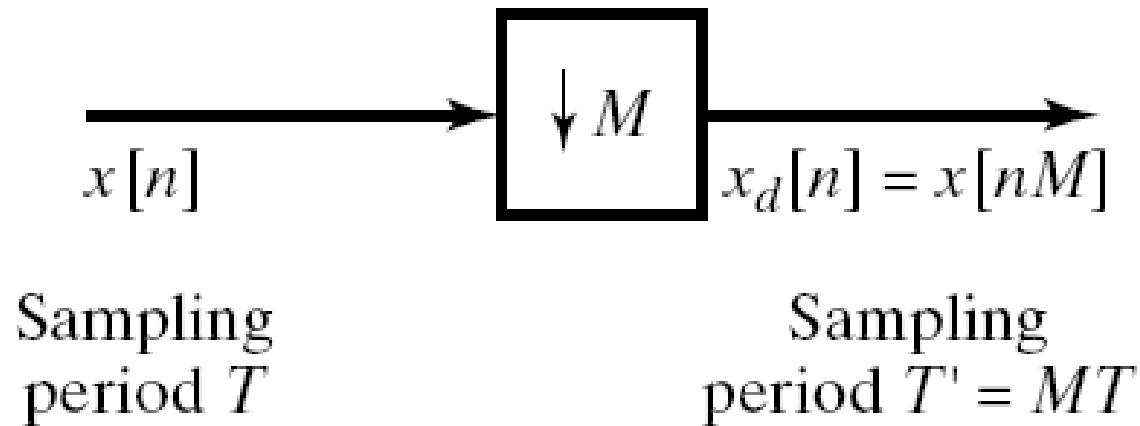


Changing the Sampling rate using discrete-time processing

- What happens when sampling in the discrete domain?



- **downsampling**; sampling rate compressor;

$$x_d[n] = x[nM]$$

Frequency domain of downsampling

- This is a **re-sampling** process. When sampling a continuous-time signal $x_c(t)$, i.e., $x[n] = x_c(nT)$ we have the following equation in the frequency domain.

$$\text{DTFT} \quad X(e^{j\omega}) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c\left(j\left(\frac{\omega}{T} - \frac{2\pi k}{T}\right)\right) \quad \text{CFT}$$

- Hence, for the down-sampled signal $x_d[m] = x_c(mT')$, (where $T' = MT$), we have

$$\text{DTFT} \quad X_d(e^{j\omega}) = \frac{1}{T'} \sum_{r=-\infty}^{\infty} X_c\left(j\left(\frac{\omega}{T'} - \frac{2\pi r}{T'}\right)\right) \quad \text{CFT}$$

Frequency domain of downsampling

- **Property:** the downsampling can be treated as a re-sampling process. Its DTFT frequency domain relationship is similar to that of sampling continuous-time signals as:

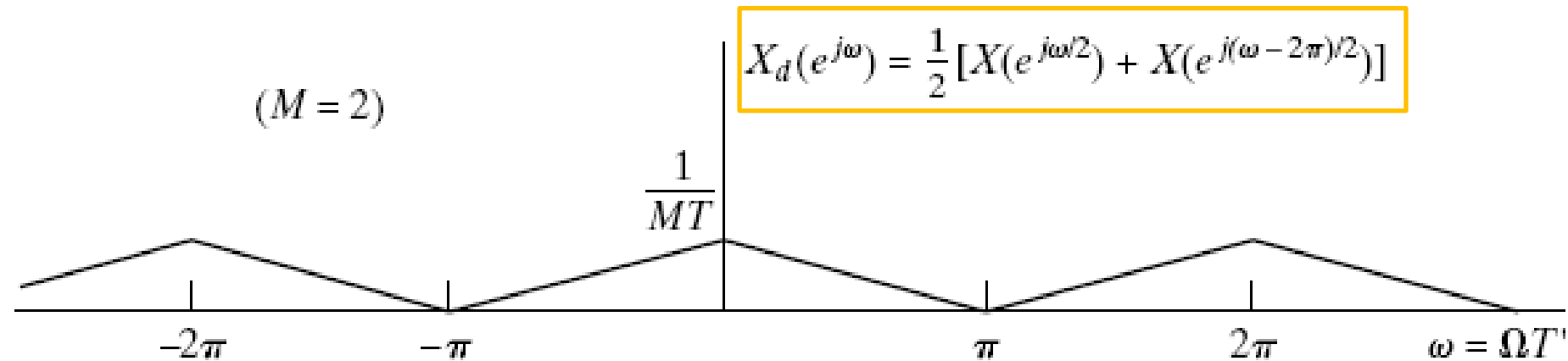
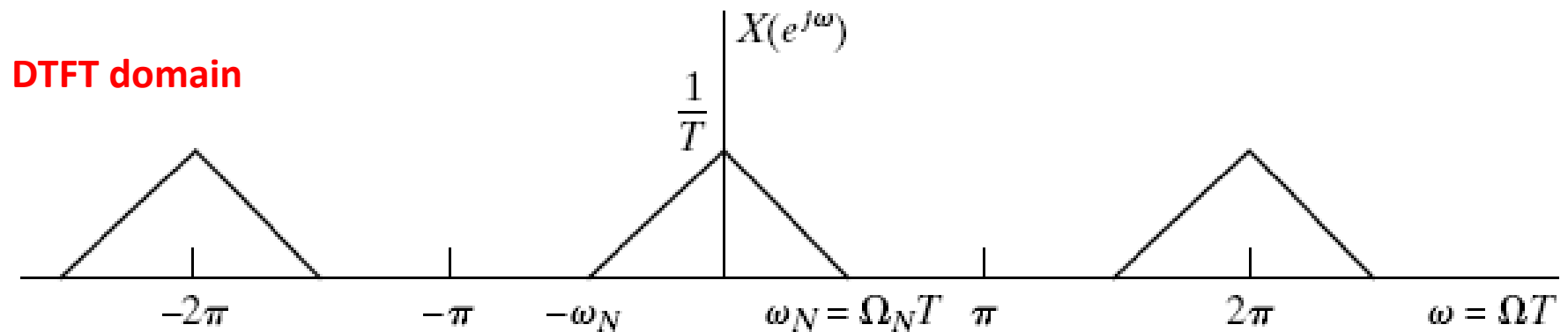
$$X_d(e^{j\omega}) = \frac{1}{M} \sum_{i=0}^{M-1} X\left(e^{j\frac{\omega - 2\pi i}{M}}\right)$$

- Let $g(\omega) = \frac{1}{M} X(e^{j\frac{\omega}{M}})$, i.e., the frequency scaled by M and amplitude scaled by $1/M$ of $X(e^{j\omega})$.
- Let $g_i(\omega) = \frac{1}{M} X(e^{j\frac{\omega - 2\pi i}{M}})$, then $g_i(\omega) = g(\omega - 2\pi i)$, i.e., the shift of $g(\omega)$ by of $2\pi i$.

Frequency domain of downsampling

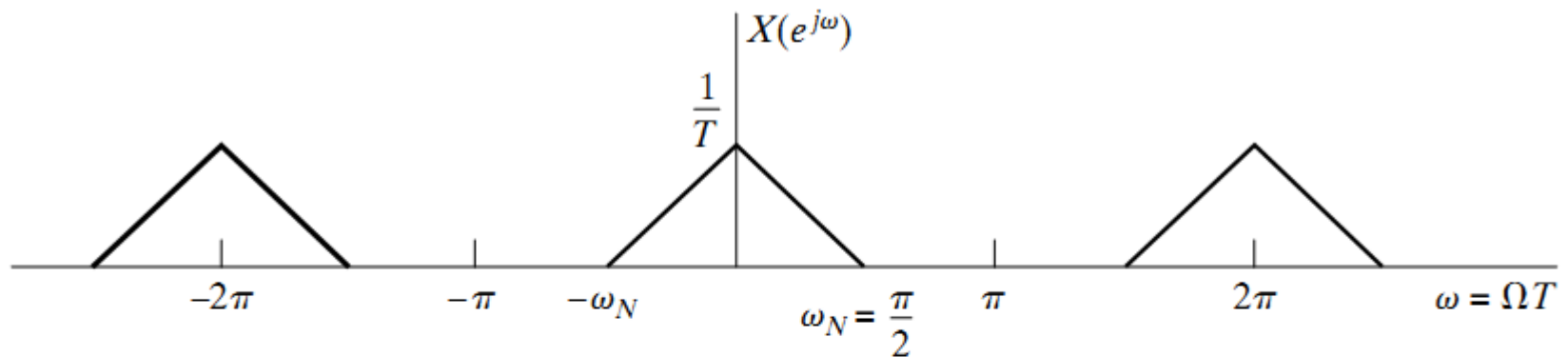
- The above equation is thus equivalent to
 - compositing M copies of the of $X(e^{j\omega})$, each is
 - frequency scaled by M (becomes flat),
amplitude scaled by $1/M$ (becomes short),
 - and shifted by integer multiples of 2π .

Downsampling by $M = 2$ in the Frequency domain (**without aliasing**)

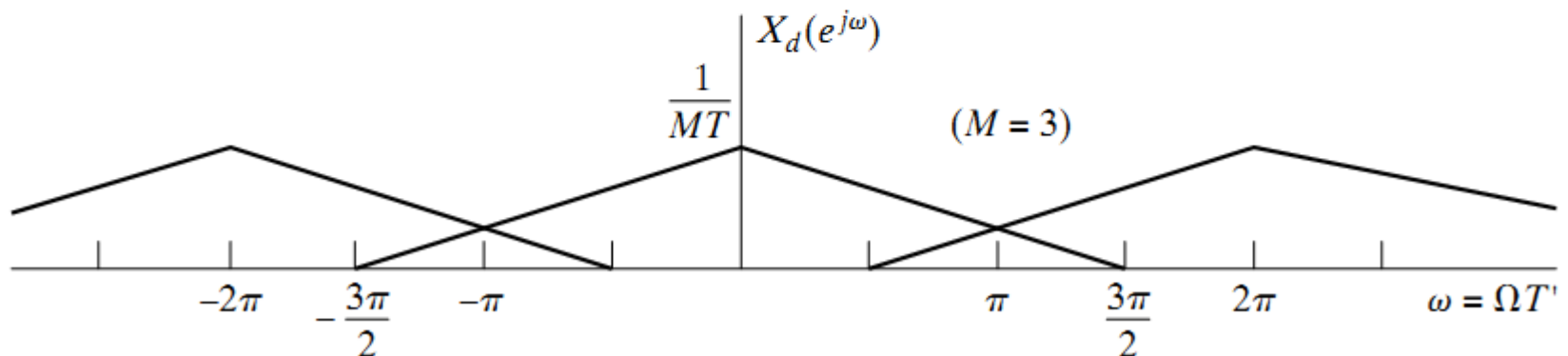


Downsampling the discrete-time signal by 2 ($M = 2$)
(Assume $\omega_N = \pi/2$)

Downsampling (example of $M = 3$) in the Frequency domain (with aliasing)



(b)



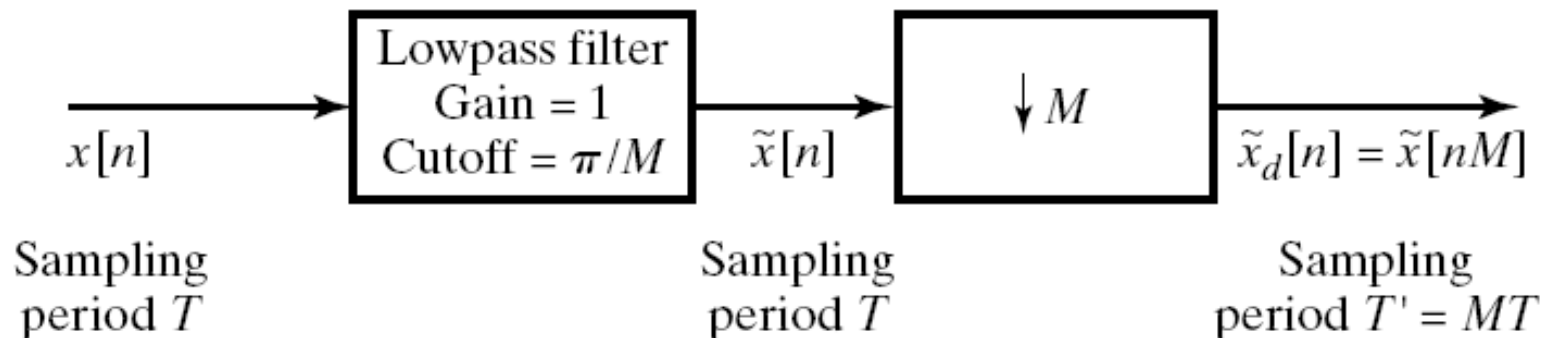
(c)

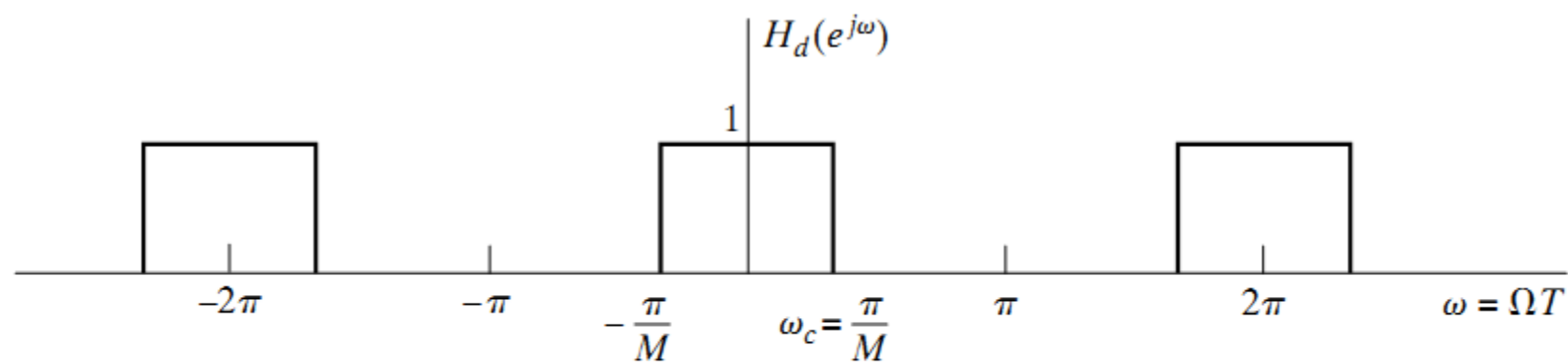
Downsampling the discrete-time signal by 3 ($M = 3$)

(Assume $\omega_N = \pi/2$)

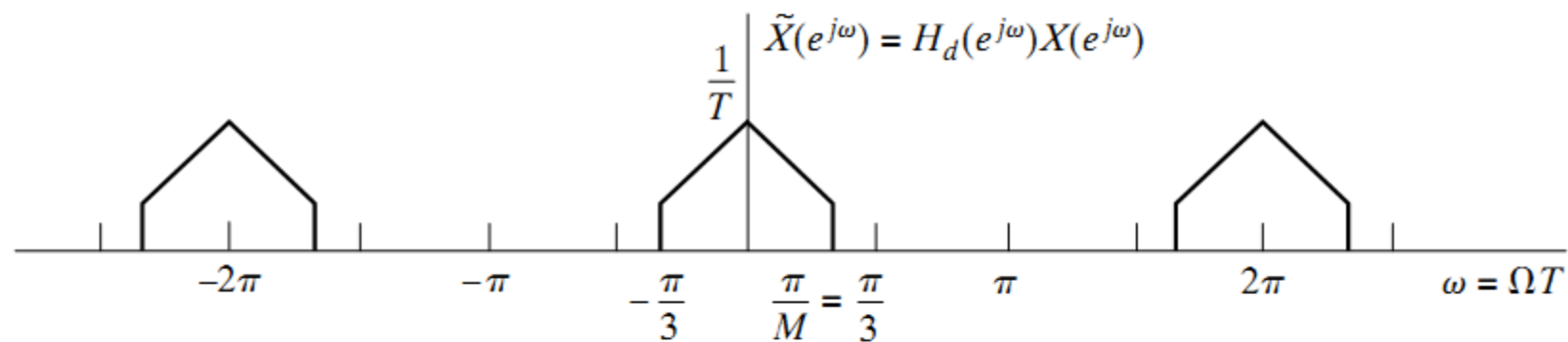
Downsampling with prefiltering to avoid aliasing (decimation)

- From the above, the DTFT of the down-sampled signal is the **superposition of M shifted/scaled versions** of the **DTFT of the original signal**.
- **To avoid aliasing, we need $|\omega_N| < \pi/M$** , where w_N is the highest frequency of the discrete-time signal $x[n]$.
- Hence, similar to the continuous-time sampling, **downsampling is often accompanied with a pre-low-pass filtering**, and a **low-pass filter followed by down-sampling** is called a **decimator**, and termed the process as **decimation**.

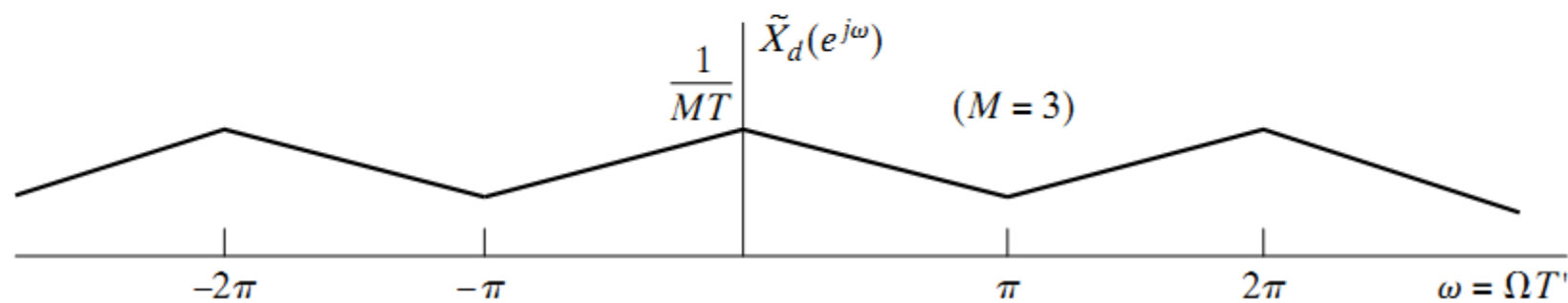




(d)



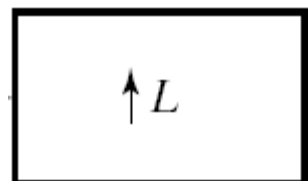
(e)



(f)

Up-sampling (or Expansion)

- Upsampling; sampling rate expander.



A block diagram of an up-sampler by factor L. It consists of a rectangular box with an upward-pointing arrow and the letter L inside. To the right of the box is a colon followed by the mathematical expression for the upsampled signal.

$$x_e[n] = \begin{cases} x[n/L], & n = 0, \pm L, \pm 2L, \dots \\ 0, & \text{otherwise} \end{cases}$$

or equivalently,

$$x_e[n] = \sum_{k=-\infty}^{\infty} x[k] \delta[n - kL]$$

Up-sampling example

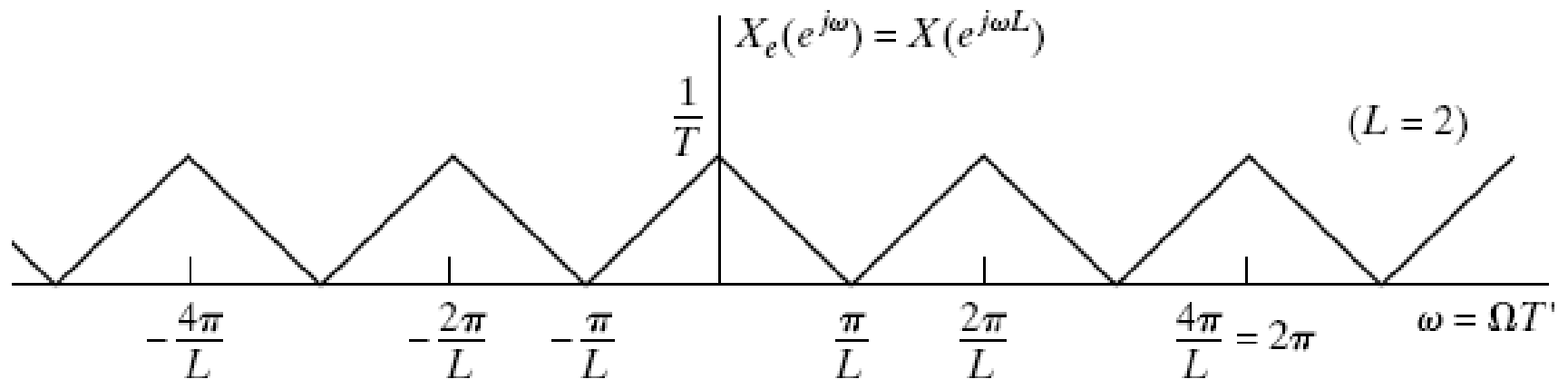
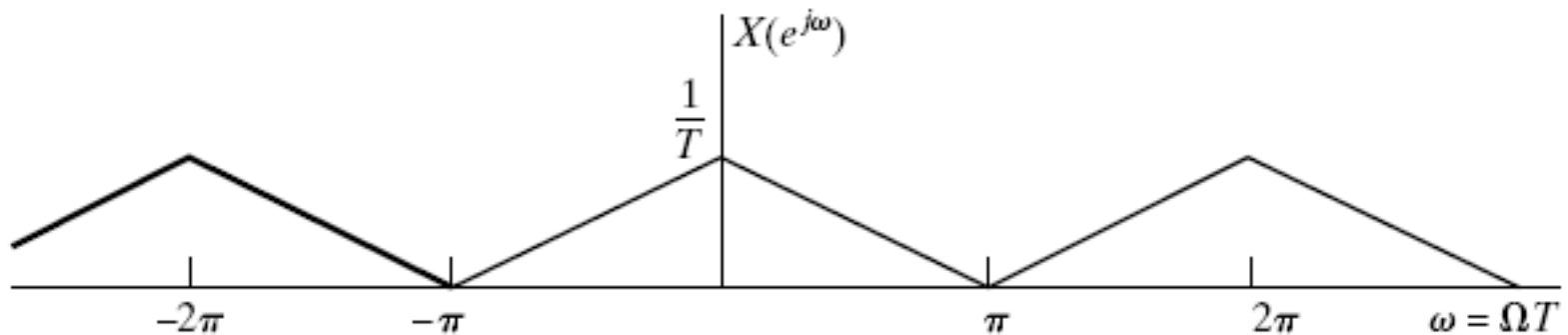
- $x = [1\ 2\ 3\ 4];$
- $y = \textit{upsample}(x, 3);$
- Display x, y
 - $x = 1\ 2\ 3\ 4$
 - $y = 1\ 0\ 0\ 2\ 0\ 0\ 3\ 0\ 0\ 4\ 0\ 0$

Up-sampling (frequency domain)

- In frequency domain:

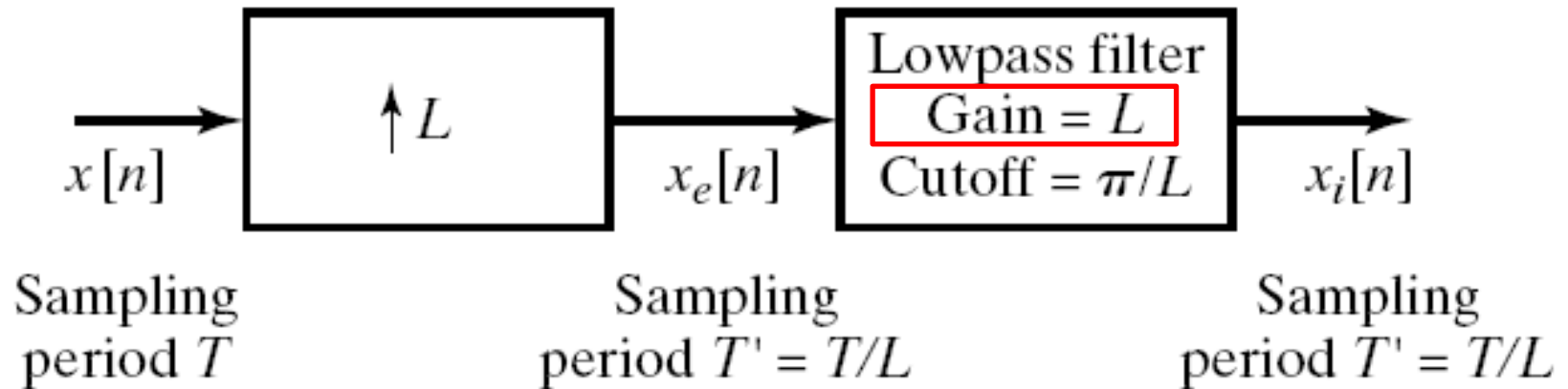
$$\begin{aligned} X_e(e^{j\omega}) &= \sum_{n=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} (x[k]\delta[n - kL])e^{-j\omega n} \\ &= \sum_{k=-\infty}^{\infty} (x[k]e^{-j\omega Lk}) \\ &= X(e^{j\omega L}) \end{aligned}$$

Example of up-sampling



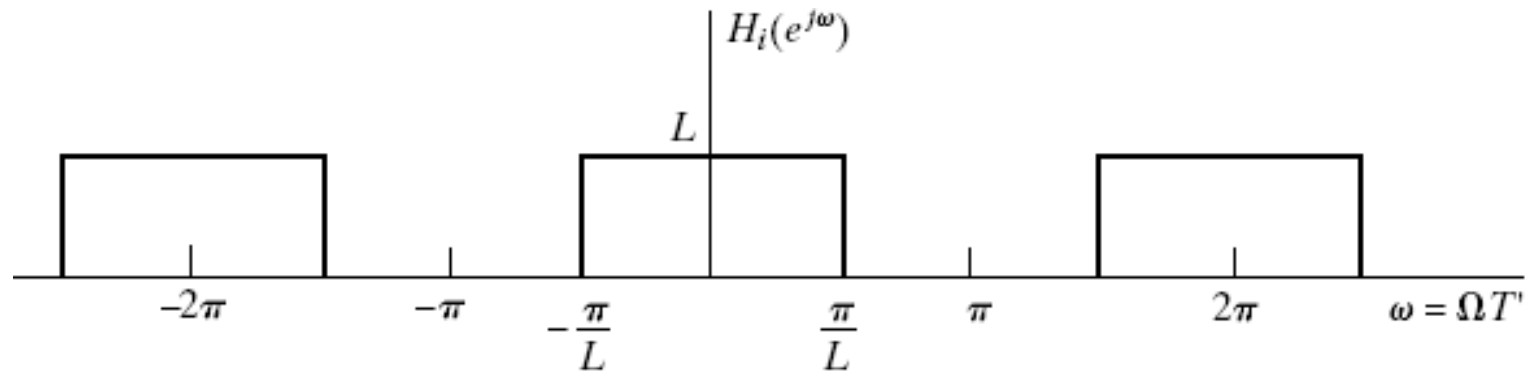
Upsampling in the DTFT domain

Up-sampling with post low-pass filtering

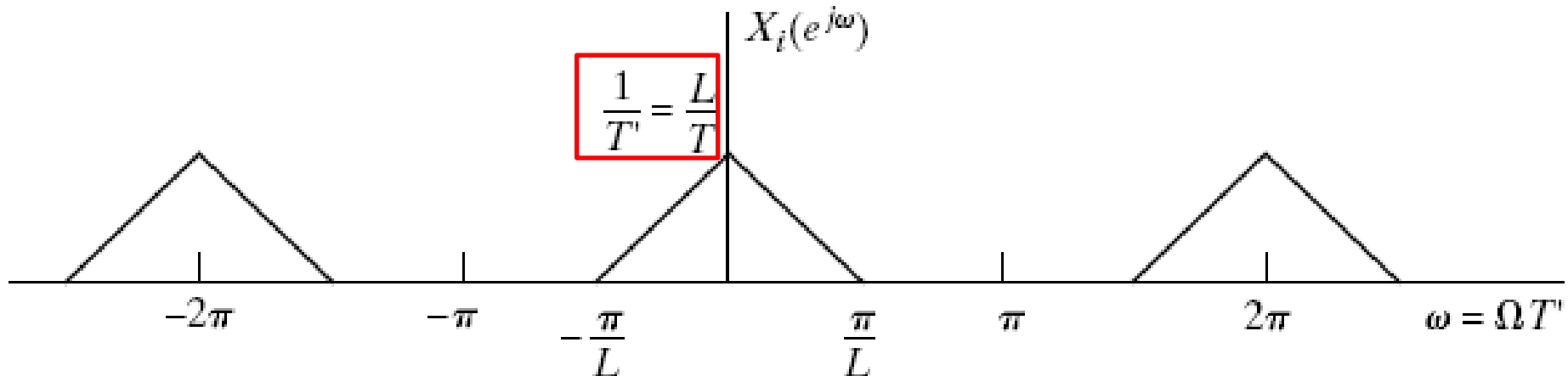


- There is **no information loss in up-sampling**. Thus the original signal can be reconstructed **(by filtering)**.
- Upsampling is often accompanied with a low-pass filter with cutoff frequency π/L and gain L , to reconstruct the sequence.
- An **up-sampling filter followed by low-pass filter** is called an **interpolator**, and the whole process is called **interpolation**.

Example of up-sampling followed by low-pass filtering



Applying low-pass filtering for signal in the previous slide



Interpolation (in time domain)

- If we choose an ideal lowpass filter with cutoff frequency π/L and gain L , its impulse response is
 - Hence

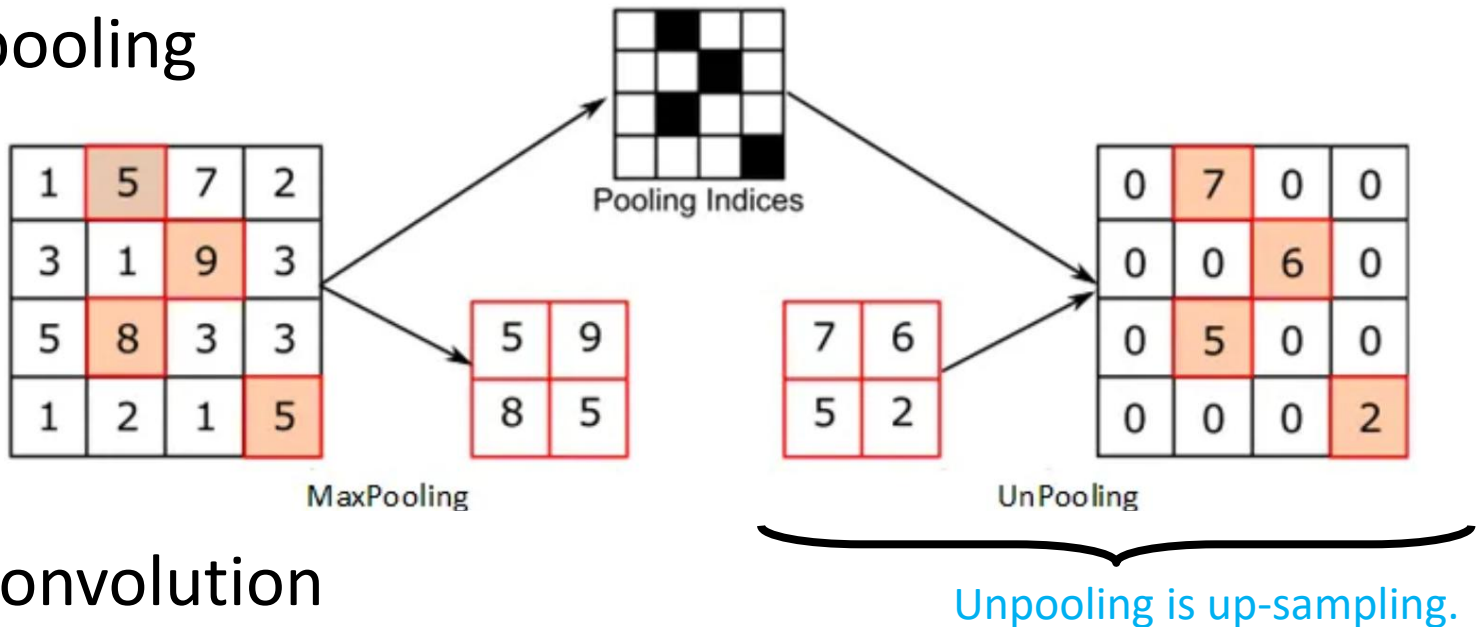
$$h_i[n] = \frac{\sin(\pi n / L)}{\pi n / L}$$

$$\begin{aligned} x_i[n] &= x_e[n] * h_i[n] = \left(\sum_{k=-\infty}^{\infty} x[k] \delta[n - kL] \right) * h_i[n] \\ &= \sum_{k=-\infty}^{\infty} x[k] \frac{\sin[\pi(n - kL) / L]}{\pi(n - kL) / L} \end{aligned}$$

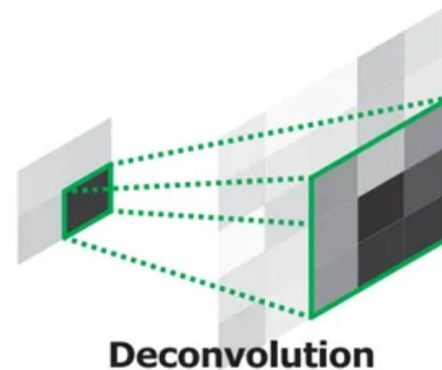
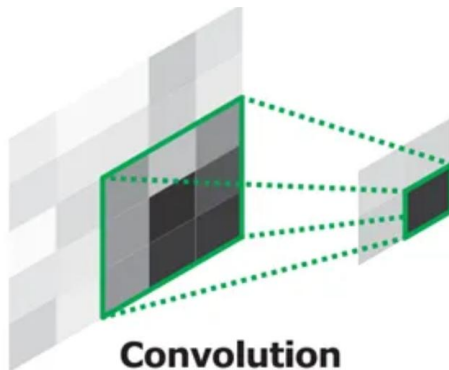
An interpolation of the discrete sequence $x[k]$

Analogous to Unpooling and Deconvolution in CNN

- Unpooling

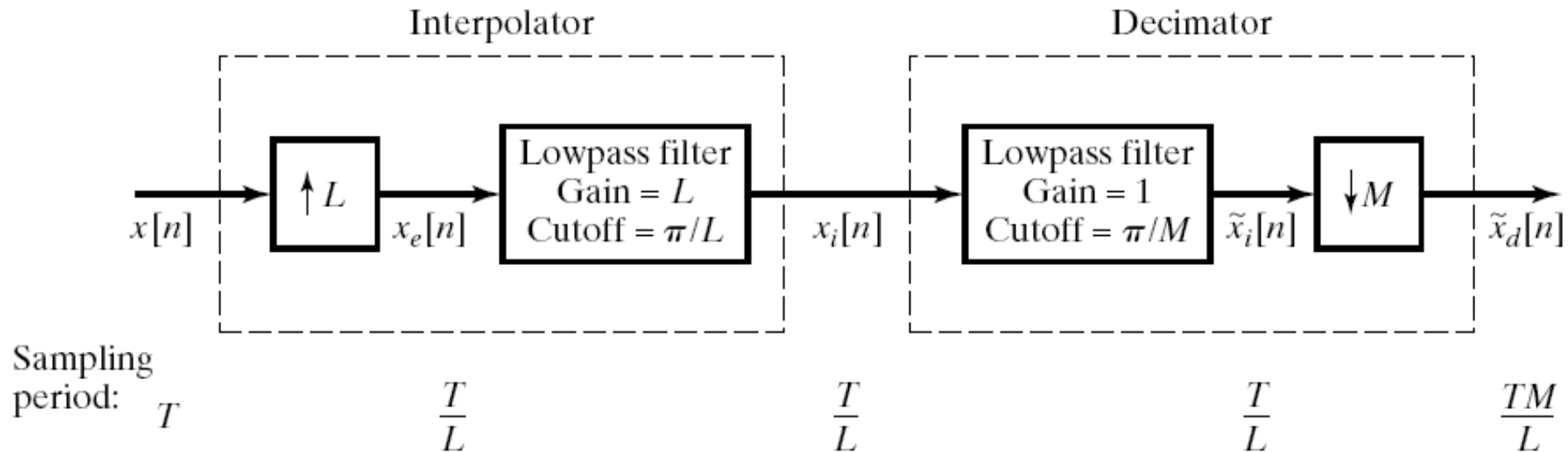


- Deconvolution



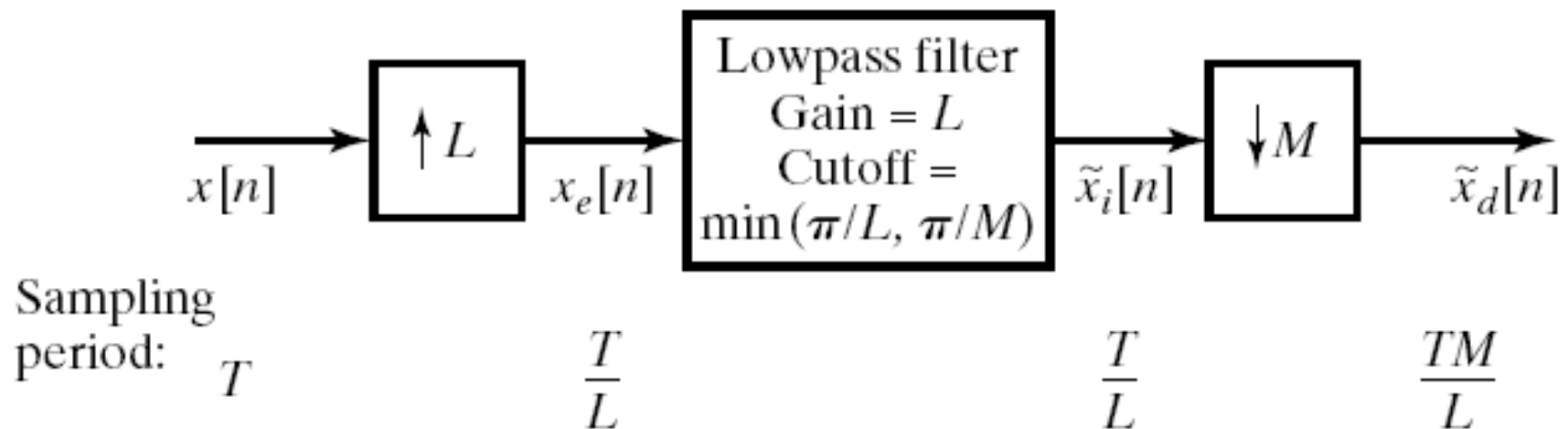
A deconvolution layer can be implemented by using **up-sampling + convolution** (FIR filter)

Sampling rate conversion by a non-integer rational factor



- By combining the decimation and interpolation, we can change the sampling rate of a sequence.
 - Changing the sampling rate by a non-integer factor $T' = TM/L$.
 - Eg., $L = 100$ and $M = 101$, then $T' = 1.01T$.

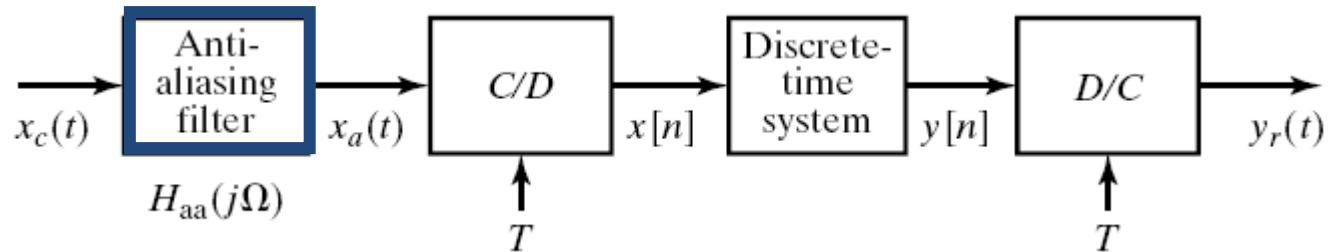
Changing the Sampling rate using discrete-time processing



- Since the interpolation and decimation filters are in cascade, they can be combined as one.

Anti-aliasing for Sampling Analog Signals

- **Pre-filtering to avoid aliasing:** When sampling a continuous-time signal, we often need to apply an anti-aliasing filter:

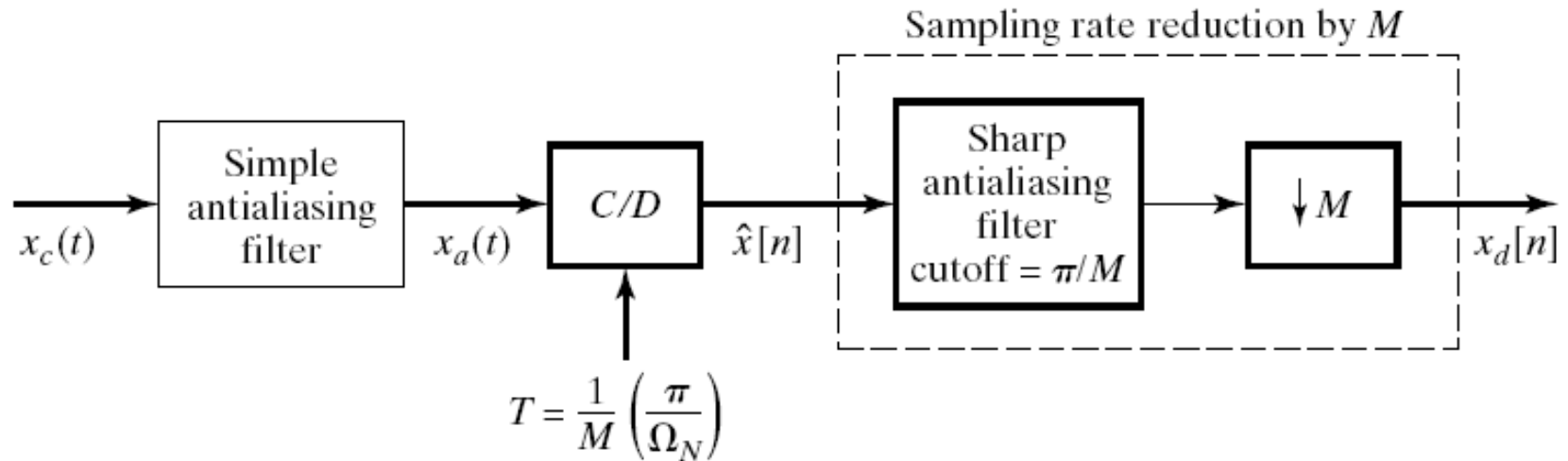


- It is generally desirable to minimize the sampling rate.
Eg., in speech signals, we would need only the low-frequency band up to about 3-4k Hz.
- Even though the speech signal may have significant frequency content in the 4k to 20k Hz range, wideband additive noise may also occur in these higher frequency range. These noise components would be aliased into the low frequency band.

Over-sampled A/D conversion

- The anti-aliasing filter above is an analog filter.
 - However, in applications involving powerful but inexpensive digital processors, these continuous-time filters may account for a major part of the cost.
- If the required highest frequency of the analog signal is Ω_N , we can first use a very simple analog anti-aliasing filter having a gradual cutoff with significant attenuation at $M\Omega_N$.
- Next, implement the continuous-to-discrete (C/D) conversion at the sampling rate higher than $2M\Omega_N$. (over sampling M times).
- After that, applying down-sampling by a factor of M that includes sharp anti-aliasing filtering realized in discrete time.

Illustration of over-sampled A/D conversion



Apply **gradual anti-aliasing filter** (with **low-cost hardware**) in the CFT domain.

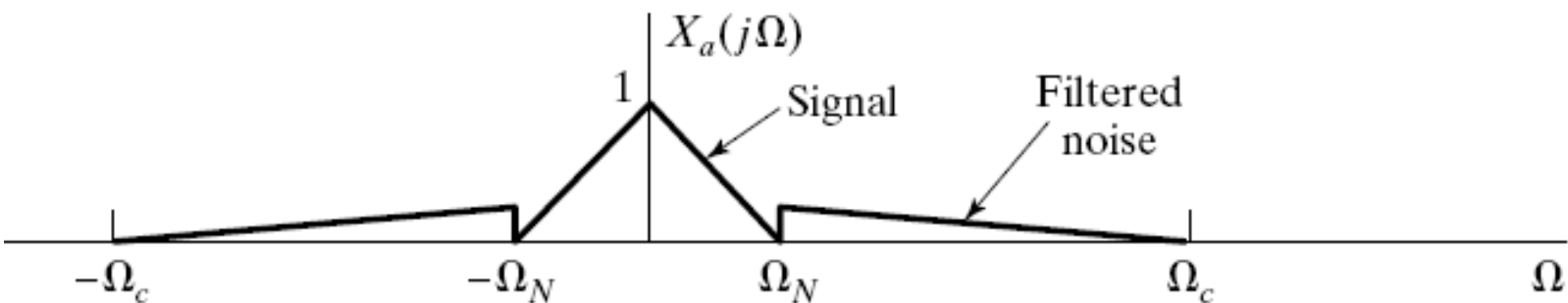
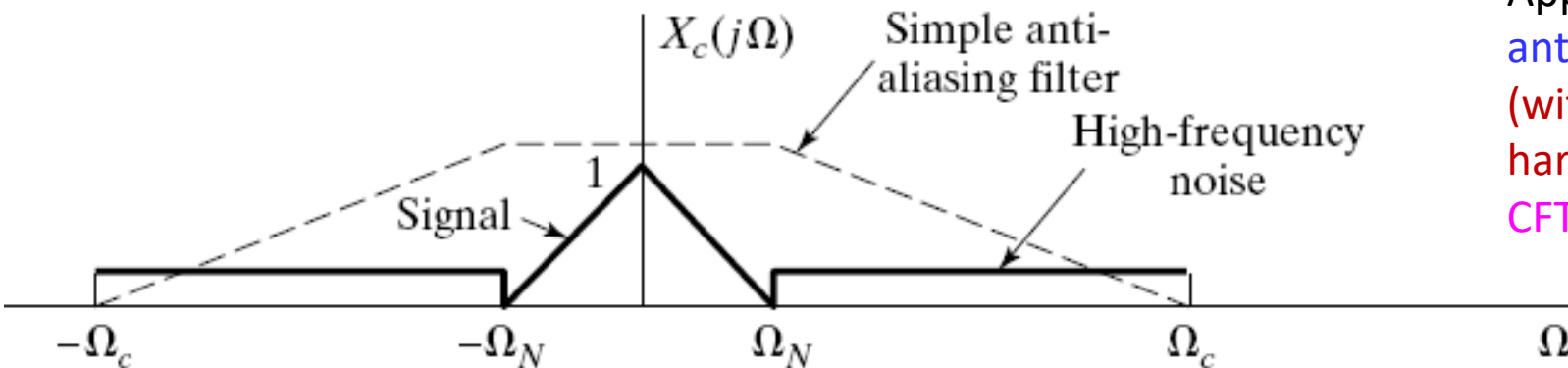
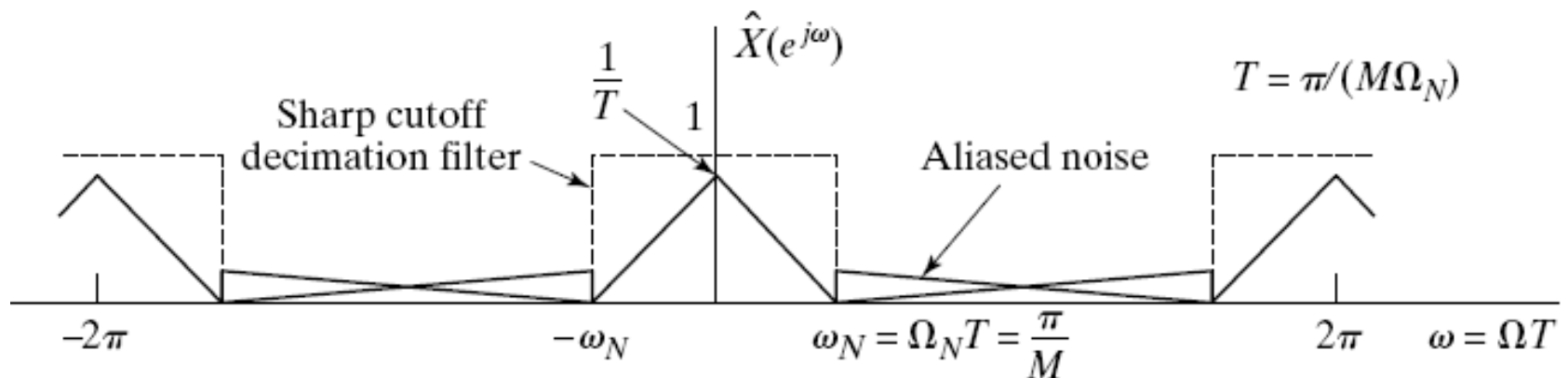


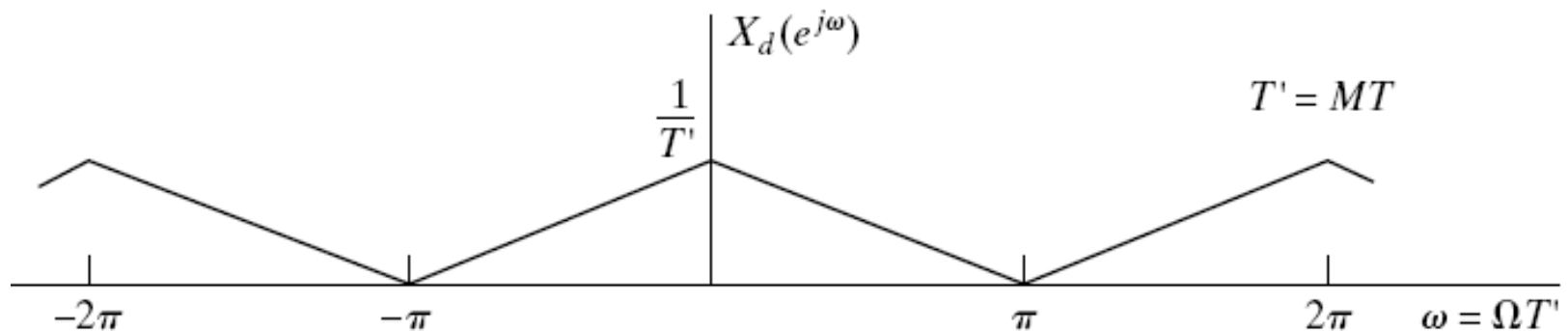
Illustration of over-sampled A/D conversion (Cont.)

Then, **over-sampling** M -times for the resulted analog signal (sampling rate $2M\Omega_N$), converting the signal to the **DTFT domain**.

After that, in DTFT domain, apply a **sharp anti-aliasing filter** (with **cutoff frequency** π/M in discrete-time)



Down-sampling the discrete-time signal by M .



Appendix

Proof of downsampling property

- When sampling a continuous-time signal $x_c(t)$, i.e., $x[n] = x_c(nT)$ we have the following equation in the frequency domain.

$$\text{DTFT} \quad X(e^{j\omega}) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c\left(j\left(\frac{\omega}{T} - \frac{2\pi k}{T}\right)\right) \quad \text{CFT}$$

- Hence, for the down-sampled signal $x_d[m] = x_c(mT')$, (where $T' = MT$), we have

$$\text{DTFT} \quad X_d(e^{j\omega}) = \frac{1}{T'} \sum_{r=-\infty}^{\infty} X_c\left(j\left(\frac{\omega}{T'} - \frac{2\pi r}{T'}\right)\right) \quad \text{CFT}$$

Proof of downsampling property (cont.)

- We are interested in the relation between the DTFTs of the originally sampled signal $X(e^{j\omega})$ and down-sampled signal $X_d(e^{j\omega})$.
- Let's represent r as $r = i + kM$, where $0 \leq i \leq M-1$, i.e., $r \equiv i \pmod{M}$. Then

$$\begin{aligned} X_d(e^{j\omega}) &= \frac{1}{MT} \sum_{r=-\infty}^{\infty} X_c \left(j \left(\frac{\omega}{MT} - \frac{2\pi r}{MT} \right) \right) \\ &= \frac{1}{M} \sum_{i=0}^{M-1} \underbrace{\frac{1}{T} \sum_{k=-\infty}^{\infty} X_c \left(j \left(\frac{\omega}{MT} - \frac{2\pi k}{T} - \frac{2\pi i}{MT} \right) \right)} \end{aligned}$$

This term is equivalent to

$$= \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c \left(j \left(\frac{\omega - 2\pi i}{MT} - \frac{2\pi k}{T} \right) \right) = X(e^{j(\omega - 2\pi i)/M})$$