\vec{x}_d(k) \end{aligned} \tag{8}$$

where $\vec{x}_d(k)$ is the value of the state $\vec{x}(t)$ at time $t = kT$. It follows that $C_d = C$ because

$$y_d(k) = y(kT) = C\vec{x}(kT) = C\vec{x}_d(k).$$

To find A_d and B_d in (8) we need the solution of (7) at $t = (k+1)T$ with initial condition $\vec{x}(kT) = \vec{x}_d(k)$ and constant input (6).

When A and B are scalars the solution of the differential equation (7) with initial condition $x(0)$ and constant input $u(t) = u$ is[2]

$$x(t) = e^{At}x(0) + \frac{e^{At} - 1}{A}Bu.$$

[2] Show that this solution indeed satisfies (7) and meets the initial condition $x(0)$.

Since A and B don't change with time[3] we can shift the time by kT and write

$$x(t+kT) = e^{At}x(kT) + \frac{e^{At}-1}{A}Bu.$$

Then, substituting $t = T$ and the constant input $u = u_d(k)$, we get

$$\underbrace{x((k+1)T)}_{x_d(k+1)} = e^{AT}\underbrace{x(kT)}_{x_d(k)} + \frac{e^{AT}-1}{A}Bu_d(k).$$

Thus, when A and B are scalars, we have

$$A_d = e^{AT} \quad \text{and} \quad B_d = \frac{e^{AT}-1}{A}B.$$

In a homework problem you will extend this derivation from scalars to matrices, assuming that A is diagonalizable.

[3] *i.e.*, the system (7) is "time-invariant"