MAE/ECE 5320 Mechatronics

2025 Spring
Lecture 03-04

Tianyi He
Utah State University

Content

- Background
- Continuous-time signal: time and frequency domains
- Signal discretization: sample and hold
- Aliasing due to improper sample rate
- Discrete-time signal: time and frequency (FFT) domains
- Continuous-time filtering
- Discrete-time filtering

Background

Continuous-time domain

Signal processing

Fourier and Laplace transformations

Continuous-time filters

→

Digitalization

Data sampling

Quantization

→

Discrete-time domain

Signal processing

Fast Fourier transformation (FFT)

Discrete-time filters

Content

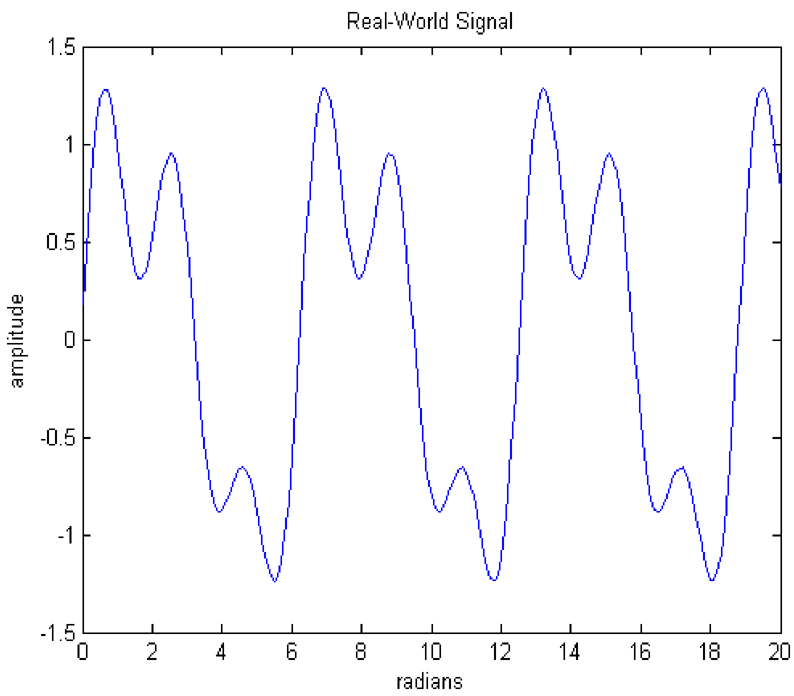
- ❑ Background
- ❑ Continuous-time signal: time and frequency domains
- ❑ Signal discretization: sample and hold
- ❑ Aliasing due to improper sample rate
- ❑ Discrete-time signal: time and frequency (FFT) domains
- ❑ Continuous-time filtering
- ❑ Discrete-time filtering

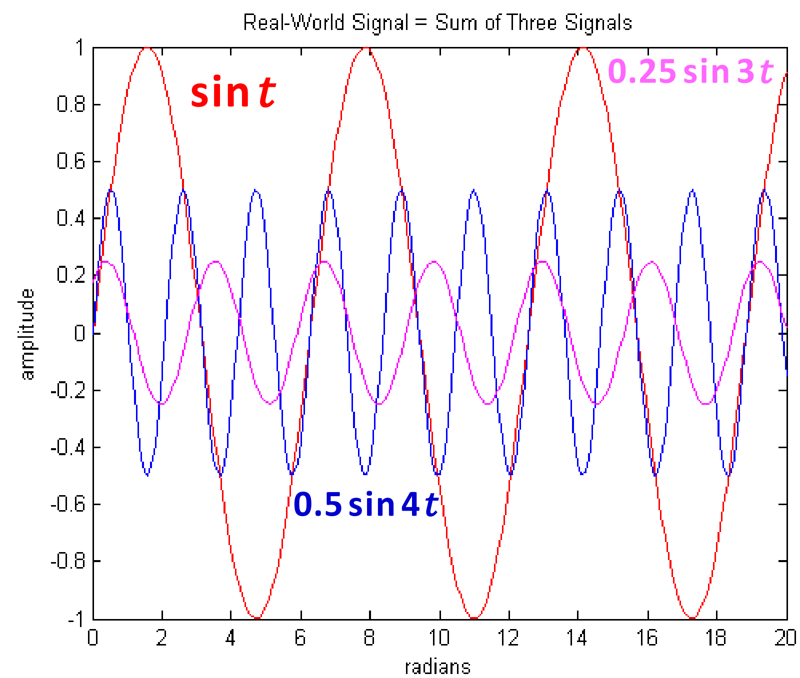
Continuous time domain signal

Consider the sum of Sinusoid signals with different frequencies & amplitudes below.

$$f(t) = \sin(t) + 0.5\sin(4t) + 0.25\sin(3t)$$

The individual signals are plotted below, and one can see that there are three main frequency components $(1,3,4)$ rad/s $=\left(\frac{1}{2\pi}, \frac{3}{2\pi}, \frac{2}{\pi}\right) Hz$





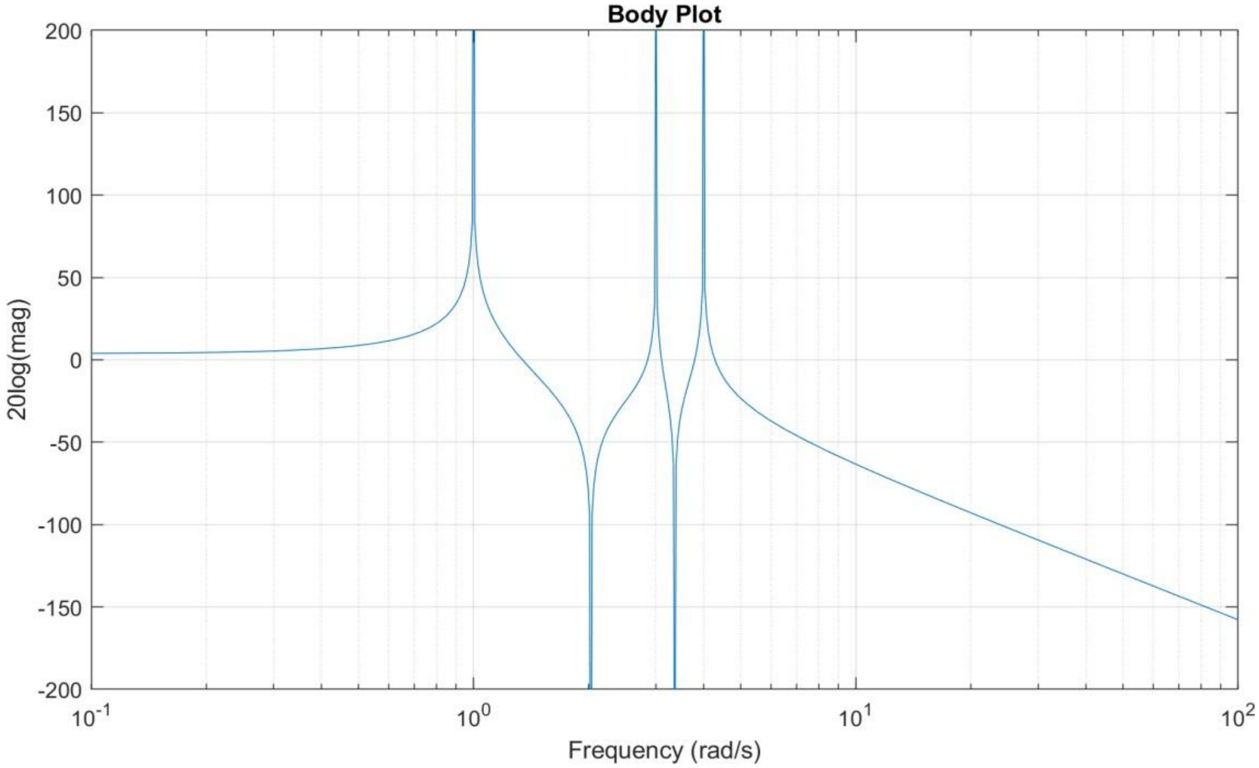
Continuous frequency responses

The magnitude and phase of Laplace transform of a continuous-time signal is called its frequency response. For the following signal

$$f(t) = \sin(t) + 0.5\sin(4t) + 0.25\sin(3t)$$

Its Laplace transform is

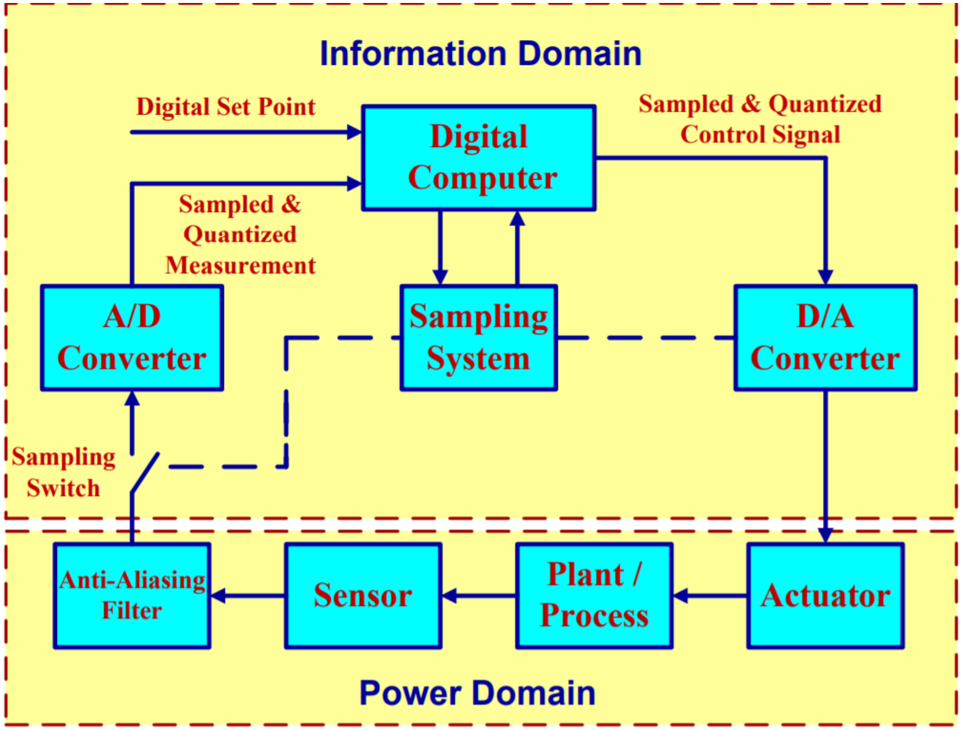
$$F(s) = \frac{1}{s^2+1} + \frac{2}{s^2+4^2} + \frac{0.75}{s^2+3^2} = \frac{3.75s^4 + 36.75s^2 + 57}{s^6 + 14s^4 + 49s^2 + 36}$$



Content

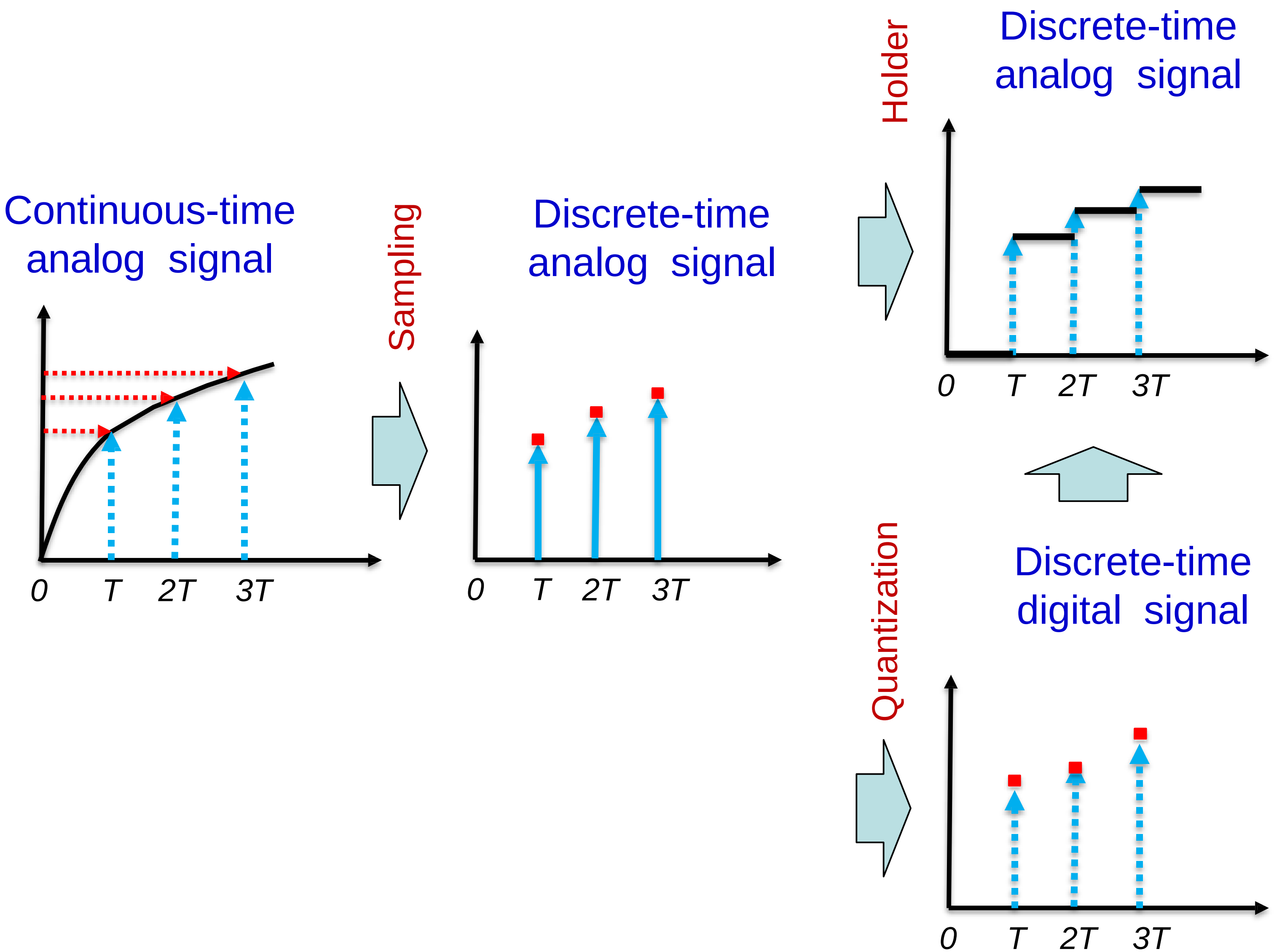
- ❑ Background
- ❑ Continuous-time signal: time and frequency domains
- ❑ Signal discretization: sample and hold
- ❑ Aliasing due to improper sample rate
- ❑ Discrete-time signal: time and frequency (FFT) domains
- ❑ Continuous-time filtering
- ❑ Discrete-time filtering

Sample, hold and quantization (1)

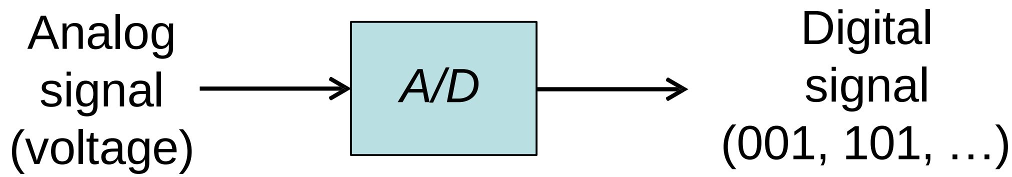


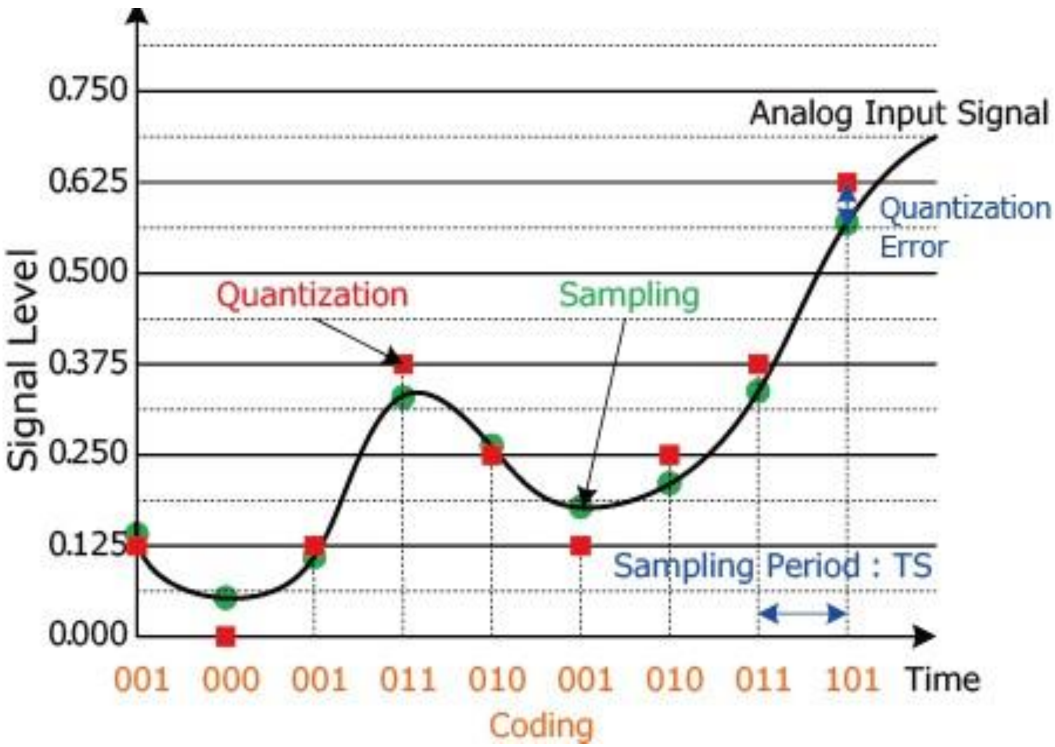
- ❑ Modern controller is implemented into a digital device such as a micro-controller equipped with a microprocessor, A/D and D/A devices, etc.
- ❑ Microcontroller connects discrete-time world to the continuous-time one through A/D and D/A converters
- ❑ A/D and D/A converters connect to physical sensors and actuators, respectively.

Sample, hold and quantization (2)



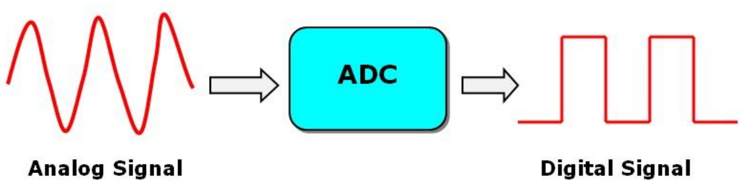
Quantization A/D and D/A (1)



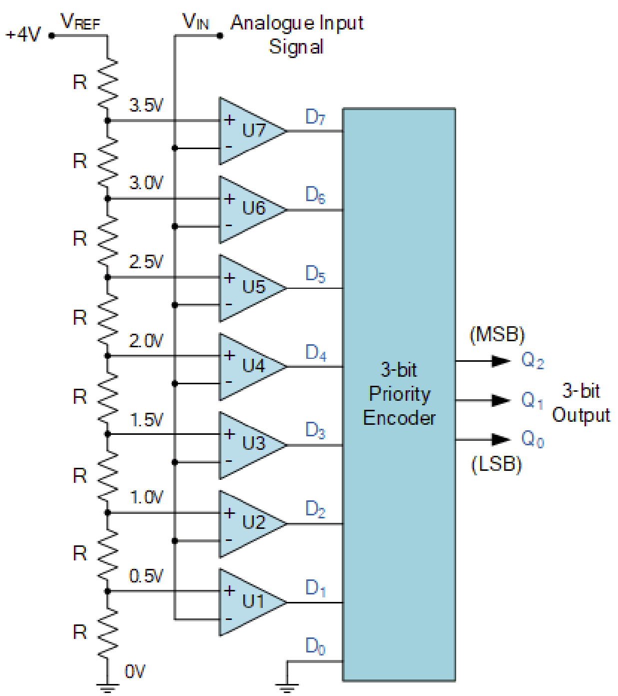


Digital #	Analog voltage
000	0.0 ~ 0.0625
001	0.0625 ~ 0.1875
010	0.1875 ~ 0.3125
011	0.3125 ~ 0.4375
100	0.4375 ~ 0.5625
101	0.5625 ~ 0.6875
110	0.6875 ~ 0.8125
111	0.8125 ~ 0.9375

Quantization A/D and D/A (2)



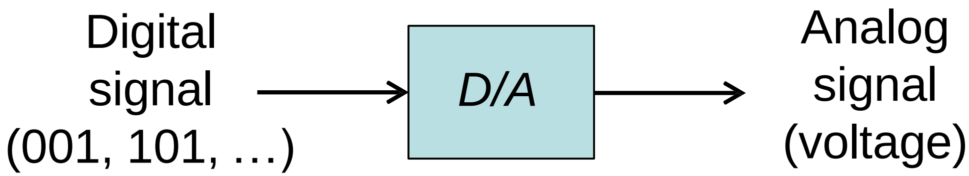
A sample 3-bit A/D circuit:
Ladder circuit

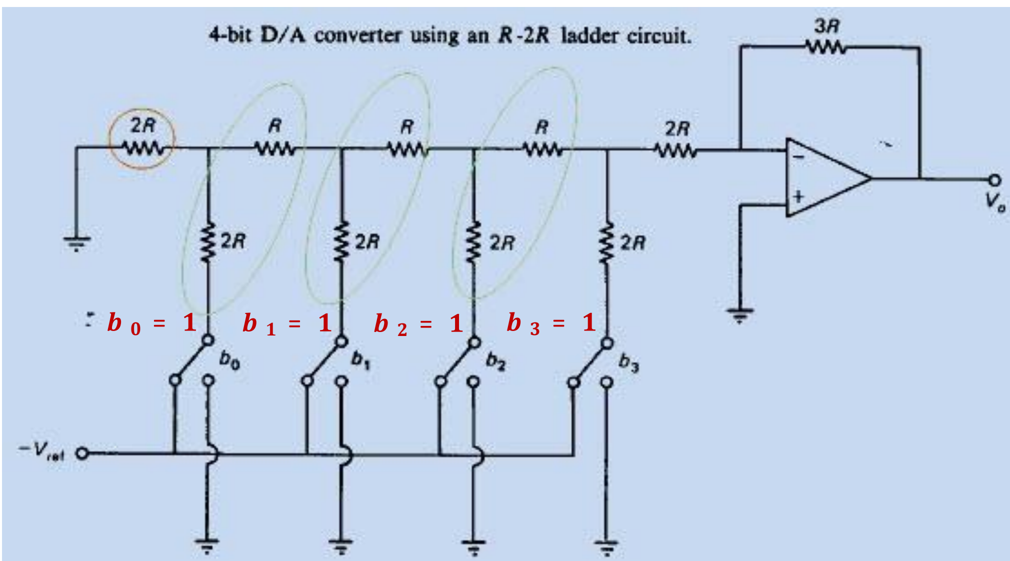


3-bit A/D converter Output

Analogue Input Voltage (V_{IN})	Comparator Outputs								Digital Outputs		
	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Q_2	Q_1	Q_0
0 to 0.5 V	0	0	0	0	0	0	0	0	0	0	0
0.5 to 1.0 V	0	0	0	0	0	0	1	X	0	0	1
1.0 to 1.5 V	0	0	0	0	0	1	X	X	0	1	0
1.5 to 2.0 V	0	0	0	0	1	X	X	X	0	1	1
2.0 to 2.5 V	0	0	0	1	X	X	X	X	1	0	0
2.5 to 3.0 V	0	0	1	X	X	X	X	X	1	0	1
3.0 to 3.5 V	0	1	X	X	X	X	X	X	1	1	0
3.5 to 4.0 V	1	X	X	X	X	X	X	X	1	1	1

Quantization A/D and D/A (3)





A sample D/A circuit

$b_3 b_2 b_1 b_0$	$V_o = x * V_{ref}$
0001	$x = 0.0625$
0010	$x = 0.1250$
0010	$x = 0.2500$
1000	$x = 0.5000$

Quantization A/D and D/A (4)

- The electric circuits that perform the A/D and D/A conversions are called analog-to-digital (A/D) and digital-to-analog (D/A) converters, respectively. Most of A/D and D/A converters are fabricated as integrated circuits and can be found any place where analog signals are to be stored, processed, or transmitted digitally, e.g., cell phones, computers, and automobiles.

- The quantization operation causes errors in representing digitized information. These errors are called quantization noise, because the effect of quantization errors sounds like noise in a digitized music signal and looks like noise in a digital image. In general, the high the A/D bits, the small the quantization error.

Quantization A/D and D/A (5)

- Computers and digital devices represent, store information in a binary number code. The smallest unit is called a bit (binary digit) which can represent only one of two states: 0 or 1. Many bits together represent more states.

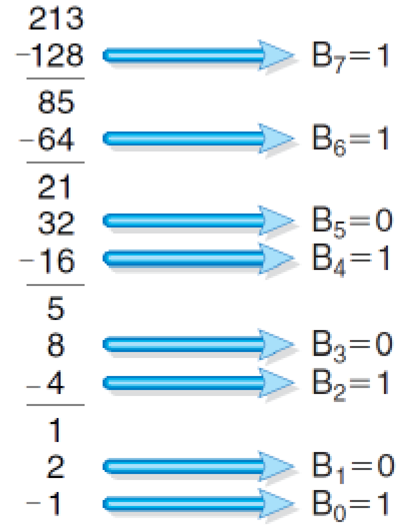
- Why bits? Why binary number representation and arithmetic?
  - A transistor is the basic building block of nearly all digital technologies and it is designed to act like a switch having two distinct states (0 and 1)
  - In addition, semiconductor memory (RAM), magnetic disks, and optical disks store only one of two possible states (0 or 1) at each physical location on the device

- N bits can represent 2^N states, and the largest integer is $2^N - 1$

Quantization A/D and D/A (6)

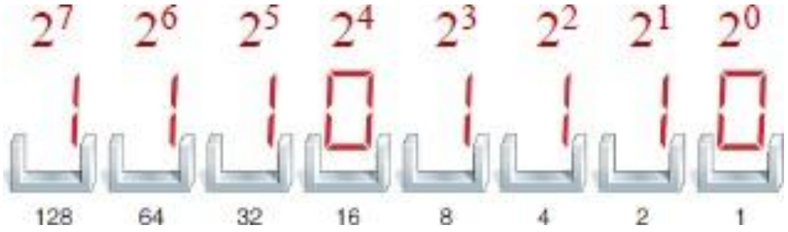
Decimal to binary

$$213 = 1 * 2^7 + 1 * 2^6 + 0 * 2^5 + 1 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0$$



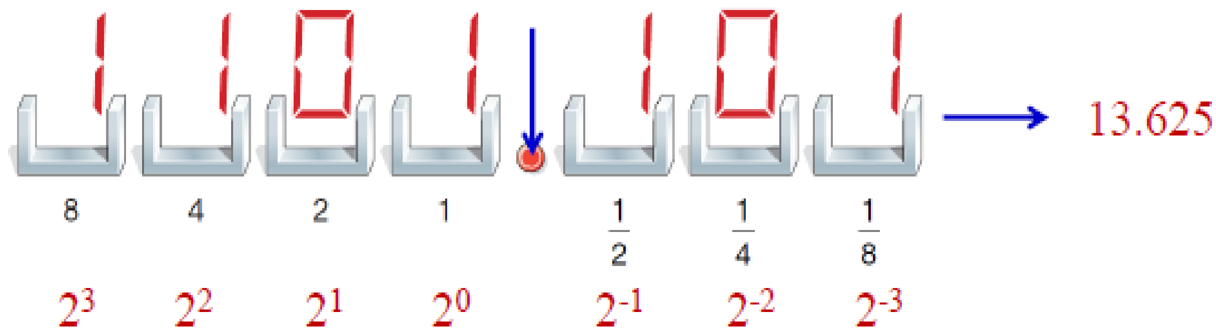
Binary to decimal

$$B_7 * 2^7 + B_6 * 2^6 + B_5 * 2^5 + B_4 * 2^4 + B_3 * 2^3 + B_2 * 2^2 + B_1 * 2^1 + B_0 * 2^0 = 238$$



Quantization A/D and D/A (7)

Decimal with fractions to binary

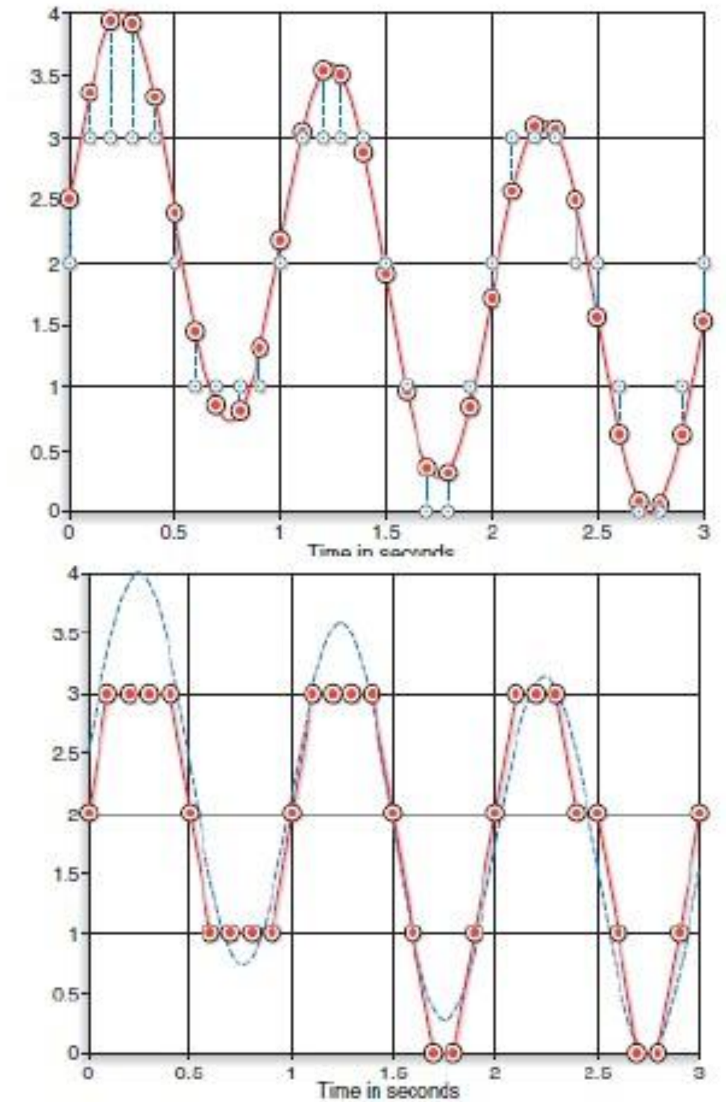


$$1101.101_2 = (1\times2^3)+(1\times2^2)+(0\times2^1)+(1\times2^0)+(1\times2^{-1})+(0\times2^{-2})+(1\times2^{-3})$$

$$13.625_{10} = (1\times10^1)+(3\times10^0)+(6\times10^{-1})+(2\times10^{-2})+(5\times10^{-3})$$

Quantization A/D and D/A (8)

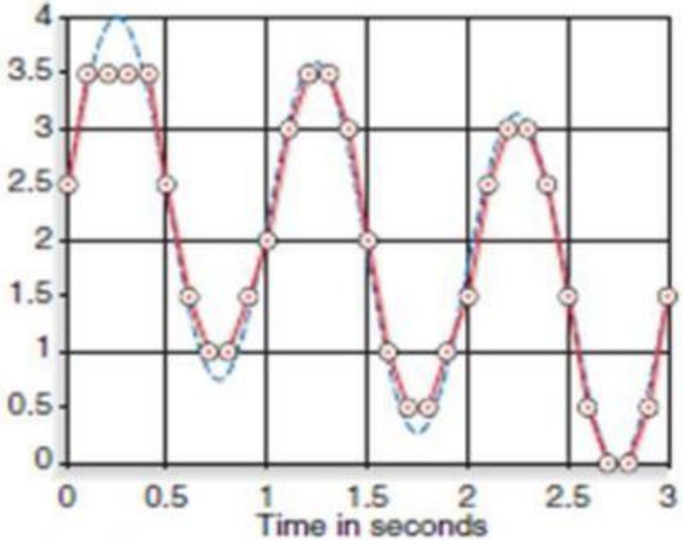
Accuracy in quantization



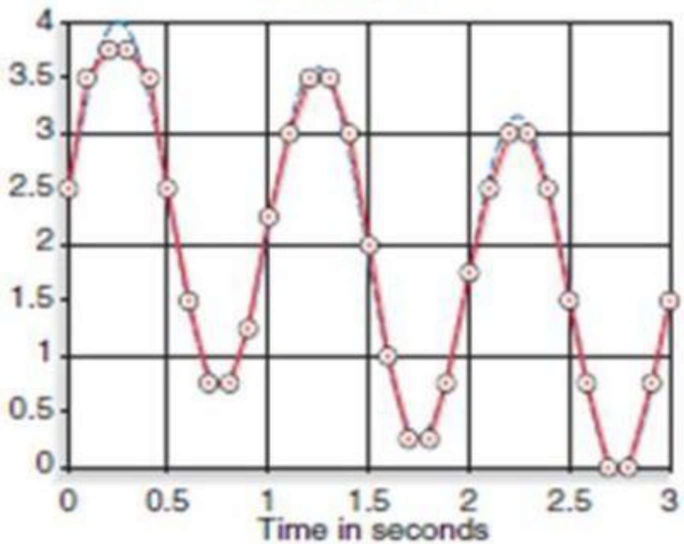
- Quantization rounds to the closest discrete level;
- The accuracy of digitized signal depends on bits used per sample. The error of quantization operation goes down as the number of bits increases;
- Using N bits to store each sample gives 2^N quantization levels.
- 2 bits give 4 levels, 3 bits give 8 levels, 4 bits give 16 levels, 8 bits give 256 levels, 10 bits give 1024 levels, 12 bits give 4096 levels, and 16 bits give 65,536 levels.

Quantization A/D and D/A (9)

Example of quantization accuracy

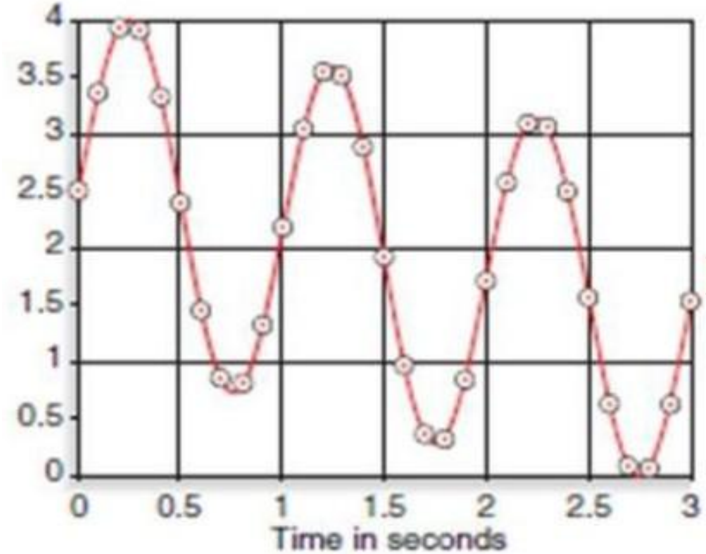


3 Bits



4 Bits

16 Bits



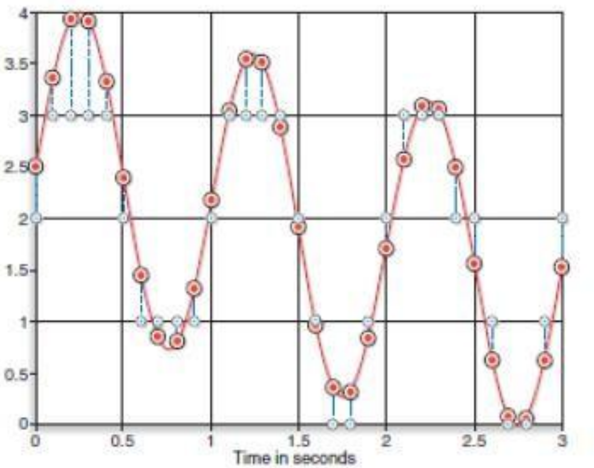
Quantization A/D and D/A (10)

Concept of signal-noise-ratio (SNR)

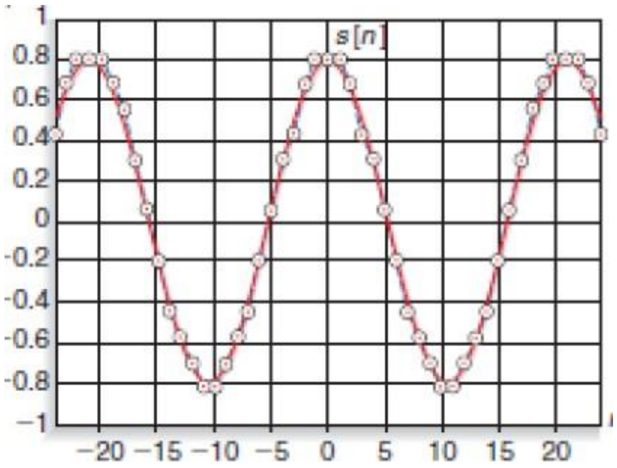
*The **SNR** is the ratio of the maximum signal level magnitude to the maximum noise level magnitude*

Example:

2 bits



4 bits



general formula for SNR

$$SNR = \frac{2^{N-1}}{\frac{1}{2}} = 2^N$$

SNR = 4/1 = 4

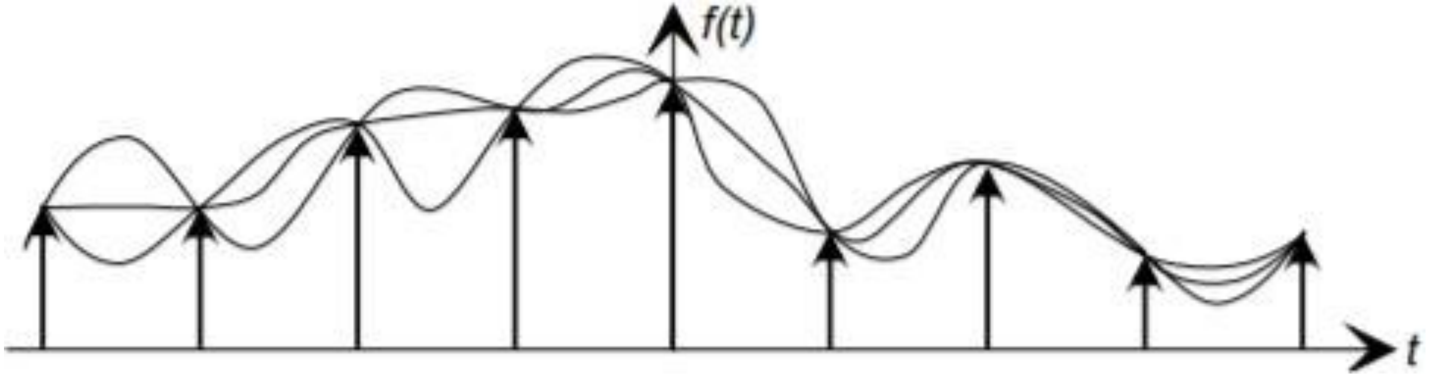
SNR = 0.8/0.0625 = 12.8

Quantization A/D and D/A (11)

Now the Question is:

What is a proper sample rate so that the analog signal can be represented by sampled digital signal?

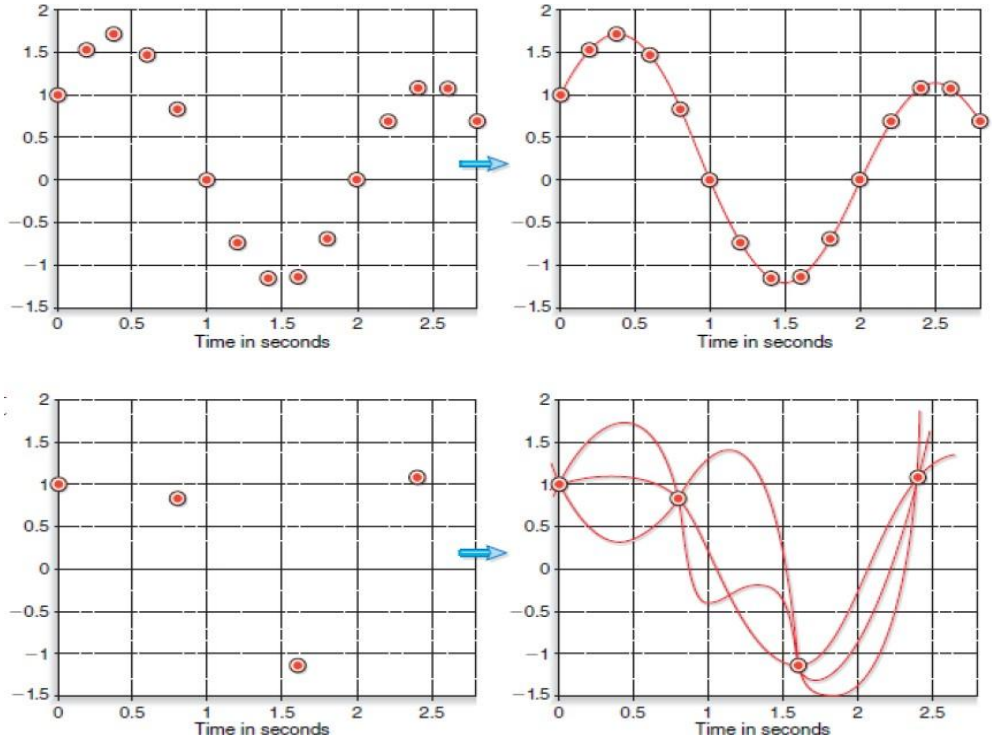
Note that the sampled digital signal only contains the information at point of sample, and therefore, certain analog signal information is lost during the sampling process.



Quantization A/D and D/A (12)

Intuitively, sampling rate should be large enough to extract information rich enough for the signal.

Slow sampling rate **cannot** **recover analog signal**



What is the minimum sampling rate to recover an analog signal ?

Nyquist-Shannon sampling theorem

The sampling frequency (f_s) should be at least twice the highest frequency (f_h) contained in the signal. Or in mathematical terms

$$f_s \geq 2f_h$$

In most of engineering applications, the sampling frequency (f_s) is often selected to be ten times of the highest frequency (f_h) contained in the signal, that is,

$$f_s \geq 10f_h$$

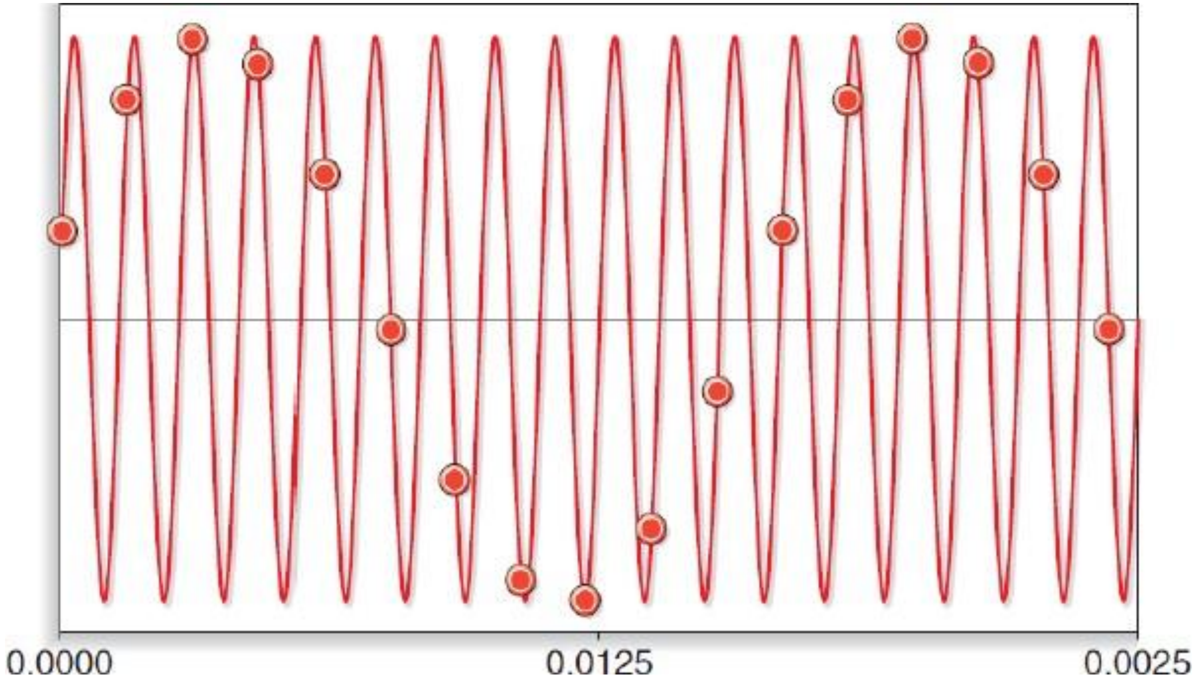
Content

- ❑ Background
- ❑ Continuous-time signal: time and frequency domains
- ❑ Signal discretization: sample and hold
- ❑ Aliasing due to improper sample rate
- ❑ Discrete-time signal: time and frequency (FFT) domains
- ❑ Continuous-time filtering
- ❑ Discrete-time filtering

Aliasing due to digitization (1)

Aliasing: due to under-sampling, the sampled signal indicates a frequency lower than the actual signal.

Example:



A 720 Hz signal sampled at 660 Hz showing a frequency of 720-660 = 60 Hz

Aliasing due to digitization (2)

Consider the continuous-time sinusoidal signal

$$h(t) = Acos(2\pi f_0 t + \varphi)$$

Let's sample the signal at frequency of $f_s = 1/T_s$ such that

$$h(nT_s) = Acos(2\pi f_0 nT_s + \varphi)$$

Note that

$$h(nT_s) = Acos(2\pi f_0 nT_s + \varphi) = Acos[2\pi(f_0 \pm l f_s)nT_s + \varphi]$$

for l = $0, 1, 2, \ldots$

Aliasing due to digitization (2-cont'd)

Consider the continuous-time sinusoidal signal

$$h(t) = Acos(2\pi f_0 t + \varphi)$$

Let's sample the signal at frequency of $f_s = 1/T_s$ such that

$$h(nT_s) = Acos(2\pi f_0 nT_s + \varphi)$$

Note that

$$h(nT_s) = Acos(2\pi f_0 nT_s + \varphi) = Acos[2\pi(f_0 \pm l f_s) nT_s + \varphi]$$

for $l = 0, 1, 2, \ldots$

That means sampling for $Acos[2\pi(f_0 \pm l f_s) nT_s + \varphi]$ with $l = 0,1,2, \ldots$ results equivalent sequences for any n and l.

Example:

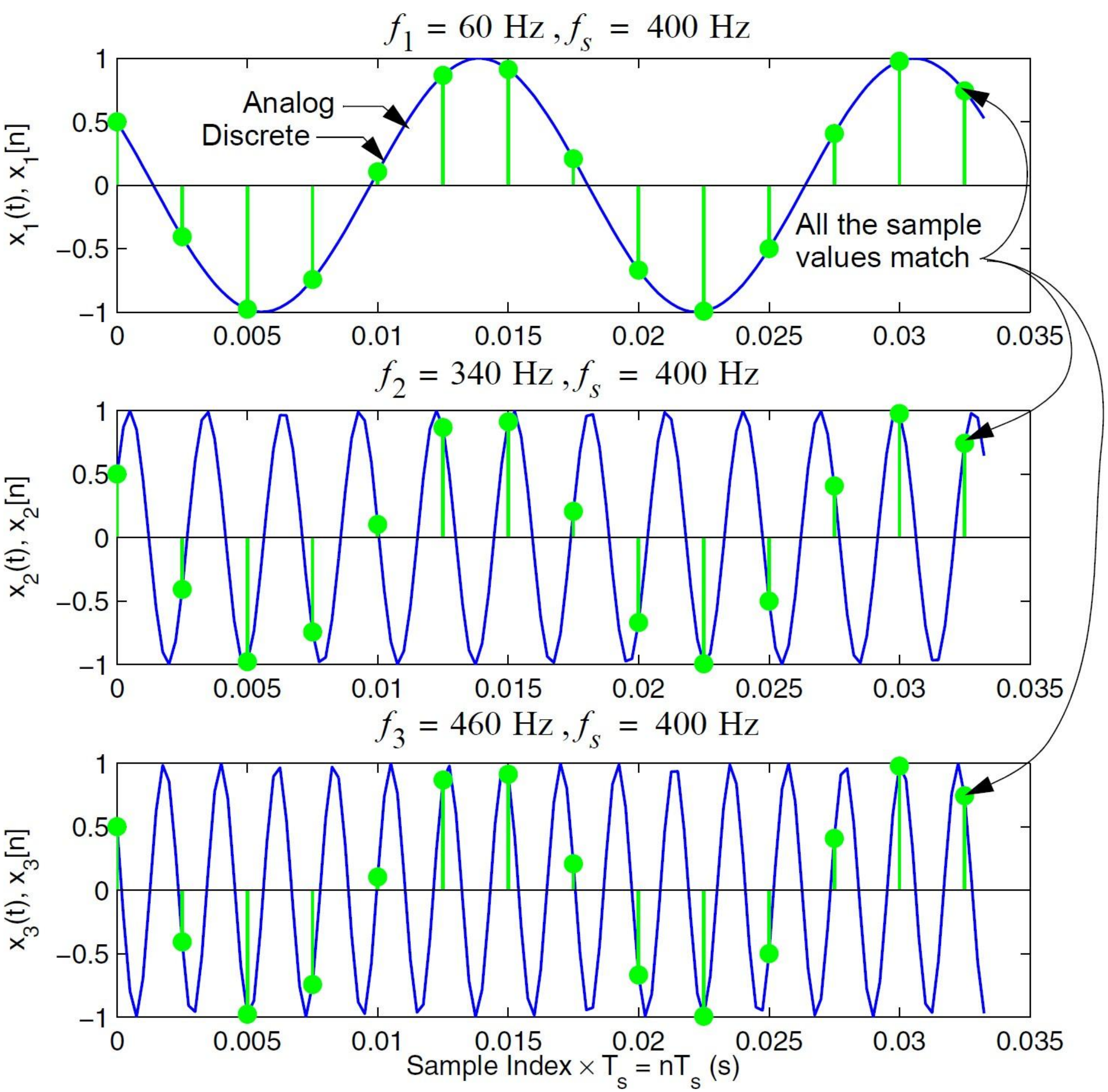
$$x_1(t) = \cos\left(2\pi 60t + \frac{\pi}{3}\right)$$
$$x_2(t) = \cos\left(2\pi 340t - \frac{\pi}{3}\right)$$
$$x_3(t) = \cos(2\pi 460t + \pi/3)$$

Sample at $f_s = 400\ Hz$

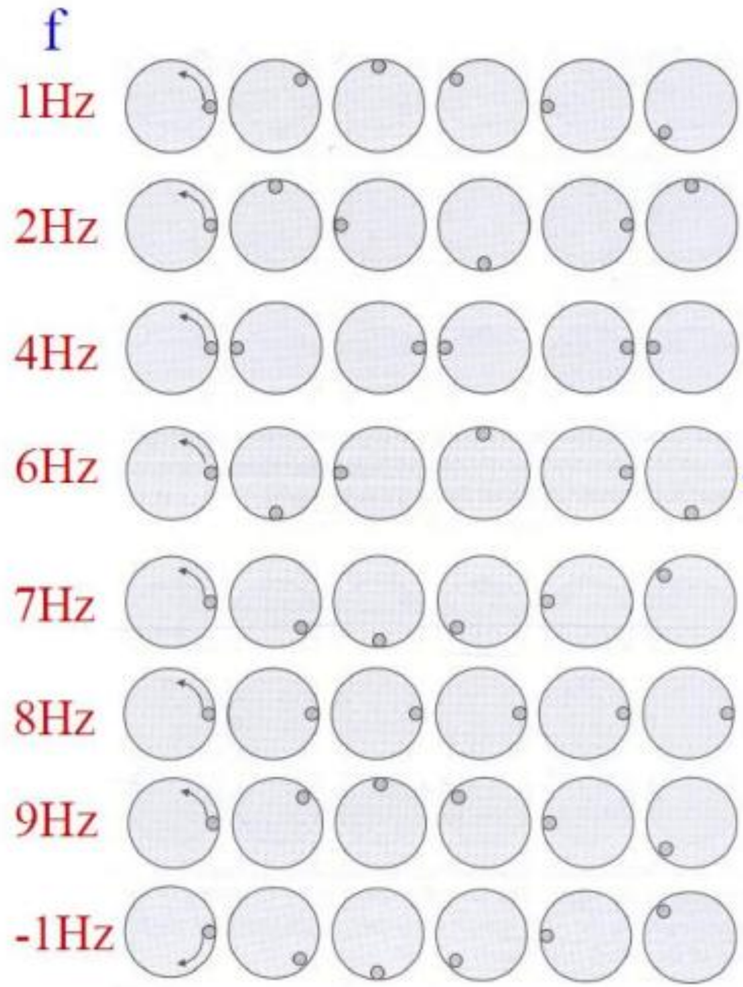
Aliasing due to digitization (3)

This simulation indicates that for a 340 or $460\ Hz$ Cosine function sampled at $400\,Hz$ rate will result in a $60\ Hz$ cosine signal due to aliasing.

Note that in this case the Nyquist- Shannon sample condition is not satisfied



Aliasing due to digitization (4)



<u>How fast must we sample?</u>

Imagine a spinning wheel rotating CCW at f Hz with a single dot near its edge. View the wheel with a strobe light set to flash 8 times per second (8 Hz).

← Which direction is wheel rotating?

← Dot appears to be moving backward.

← Dot definitely appears to be moving backward.

← Is the wheel spinning forward or backward, or has it stopped?

← Image looks the same as when the wheel was rotating at 1 Hz.

← Image looks the same as when the wheel was rotating at 7 Hz.

Aliasing in common life

Video: wheel and fan illusion

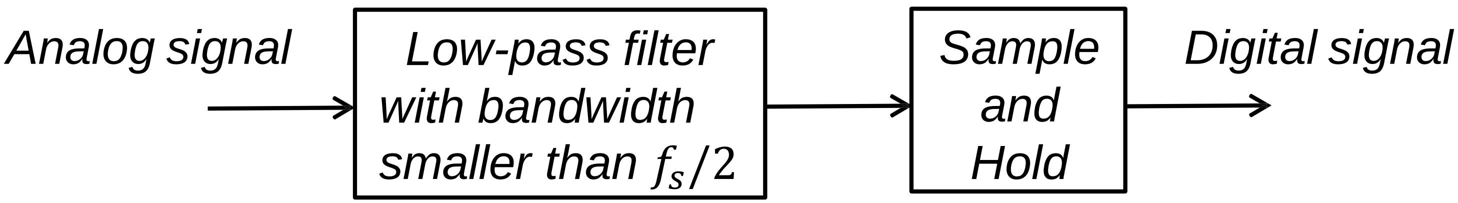
Aliasing due to digitization (5)

The Sampling Theorem

The low-pass sampling theorem states that to avoid aliasing the sample rate f_s should be at least twice that of the highest frequency f_h in the analog signal $x(t)$. Specifically,

$$f_s \geq 2f_h$$

To avoid aliasing in the sampling process, an anti-aliasing filter should be used (see below):



Analog anti-aliasing filter

Note that the anti-aliasing filter shall be an analog filter

Aliasing due to digitization (6)

Example: telephone system

*Human voice max frequency **3600** Hz, ear detection capability frequency **15k** Hz*

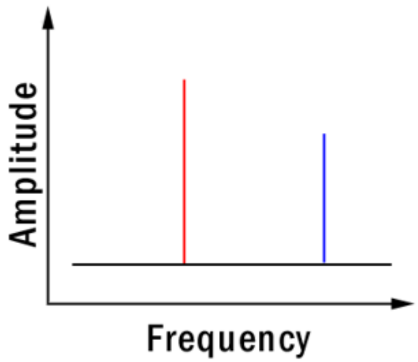
*At the microphone, our voice is filtered by an analog filter to substantially reduce frequencies above **3600** Hz. Then the microphone signal is sampled at **8 k** Hz based in low pass sample theorem to avoid alliasing.*

Since voice signal frequency is shifted by the low-pass filter, it sounds different from original voice.

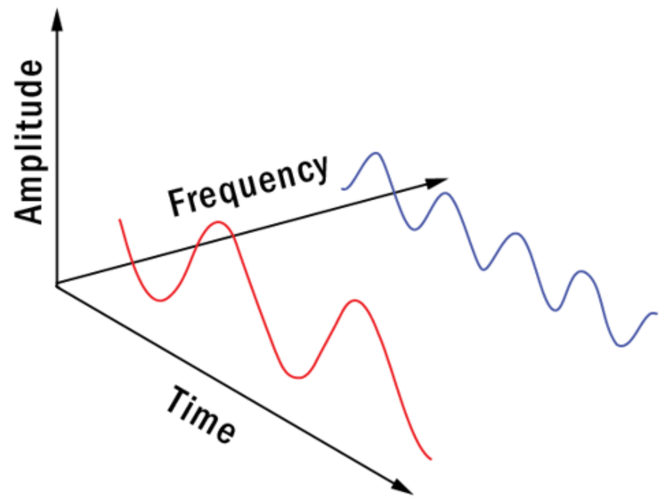
Content

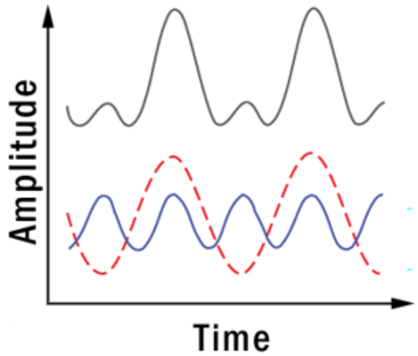
- Background
- Continuous-time signal: time and frequency domains
- Signal discretization: sample and hold
- Aliasing due to improper sample rate
- Discrete-time signal: time and frequency (FFT) domains
- Continuous-time filtering
- Discrete-time filtering

Fourier transformation



[Time & Frequency Domains]





Fourier transformation

For any CT waveform signal x(t), it can be expressed as summation of signals of different frequencies.

	Sinusoidal formulation	Exponential formulation
Synthesis:	$x(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} (a_n \cos(n\Omega_0 t) + b_n \sin(n\Omega_0 t))$	$x(t) = \sum_{n=-\infty}^{+\infty} X_n e^{jn\Omega_0 t}$
Analysis:	$a_n = \frac{2}{T}\int_{t_1}^{t_1+T} x(t)\cos(n\Omega_0 t)dt$ $b_n = \frac{2}{T}\int_{t_1}^{t_1+T} x(t)\sin(n\Omega_0 t)dt$	$X_n = \frac{1}{T}\int_{t_1}^{t_1+T} x(t)e^{-jn\Omega_0 t}dt$

Fundamental frequency $\Omega_0 = {}^{2\pi}/_{T}$

Response of linear system to periodic inputs

Consider a linear single-input, single-output system with a frequency response function $H(j\Omega)$, and $u(t)$ be a periodic signal with T.

$$u(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} A_n \sin(n\Omega_0 t + \phi_n)$$

Harmonic components $u_n(t) = A_n \sin(n\Omega_0 t + \phi_n)$

Principle of superposition will lead output as

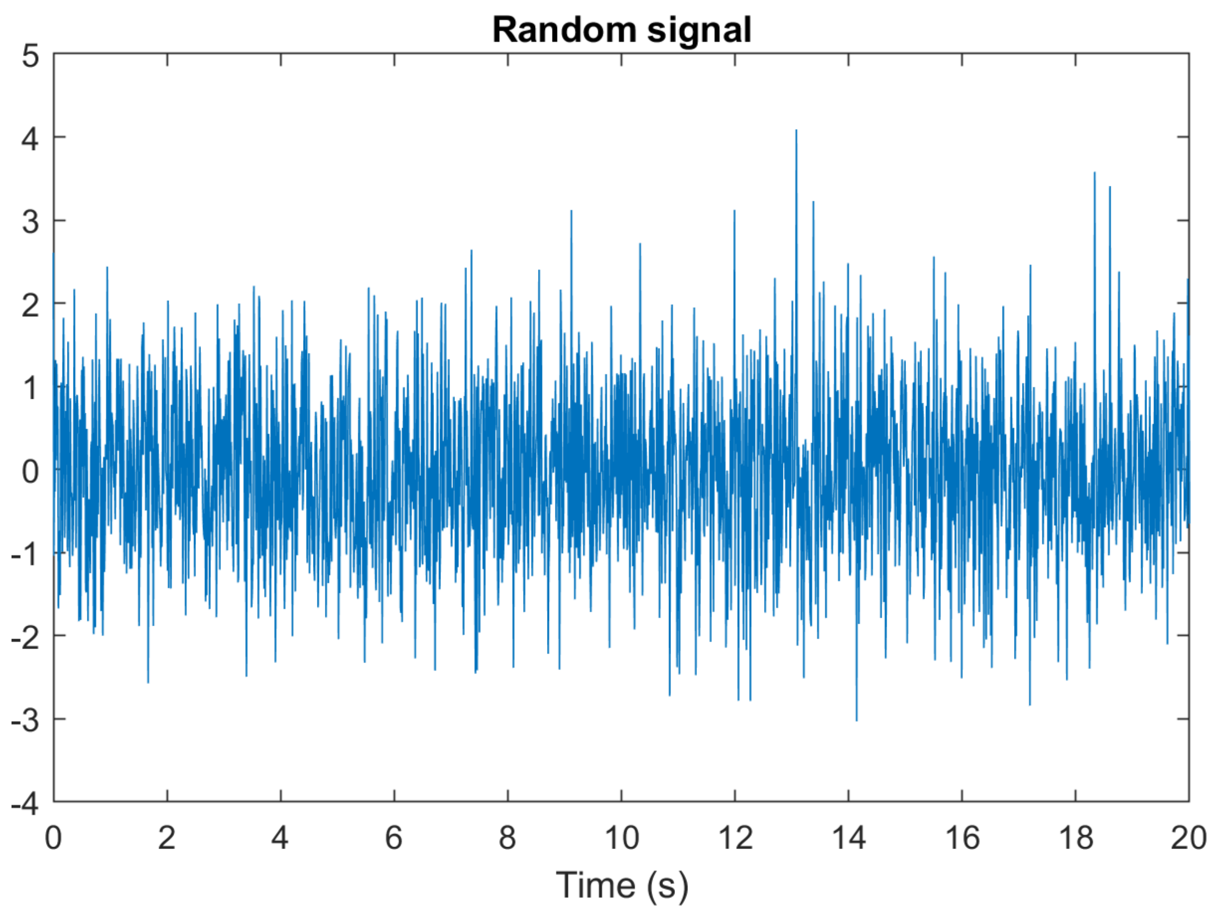
$$y_n(t) = |H(jn\Omega_0)|\, A_n \sin(n\Omega_0 t + \phi_n + \angle H(jn\Omega_0)).$$

$$\begin{aligned} y(t) &= \sum_{n=0}^{\infty} y_n(t) \\ &= \frac{1}{2}a_0 H(j0) + \sum_{n=1}^{\infty} A_n\, |H(jn\Omega_0)| \sin\left(n\Omega_0 t + \phi_n + \angle H(jn\Omega_0)\right) \end{aligned}$$

The output $y(t)$ is also a periodic function with the same period T as the input. But linear system has modified the *relative magnitudes* and shifted phases.

Discrete-time domain signal

A discrete-time white noise generated by Matlab (using command “randn”) that have wide spectrum.



Discrete Fourier Transform (DFT) (1)

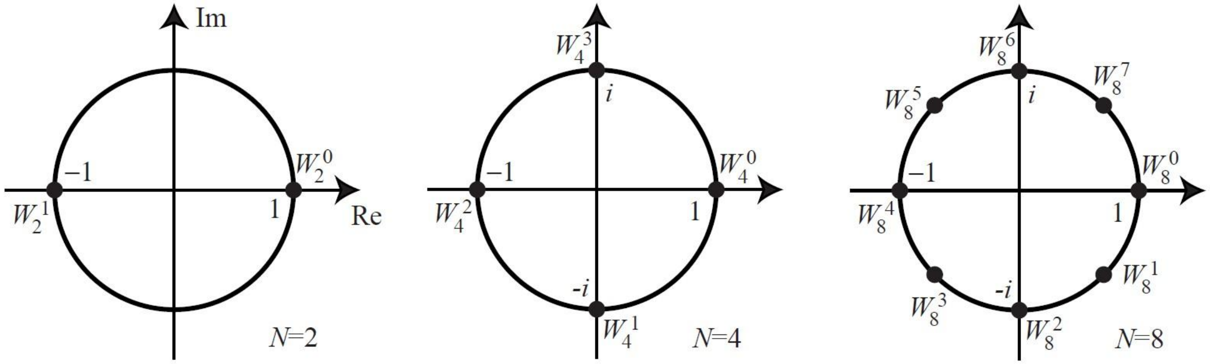
When a signal is discrete and periodic, the discrete Fourier transform (DFT) is used. DFT transforms a sequence of N-length number into another sequence of N-length. Suppose the discrete signal is in the following form:

$$x(n) \to x(k), \qquad n, k = 0,1, \dots, N-1$$

The discrete Fourier transform of $x(k)$ is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-i\frac{2kn\pi}{N}}$$

Where $W_N^k = e^{-i\frac{2k\pi}{N}}$ $(k = 0,1, \dots, N-1)$ are called the *N-th* roots of unity.



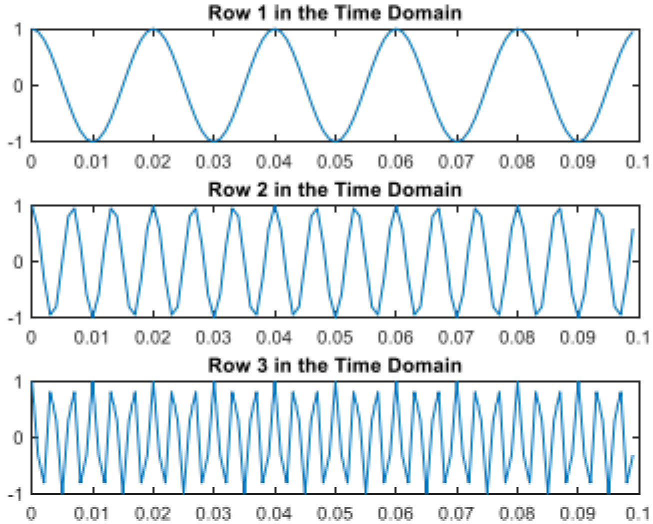
The inverse discrete Fourier transform is defined below,

$$x(n) = \frac{1}{N} \sum_{n=0}^{N-1} X(k) e^{i\frac{2kn\pi}{N}}$$

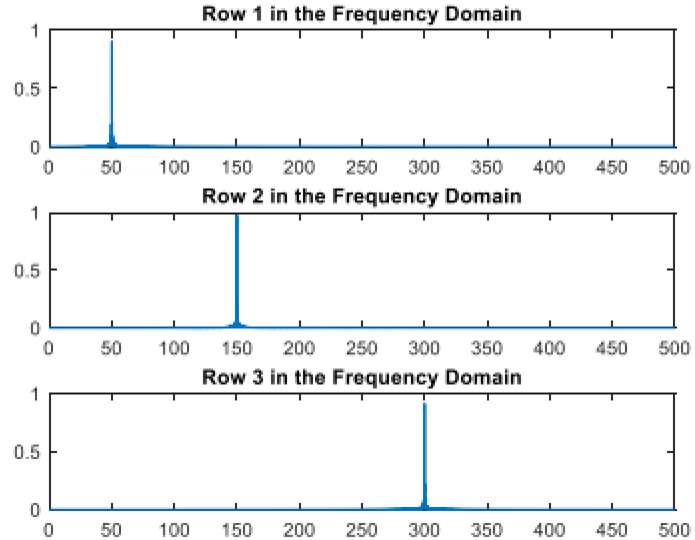
Discrete Fourier Transform (DFT) (2)

Discrete Fourier Transform (DFT) algorithm

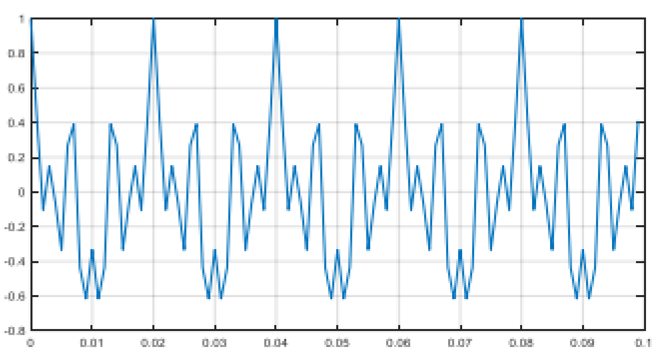
$$x_i = \cos(2\pi\omega_i t)$$

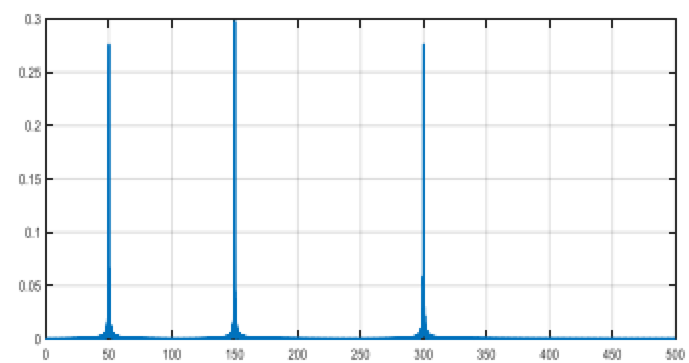


$$\omega_i = 50, 150, 300\ Hz$$



$$x = \Sigma(x_i)$$





Fast Fourier Transform (FFT) (1)

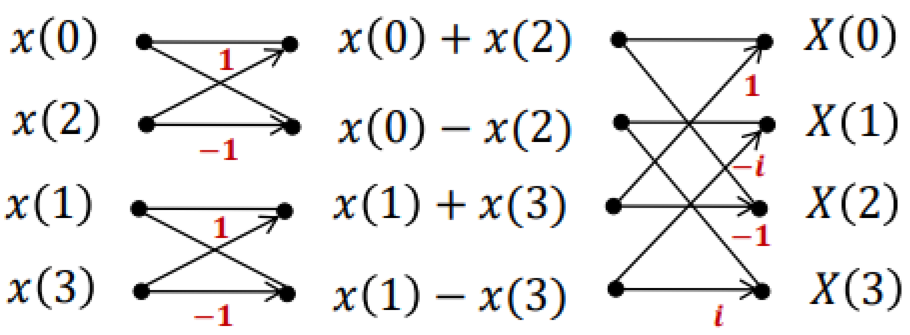
Consider a four-point DFT ($N = 4$) of $x(k)$. Then,

$$X(k) = \sum_{n=0}^{4-1} x(n)e^{-i\frac{kn\pi}{2}} = x(0) + (-i)^k x(1) + (-1)^k x(2) + (i)^k x(3)$$

or

$$X(0) = \big(x(0) + x(2)\big) + \big(x(1) + x(3)\big)$$
$$X(1) = \big(x(0) - x(2)\big) - i\big(x(1) - x(3)\big)$$
$$X(2) = \big(x(0) + x(2)\big) - \big(x(1) + x(3)\big)$$
$$X(3) = \big(x(0) - x(2)\big) + i\big(x(1) - x(3)\big)$$

Using the following calculation mechanism

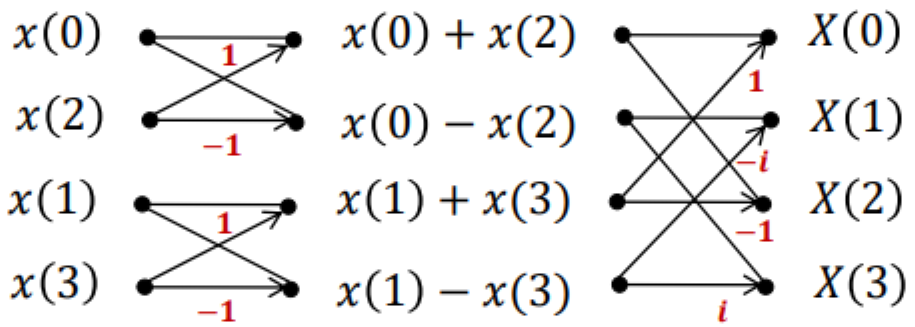


Fast Fourier Transform (FFT) (2)

Note that the four-point discrete Fourier transform below requires 16 multiplies.

$$X(k) = \sum_{n=0}^{4-1} x(n) e^{-i\frac{kn\pi}{2}} = x(0) + (-i)^k x(1) + (-1)^k x(2) + (i)^k x(3)$$

While the calculation proposed below for a four point DFT requires only eight

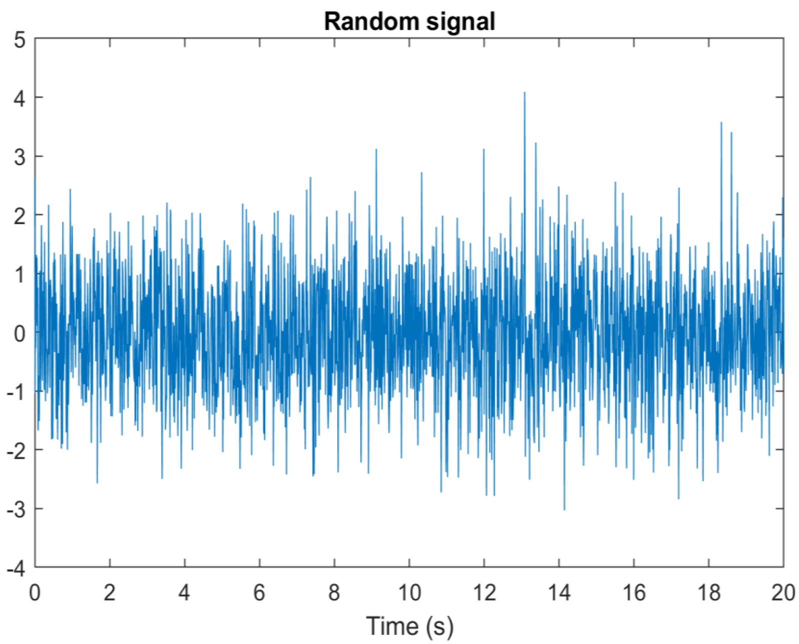


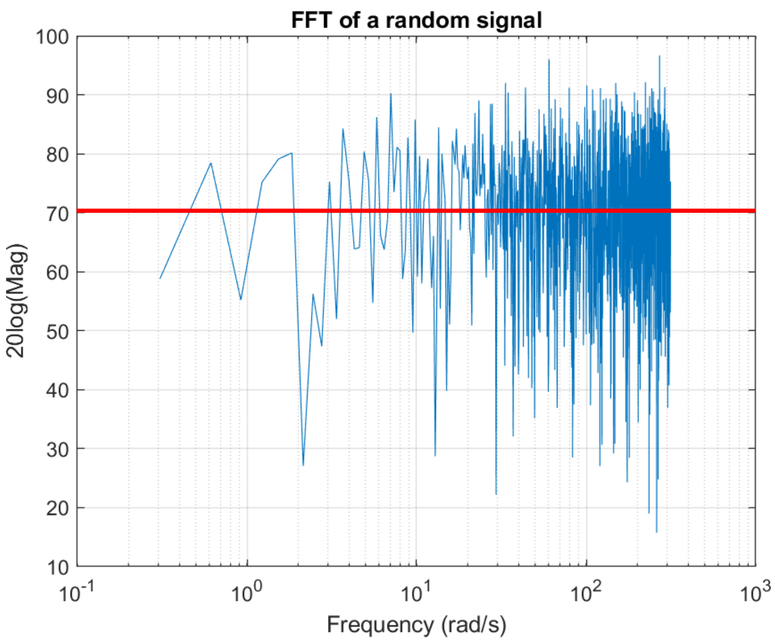
which is a 50% reduction. This calculation scheme can be extended to any number points in the form of $N = 2^m$, where m is a positive integer. This calculation scheme for the DFT is called Fast Fourier Transform (FFT).

Note that as the number of DFT points increases, the percentage of calculation saving increases significantly.

Fast Fourier Transform (FFT) (3)

Fast Fourier Transform (FFT) algorithm





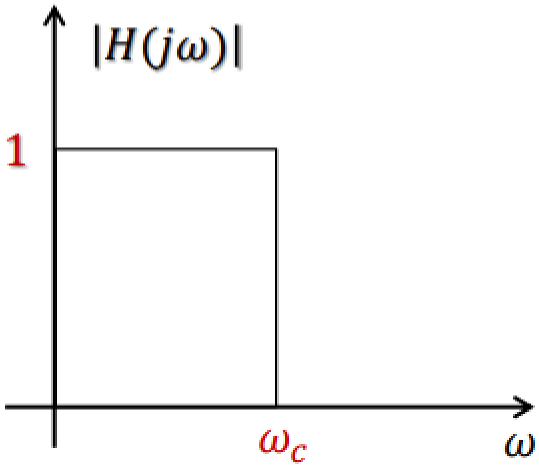
White noise has a uniform spectrum (constant over the frequency)

Content

- ❑ Background
- ❑ Continuous-time signal: time and frequency domains
- ❑ Signal discretization: sample and hold
- ❑ Aliasing due to improper sample rate
- ❑ Discrete-time signal: time and frequency (FFT) domains
- ❑ Continuous-time filtering
- ❑ Discrete-time filtering

Continuous-time filters (1)

1. Ideal Low-Pass Filter represented by transfer function $H(s)$



Cut-off frequency ω_c

Low-pass $H(s)$ characteristics:

number of poles > number of zeros

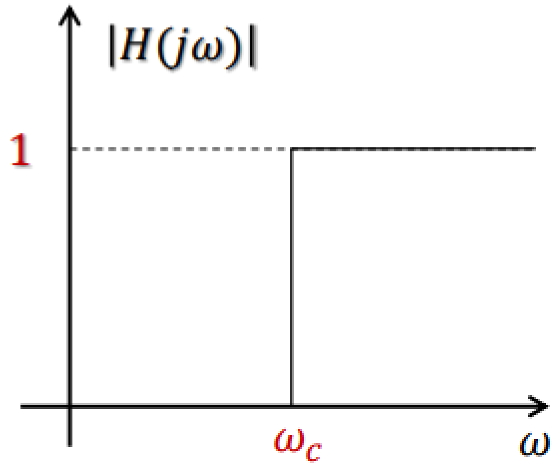
Poles/zeros are not at origin to ensure a finite low-frequency gain

At low frequency $\omega < \omega_c$, $|H(j\omega)| = 1$,
allowing low frequency signal to pass

At high frequency $\omega > \omega_c$, $|H(j\omega)| = 0$,
preventing high frequency signal passing

Continuous-time filters (2)

2. Ideal High-Pass Filter represented by transfer function $H(s)$



Cut-off frequency ω_c

High-pass $H(s)$ characteristics:

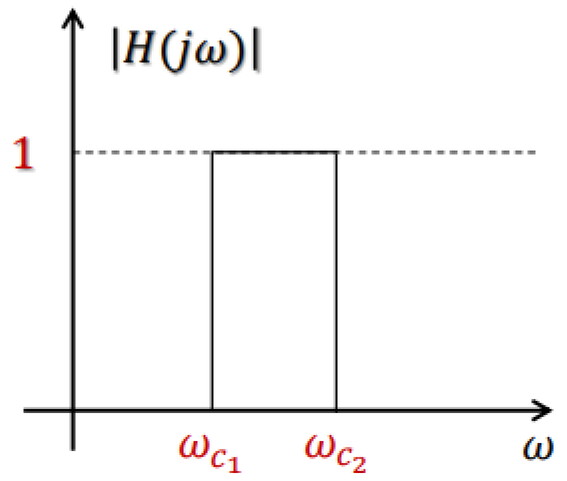
number of poles = number of zeros

At least one zero not at origin to ensure a small low-frequency gain

At low frequency $\omega < \omega_c$, $|H(j\omega)| = 1$,
preventing high frequency signal passing
At high frequency $\omega > \omega_c$, $|H(j\omega)| = 0$,
allowing low frequency signal to pass

Continuous-time filters (3)

3. Ideal Band-Pass Filter represented by transfer function $H(s)$



Cut-off frequencies ω_{c_1} and ω_{c_2}

Band-pass $H(s)$ characteristics:

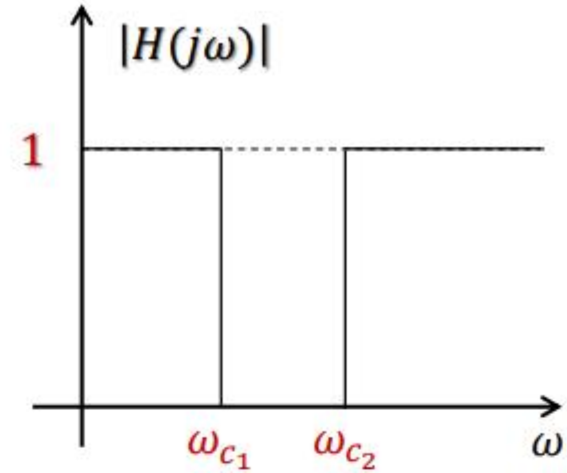
number of poles > number of zeros

At least one zero not at origin to ensure a small low-frequency gain

At frequency $\omega_{c_1} < \omega < \omega_{c_2}$, $|H(j\omega)| = 1$,
allowing signal to pass
At rest of frequency, $|H(j\omega)| = 0$,
preventing signal passing

Continuous-time filters (4)

4. Ideal Band-Stop Filter represented by transfer function $H(s)$



Cut-off frequencies ω_{c_1} and ω_{c_2}

Band-stop $H(s)$ characteristics:

number of poles = number of zeros

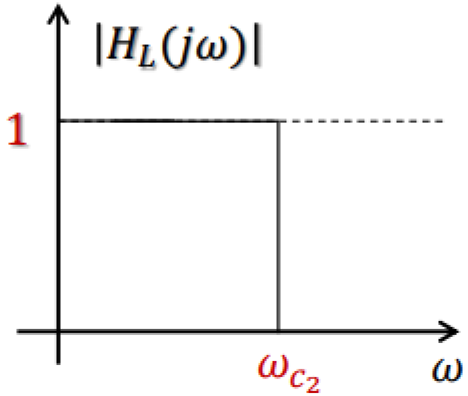
Poles/zeros are not at origin to ensure a finite low-frequency gain

At frequency $\omega < \omega_{c_1} \& \, \omega > \omega_{c_2}$, $|H(j\omega)| = 1$,
allowing signal to pass

At frequency $\omega_{c_1} < \omega < \omega_{c_2}$, $|H(j\omega)| = 0$,
preventing signal passing

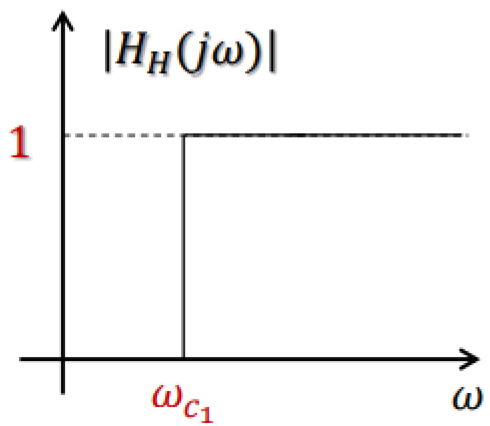
Continuous-time filters (5)

Low-pass filter

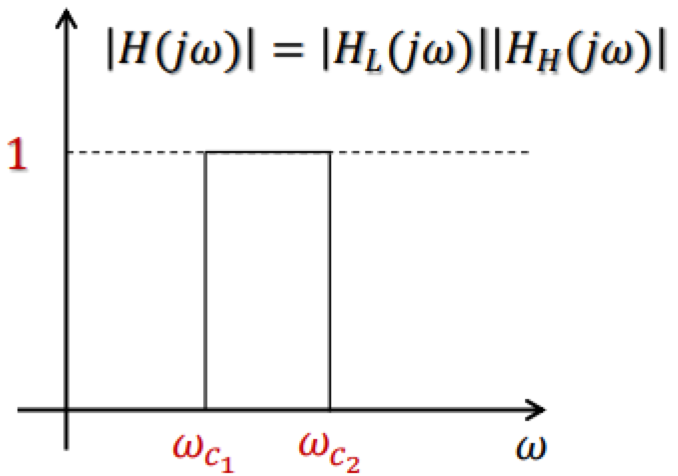


×

High-pass filter



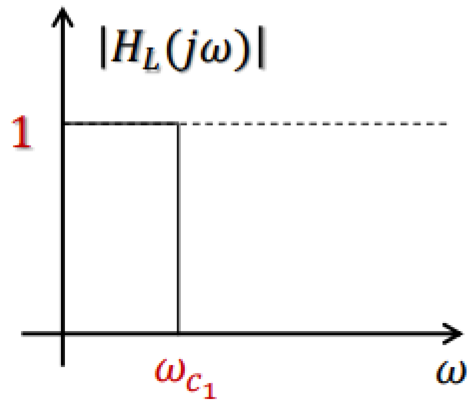
Relationship between band-pass and low/high-pass filters



Low-pass cut-off frequency ω_{c_2} greater than high-pass cut-off frequency ω_{c_1}

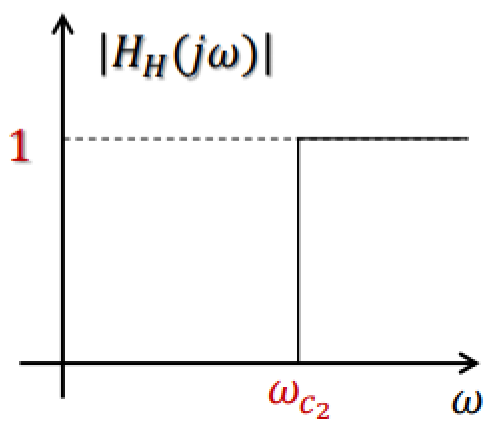
Continuous-time filters (6)

Low-pass filter

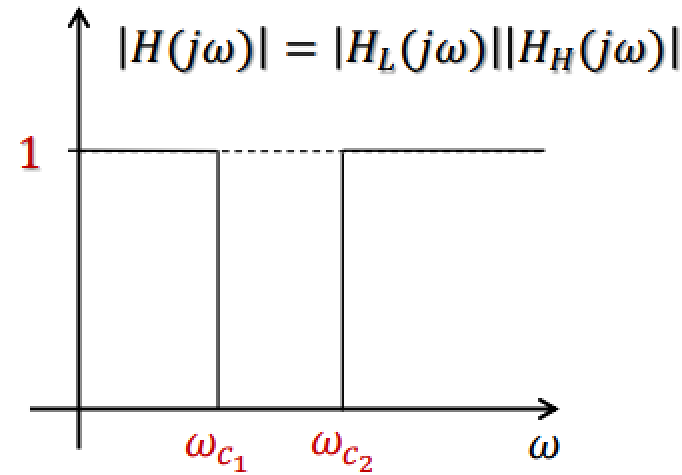


×

High-pass filter



Relationship between band-stop and low/high-pass filters



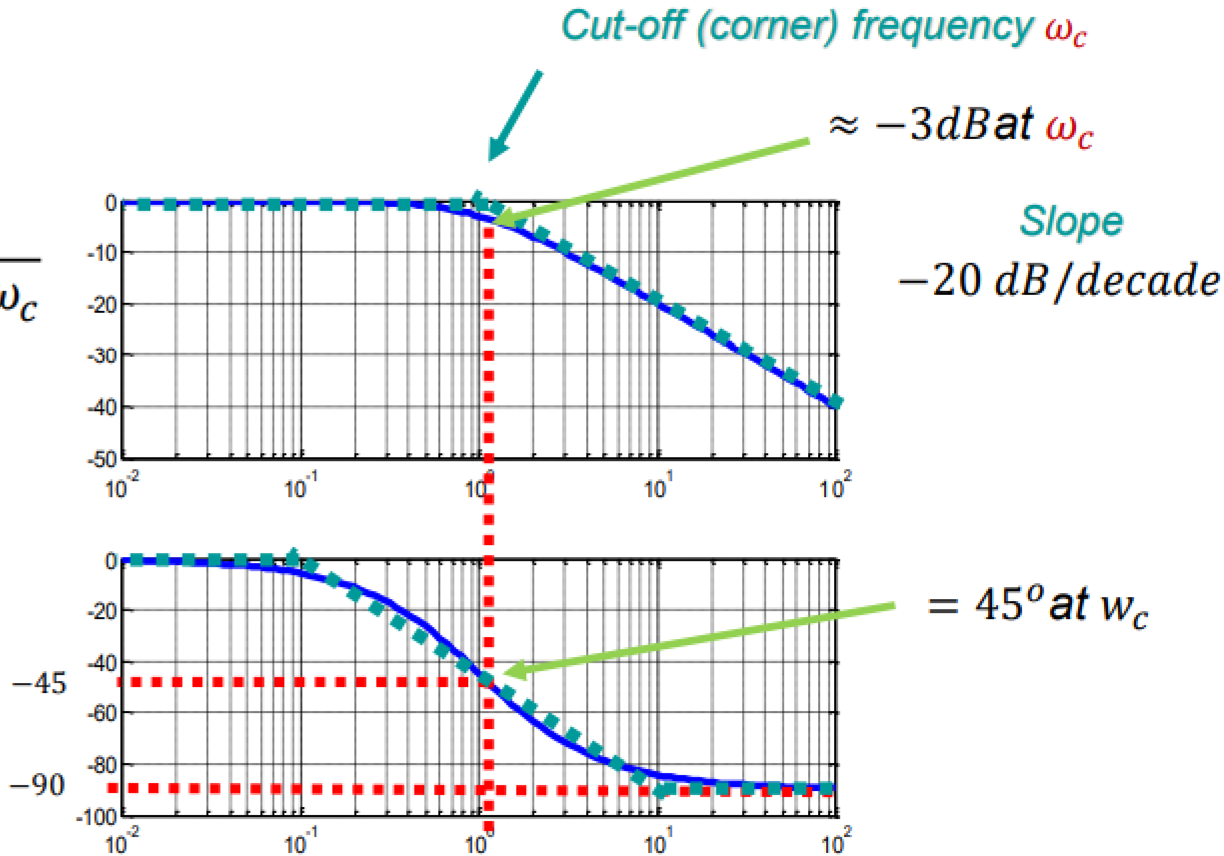
Low-pass cut-off frequency ω_{c_2} *greater than high-pass cut-off frequency* ω_{c_1}

Continuous-time filter design (1)

First-order (1-pole) low-pass filter design (recall Lecture 2)

Transfer function

$$H(s) = \frac{\omega_c}{s+\omega_c} = \frac{1}{1+s/\omega_c}$$



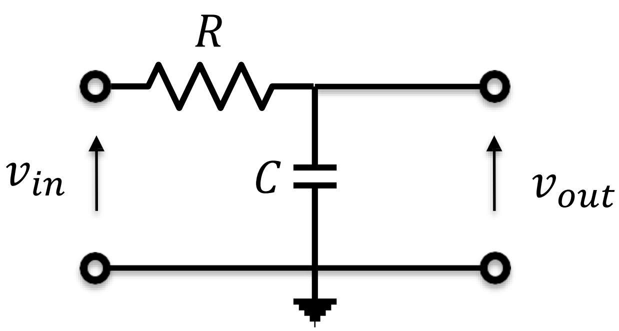
Continuous-time filter design (2)

First-order (1-pole) low-pass filter

Filter realization by analog circuits

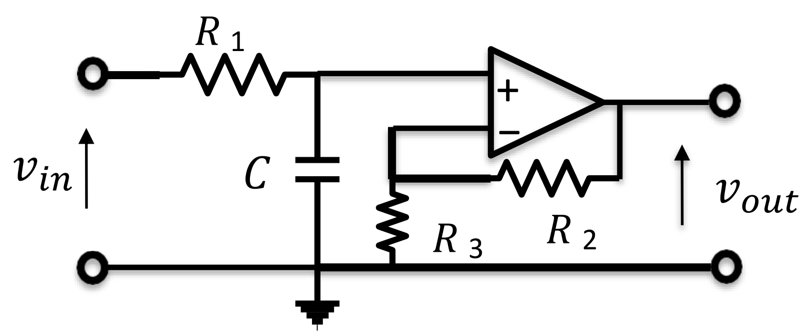
Analog filter by RC circuit

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{1 + sRC} = \frac{1}{1 + s/\omega_c}$$



Analog filter by active RC circuit

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{1 + R_2/R_3}{1 + sR_1C}$$

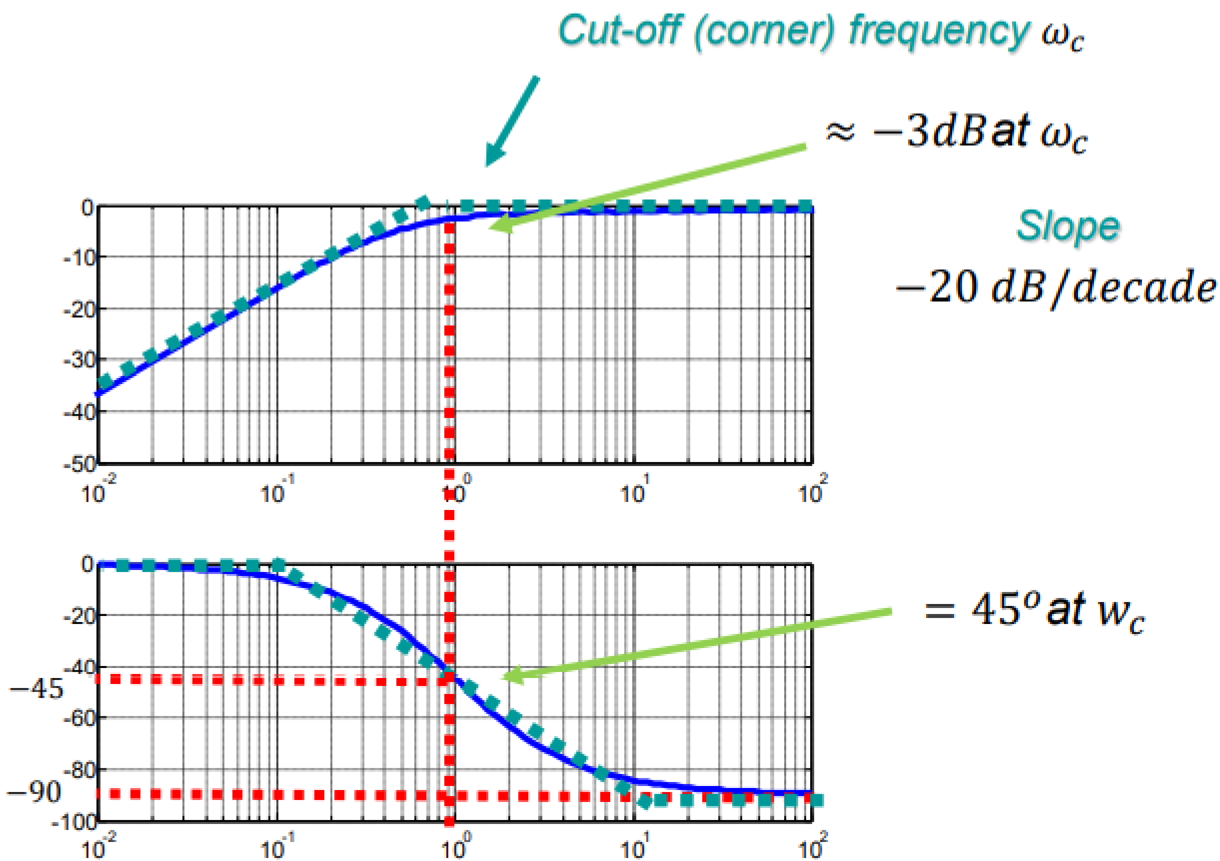


Continuous-time filter design (3)

First-order (1-pole) high-pass filter design (recall Lecture 2)

Transfer function

$$H(s) = \frac{s}{s + \omega_c} = \frac{s/\omega_c}{1 + s/\omega_c}$$



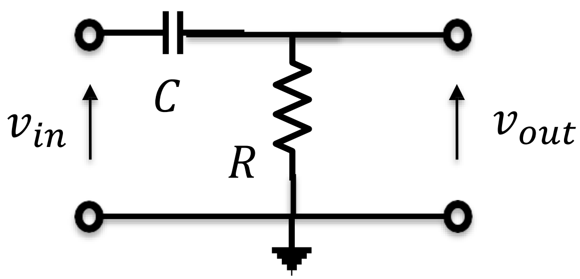
Continuous-time filter design (4)

First-order (1-pole) high-pass filter

Filter realization by analog circuits

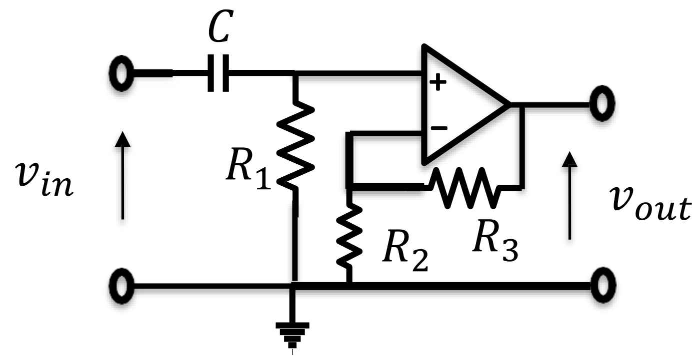
Analog filter by RC circuit

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{s/\omega_c}{1 + s/\omega_c}$$



Analog filter by active RC circuit

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{(1 + R_2/R_3)s/R_1C}{1 + s/R_1C}$$

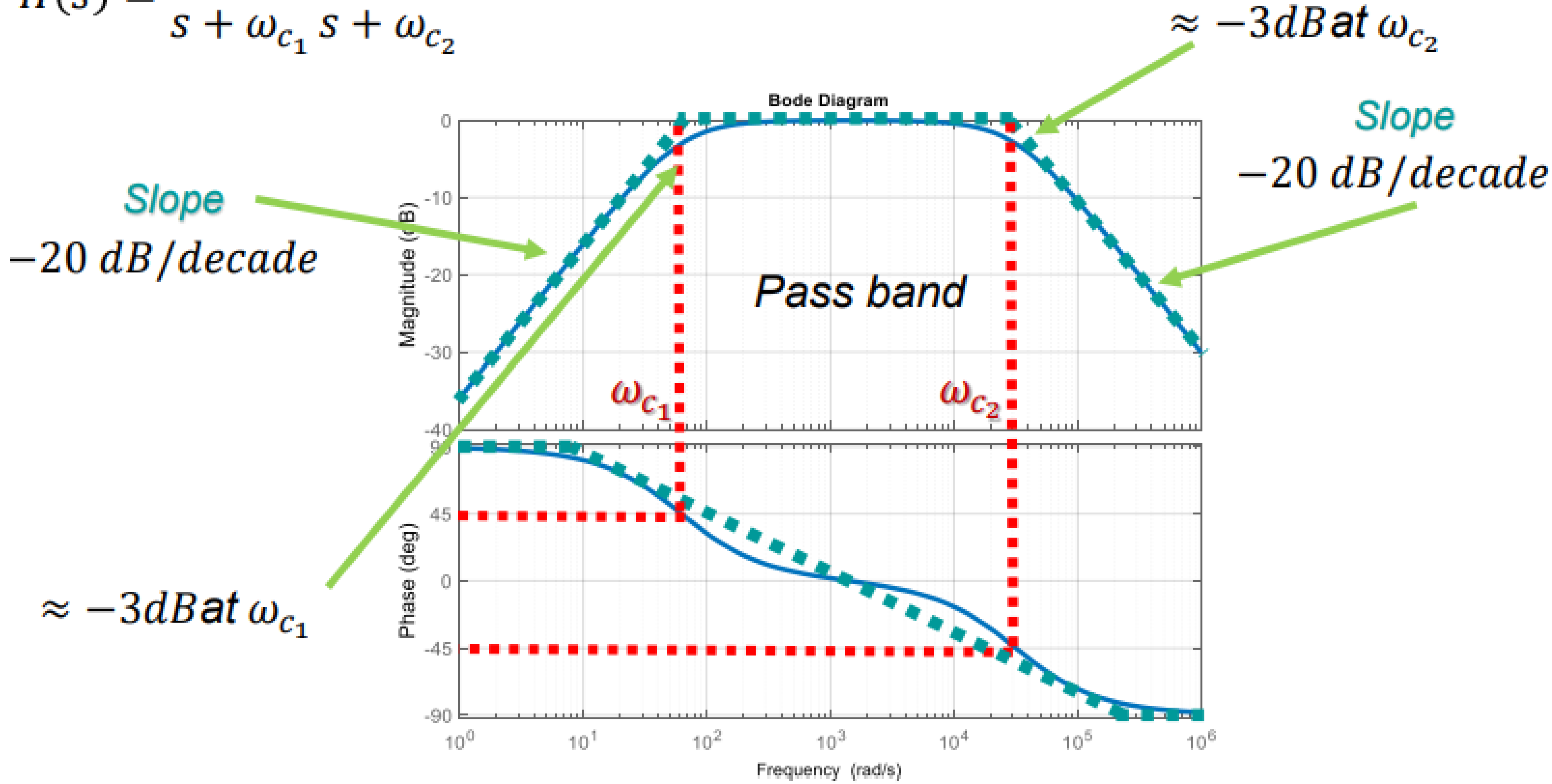


Continuous-time filter design (5)

Second-order (2-pole) band-pass filter design (recall Lecture 2)

Transfer function by multiplying low-pass and high-pass one

$$H(s) = \frac{s}{s+\omega_{c_1}} \frac{\omega_{c_2}}{s+\omega_{c_2}}$$

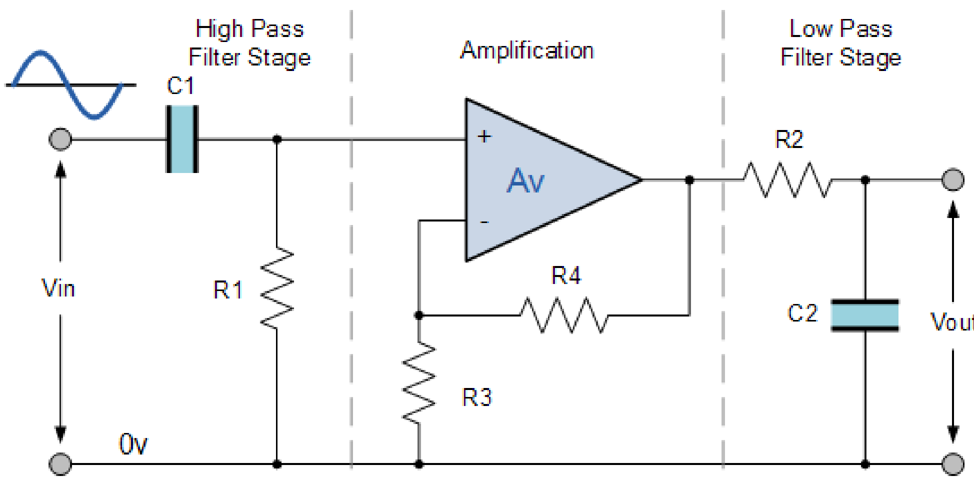


Continuous-time filter design (6)

Band-pass filter

Filter realization by a simple circuit

Analog filter by active RC circuit



$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{s/\omega_{c1}}{1 + s/\omega_{c1}} \frac{\omega_{c2}}{s + \omega_{c2}}$$

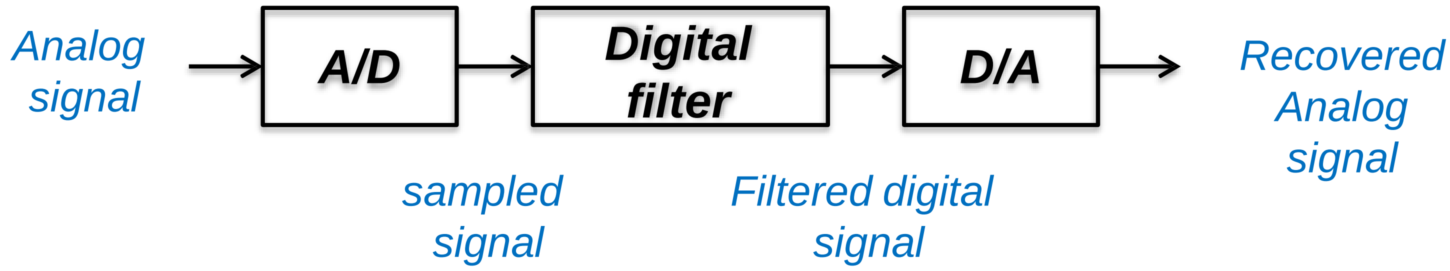
$$\omega_{c1} = \frac{1}{R_1 C_1}, \omega_{c2} = \frac{1}{R_2 C_2}$$

Content

- Background
- Continuous-time signal: time and frequency domains
- Signal discretization: sample and hold
- Aliasing due to improper sample rate
- Discrete-time signal: time and frequency (FFT) domains
- Continuous-time filtering
- Discrete-time filtering

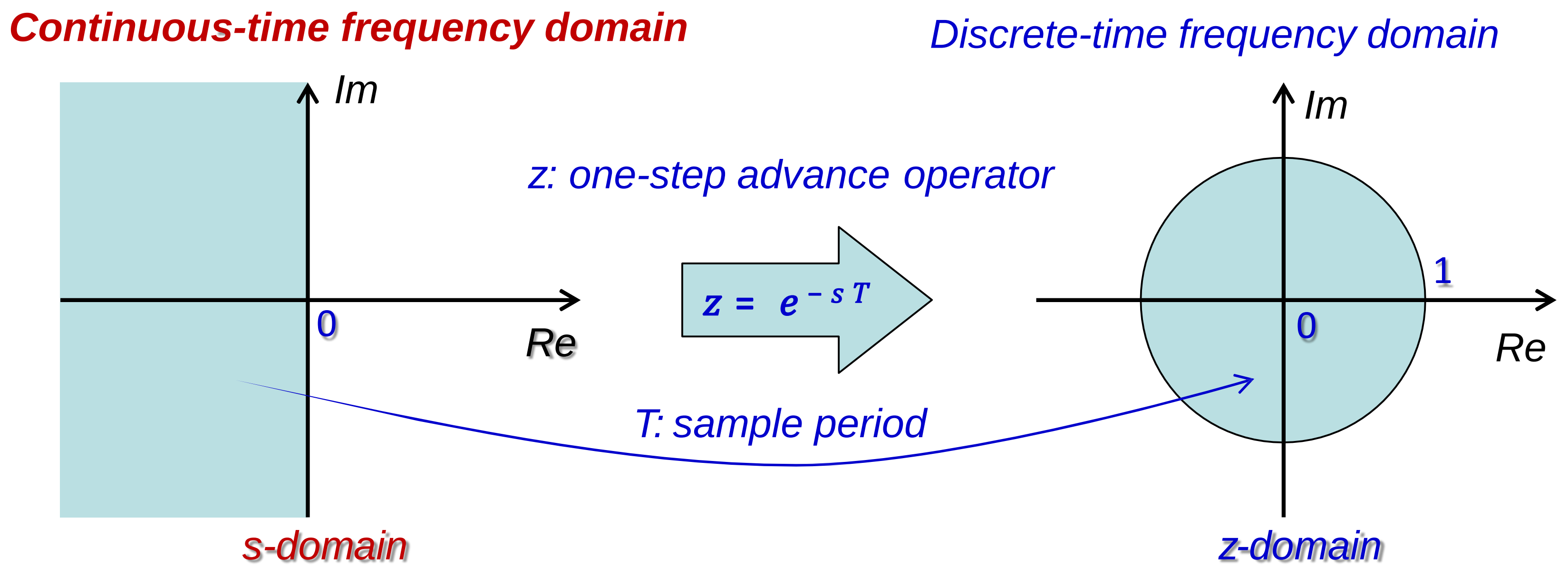
Discrete-time filter design (1)

A digital filter uses digital processors to operate numerical calculations on sampled values of signals.



Advantages of digital filter:
a) Programmable in digital computer without changing hardware;
b) Easy to design, test, and implement;
c) Avoiding hardware limitations, accuracy;
d) Suitable for modern control system and signal processing

Discrete-time filter design (2)



s-domain	Time domain	z-domain
1	$\delta(t)$	1
$\frac{1}{s}$	1	$\frac{z}{z-1}$
$\frac{1}{s^2}$	t	$\frac{Tz}{(z-1)^2}$
$\frac{1}{s+a}$	e^{-at}	$\frac{z}{z-e^{-aT}}$
$\frac{1}{(s+a)^2}$	te^{-at}	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$

Discrete-time filter design (3)

Discrete-time Low-Pass first-order filter

Revisit the continuous-time low-pass filter in the following form

Transfer function

$$H(s) = \frac{\omega_c}{s + \omega_c} = \frac{1}{1 + s/\omega_c}$$

The corresponding discrete-time low-pass filter (see the table in the previous page)

Transfer function

$$\frac{Y(z)}{U(z)} = H(z) = \frac{1 - e^{-\omega_c T}}{1 - e^{-\omega_c T} z^{-1}}$$

Time domain calculation algorithm

Note that from filter transfer function, one can have

$$y(k) = e^{-\omega_c T} y(k-1) + (1 - e^{-\omega_c T}) u(k)$$

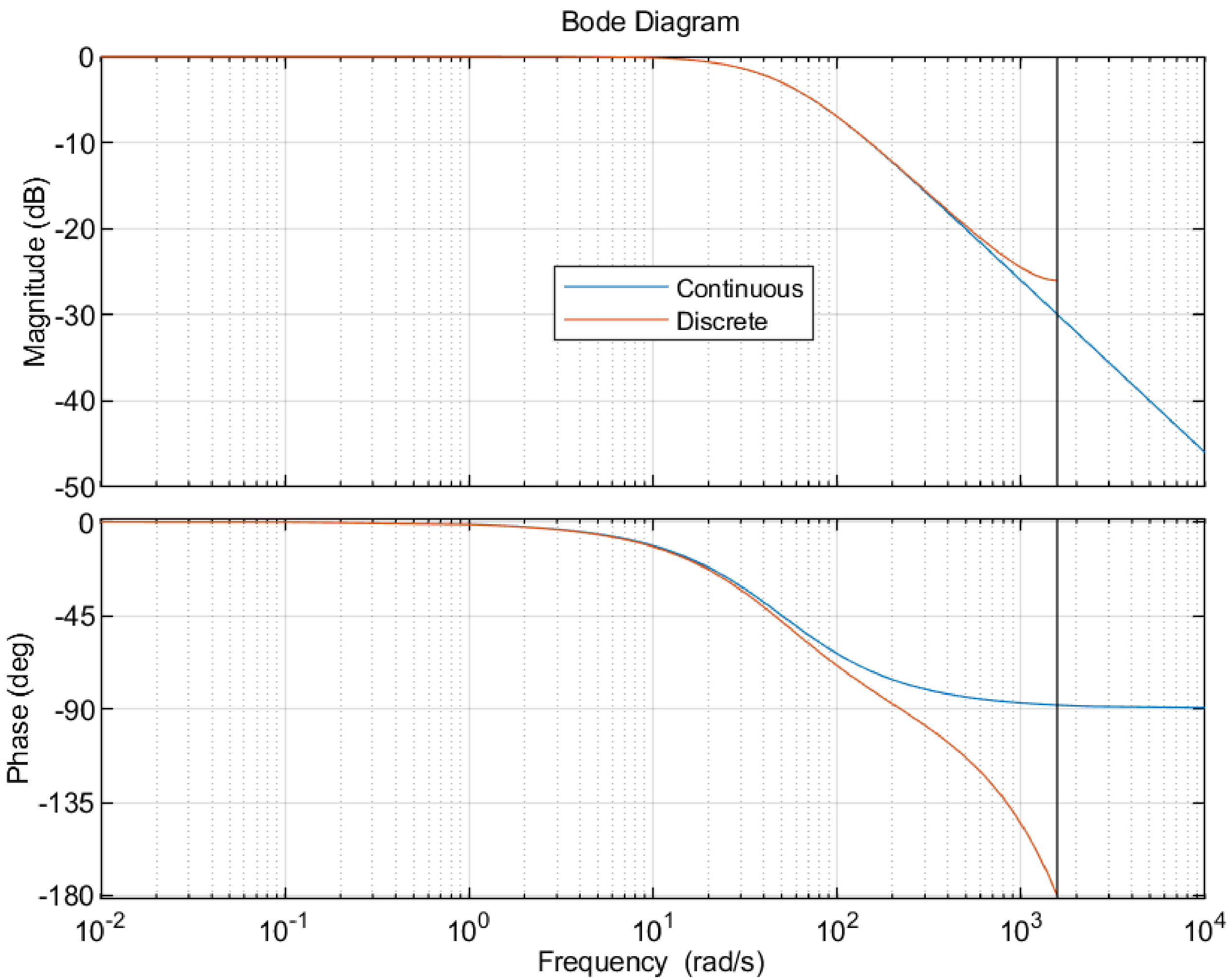
Also, by final value theorem, the DC gain of the discrete-time low-pass filter is one

Note that the final valve theorem for Laplace transform is to set "s" to zero and the corresponding theorem for z-transform is to set "z" to one.

Discrete-time filter design (4)

Bode plot comparison of continuous- and discrete-time Low-Pass filters

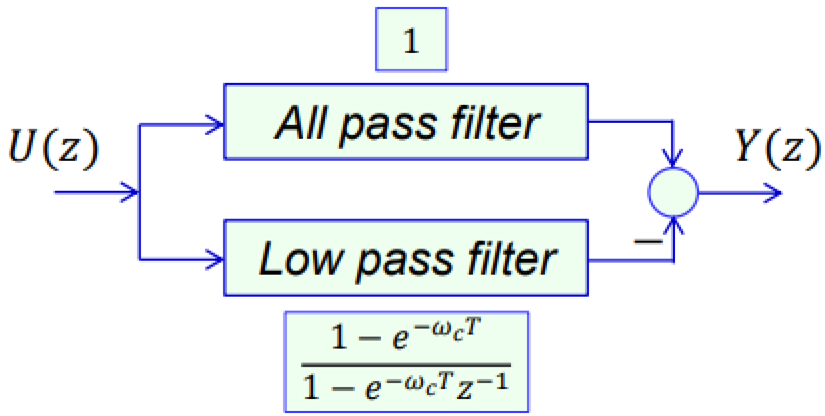
(first-order)



Discrete-time filter design (5)

Discrete-time High-Pass first-order filter

Note that high-pass filter can be considered as the all pass filtered signal minus the low-pass filtered one



Transfer function

$$\frac{Y(z)}{U(z)} = H(z) = 1 - \frac{1-e^{-\omega_c T}}{1-e^{-\omega_c T}z^{-1}}$$
$$= \frac{e^{-\omega_c T}(1-z^{-1})}{1-e^{-\omega_c T}z^{-1}}$$

Time domain calculation algorithm

Note that from filter transfer function, one can have

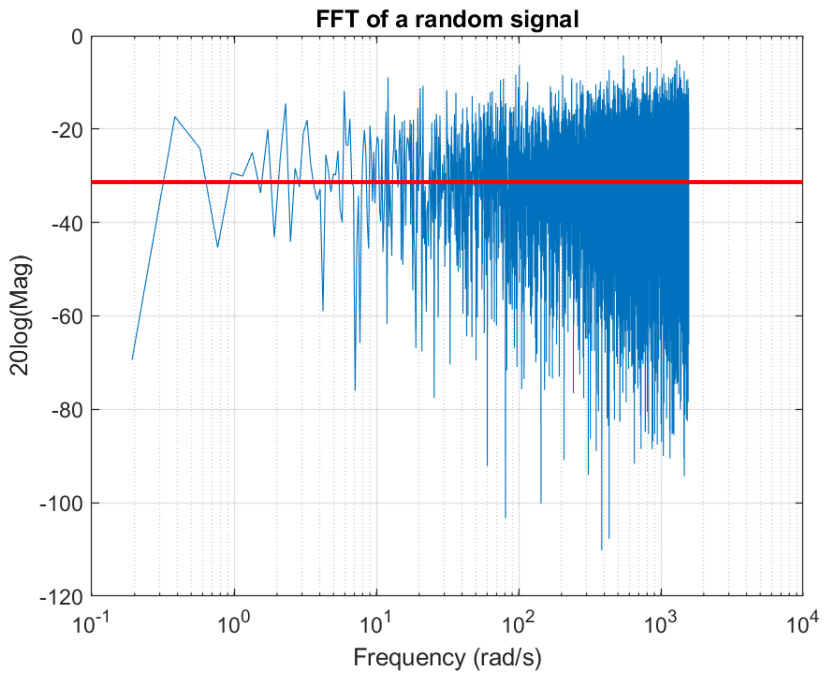
$$y(k) = e^{-\omega_c T}y(k-1) + e^{-\omega_c T}(u(k)-u(k-1))$$

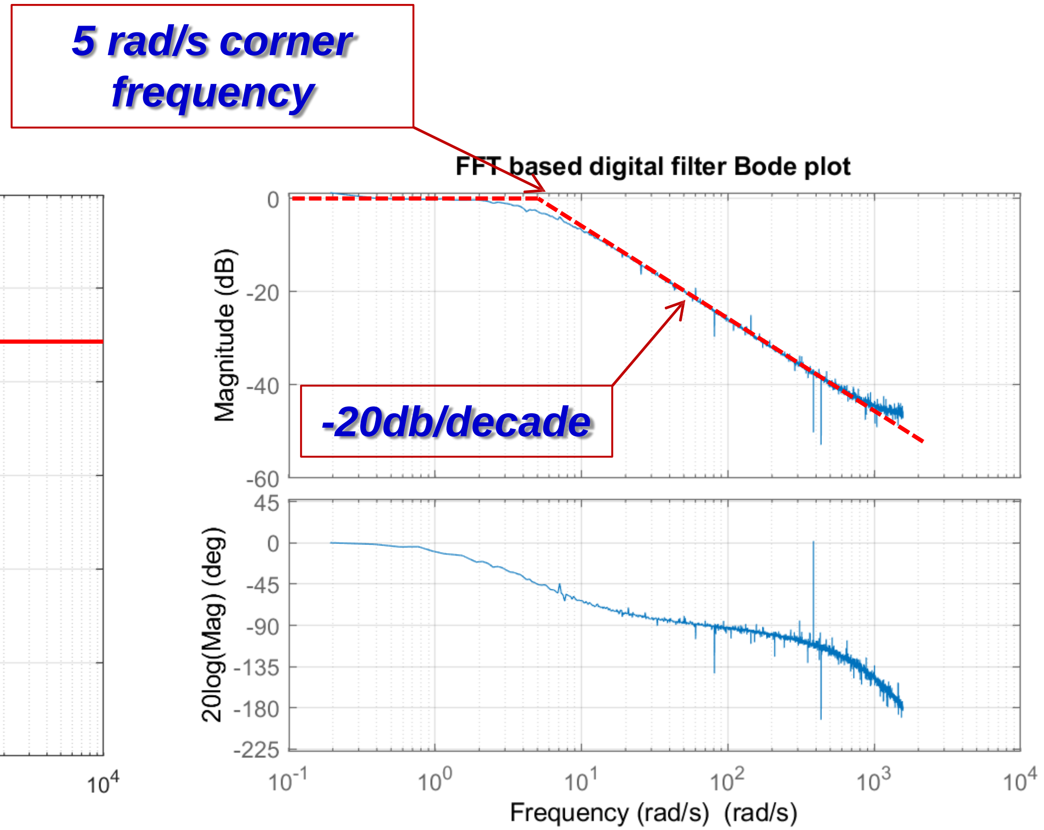
Also, by final value theorem, the DC gain of the discrete-time low-pass filter is zero

Discrete-time filter design (6)

Discrete-time Low-Pass first-order filter

$$H(s) = \frac{a}{s+a} \quad (a = 5, T_s = 0.01), \qquad H(z) = \frac{1-b}{1-bz^{-1}} \quad (b = e^{-aT_s} = 0.9512)$$





Discrete-time filter design (7)

Discrete-time Low-Pass first-order filter

$$H(s) = \frac{a}{s+a} \quad (a = 5, T_s = 0.01), \qquad H(z) = \frac{1-b}{1-bz^{-1}} \quad (b = e^{-aT_s} = 0.9512)$$