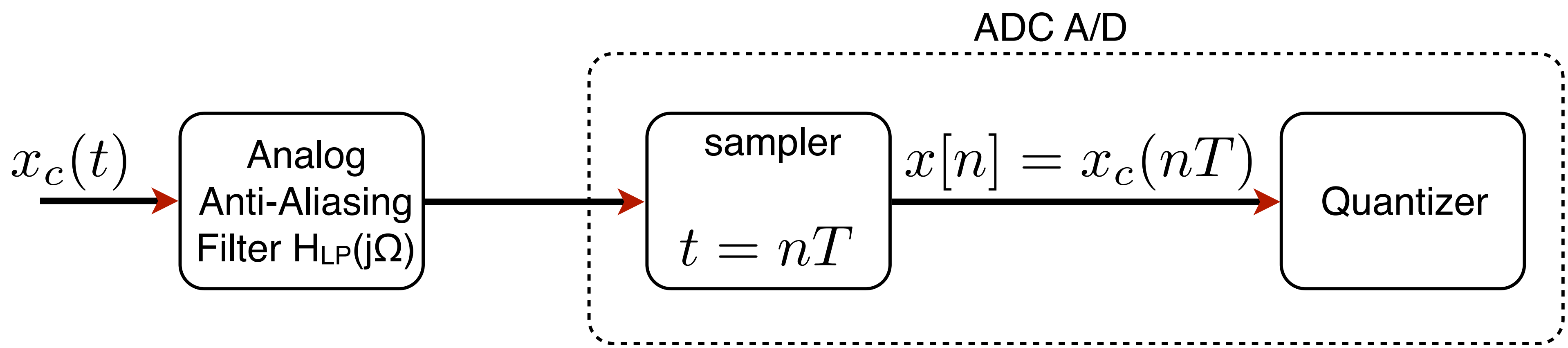
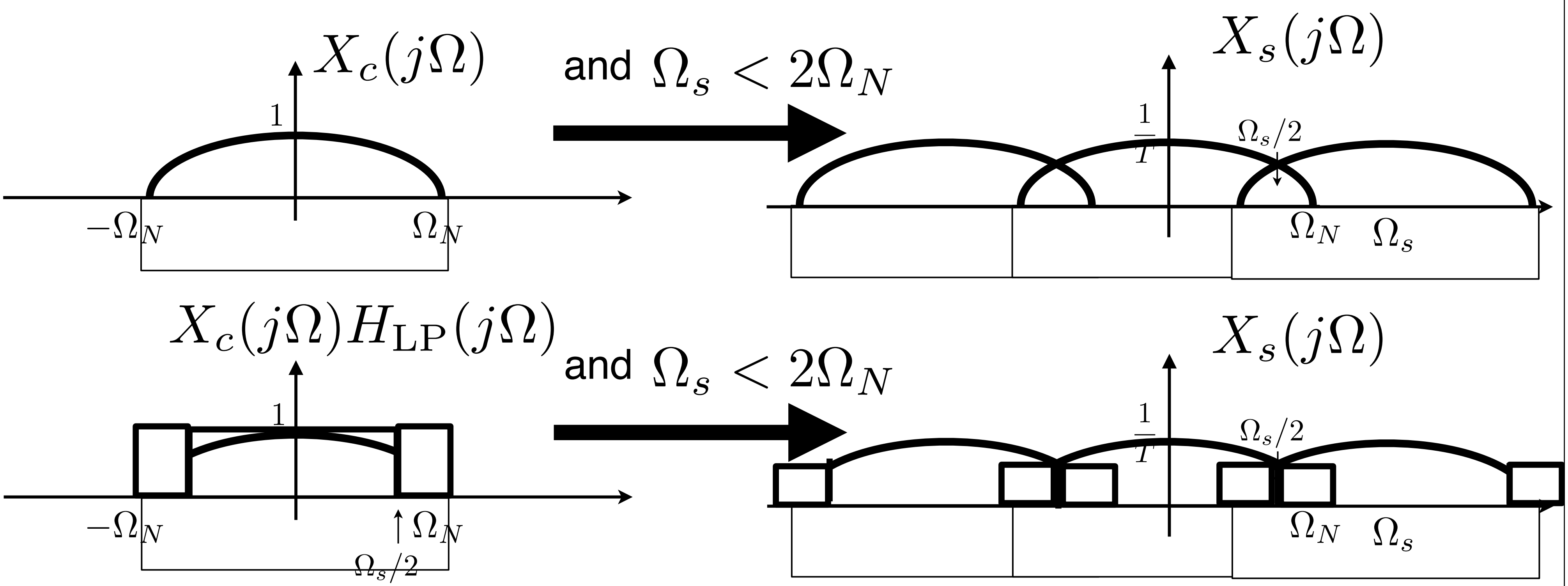
EE123
Digital Signal Processing

Lecture 19
Practical ADC/DAC

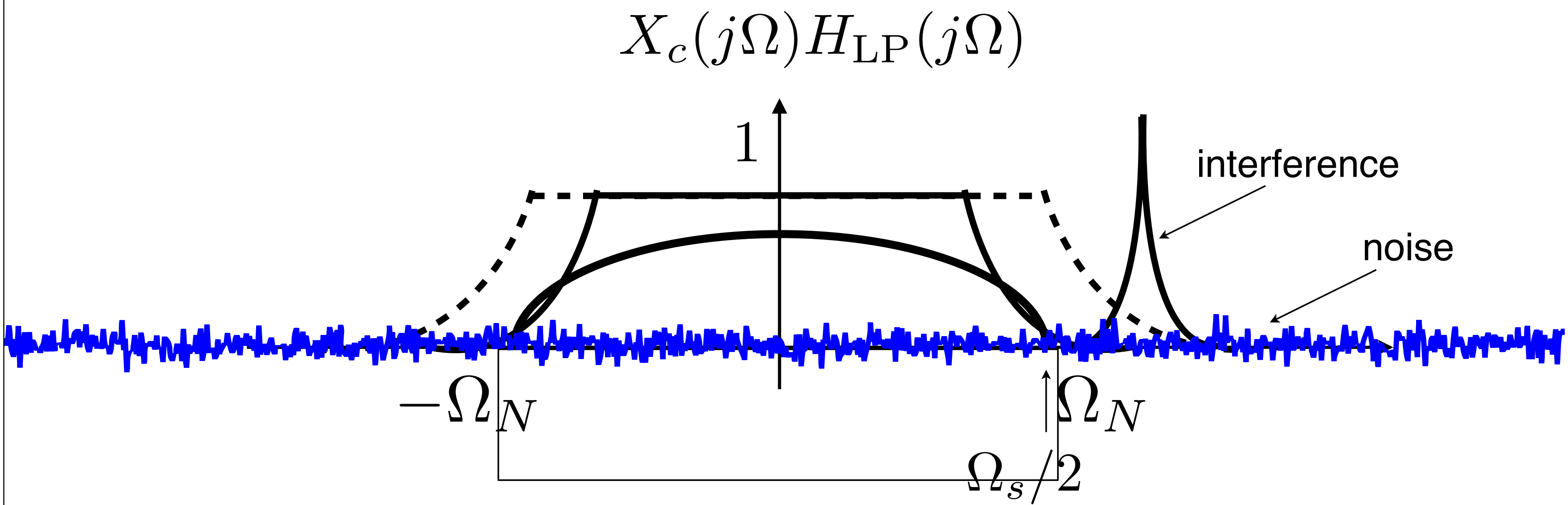
M. Lustig, EECS UC Berkeley

Ideal Anti-Aliasing



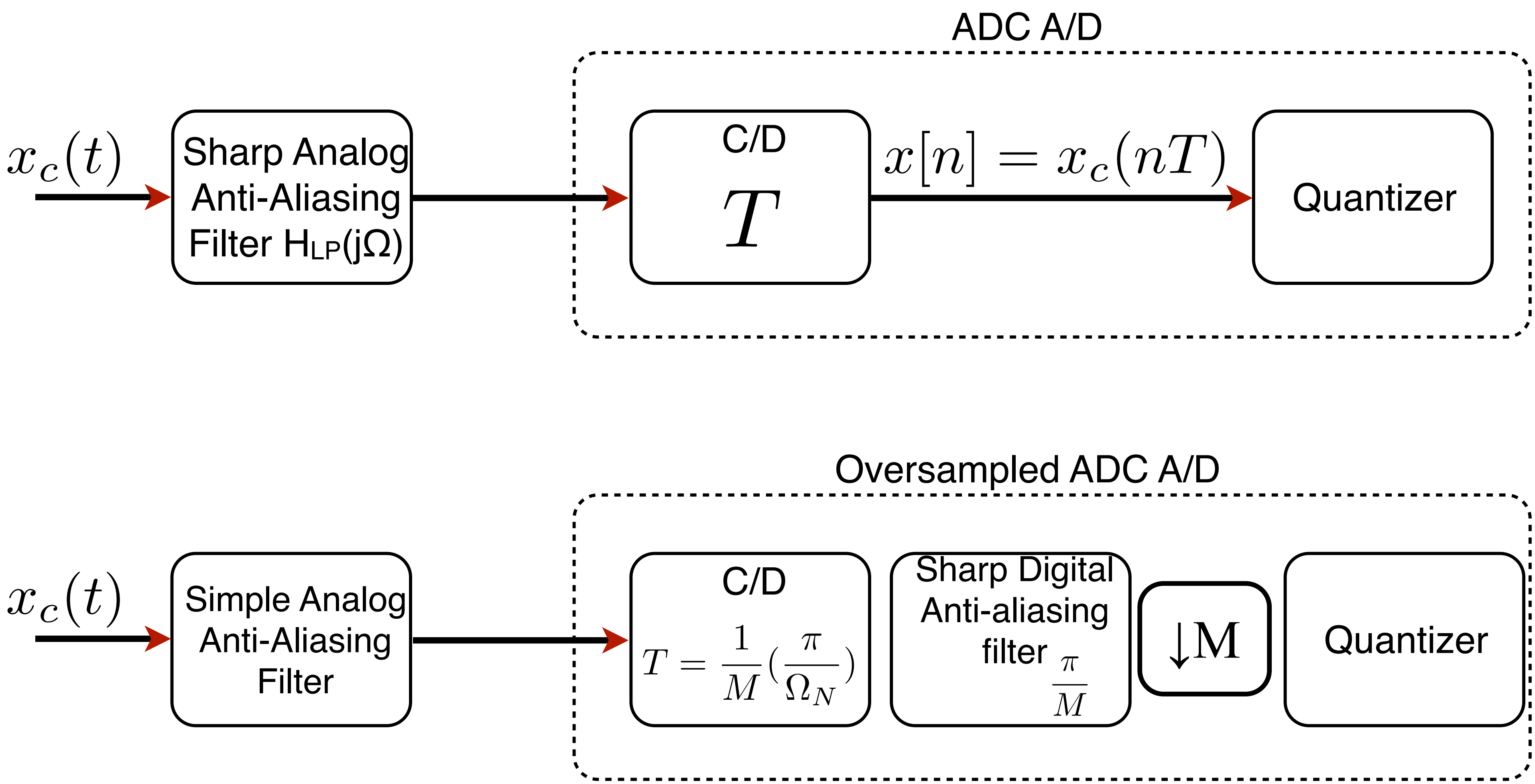


Non Ideal Anti-Aliasing

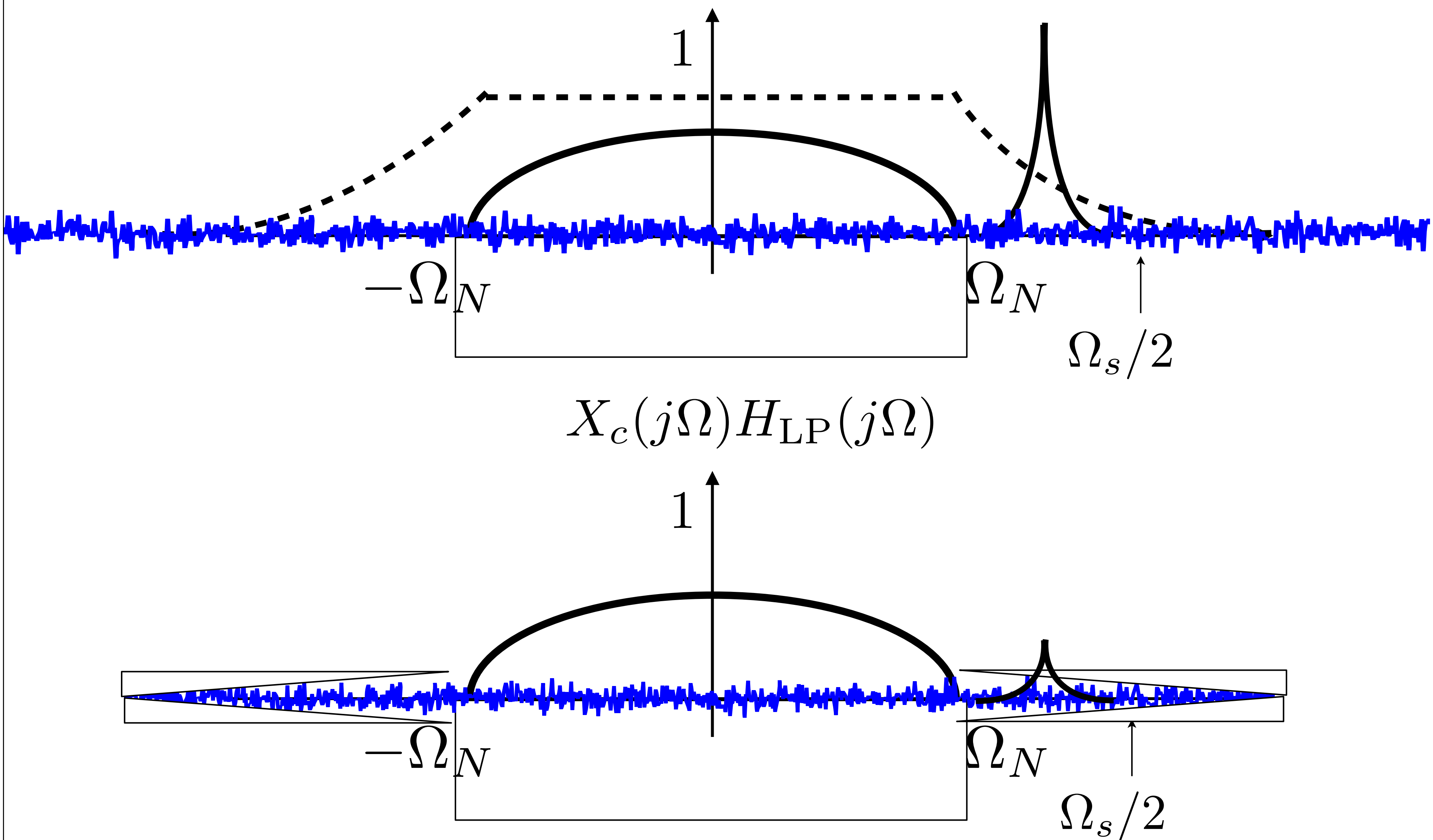


- Problem: Hard to implement sharp analog filter
- Tradeoff:
  - –Crop part of the signal
  - –Suffer from noise and interference (See lab II !)

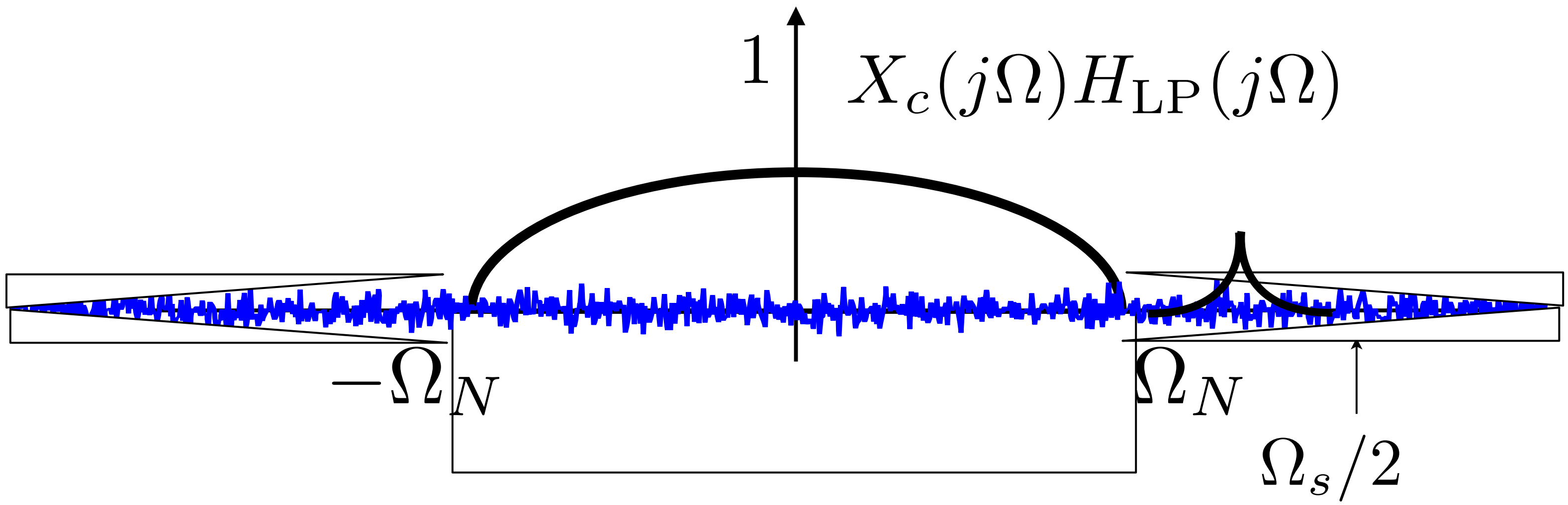
Oversampled ADC



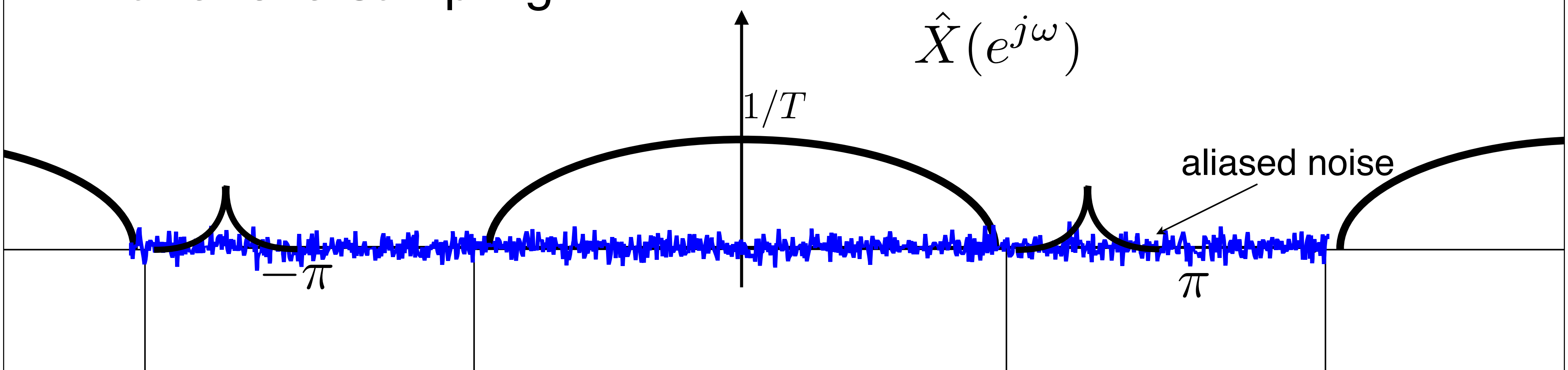
Oversampled ADC



Oversampled ADC

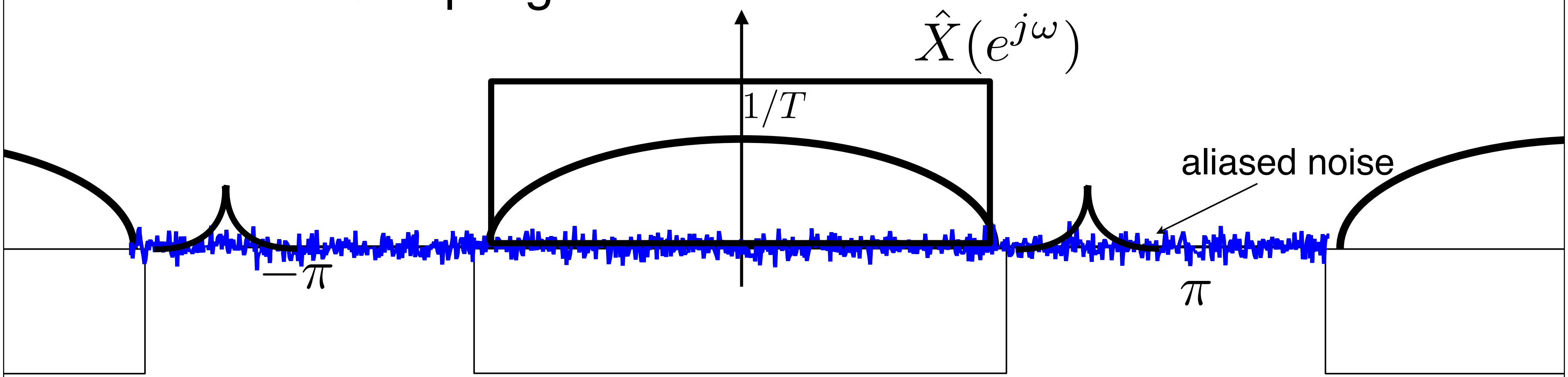


after oversampling x2

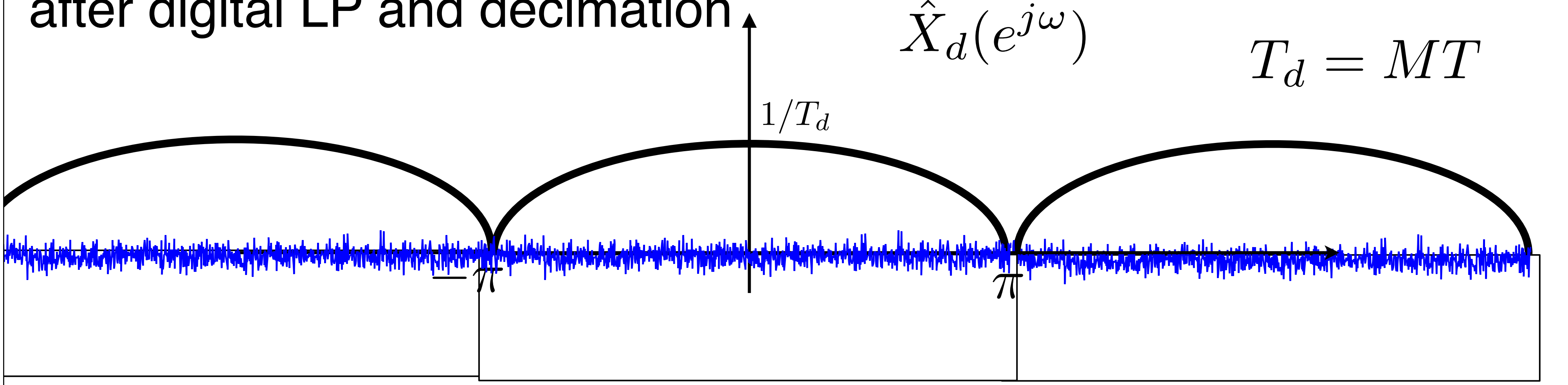


Oversampled ADC

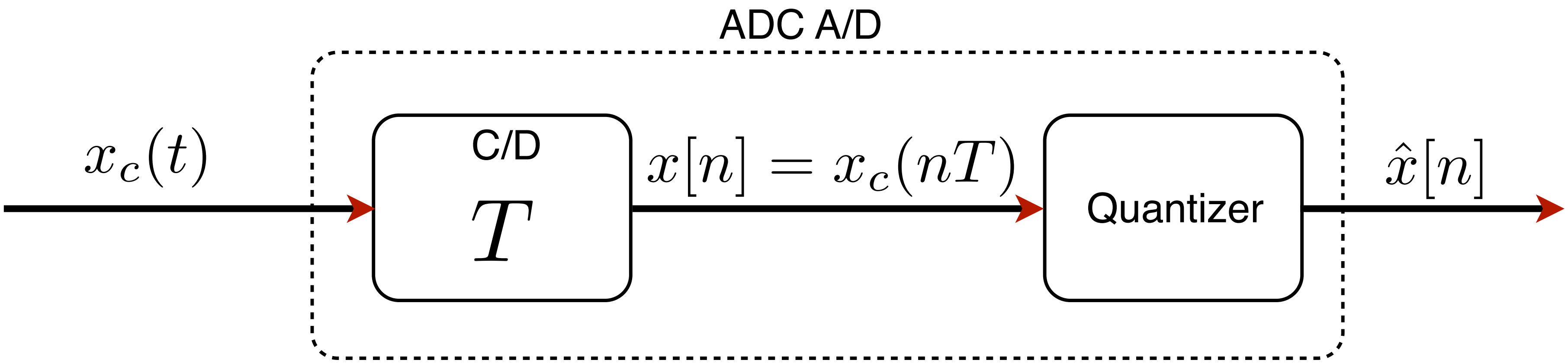
after oversampling x2

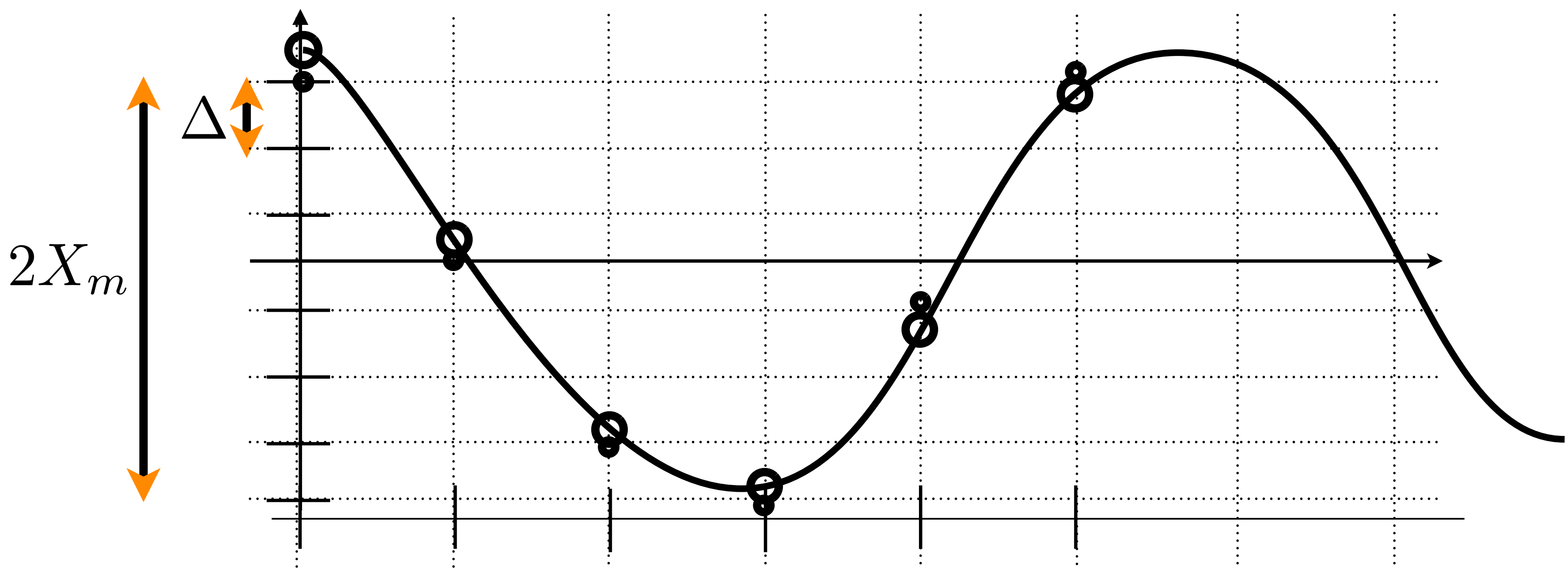


after digital LP and decimation



Sampling and Quantization



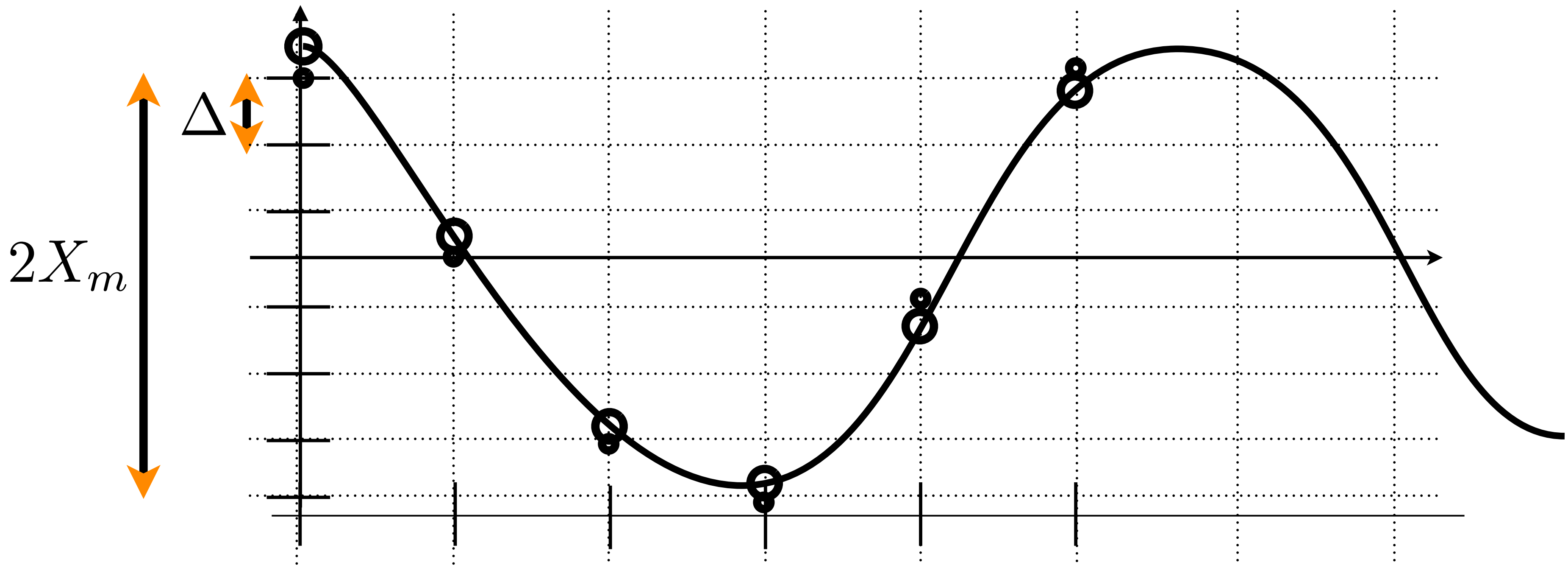


Sampling and Quantization

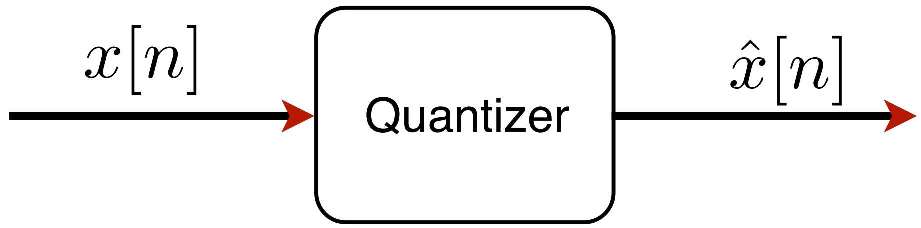
- for 2's complement with B+1 bits $-1 \leq \hat{x}_B[n] < 1$

$$\Delta = \frac{2X_m}{2^{B+1}} = \frac{X_m}{2^B}$$

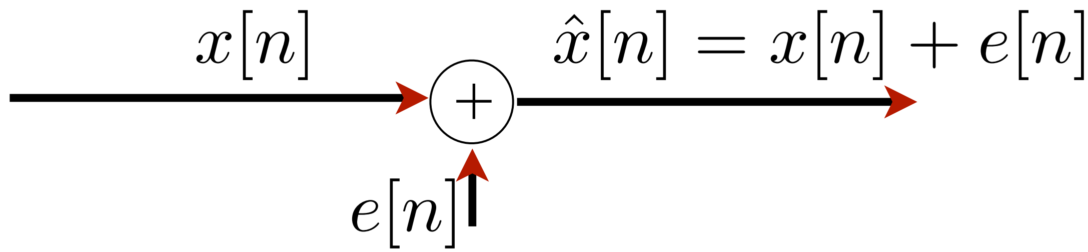
$$\hat{x}[n] = X_m \hat{x}_B[n]$$



Quantization Error



- Model quantization error as noise



- In that case:

$$-\Delta/2 \leq e[n] < \Delta/2$$

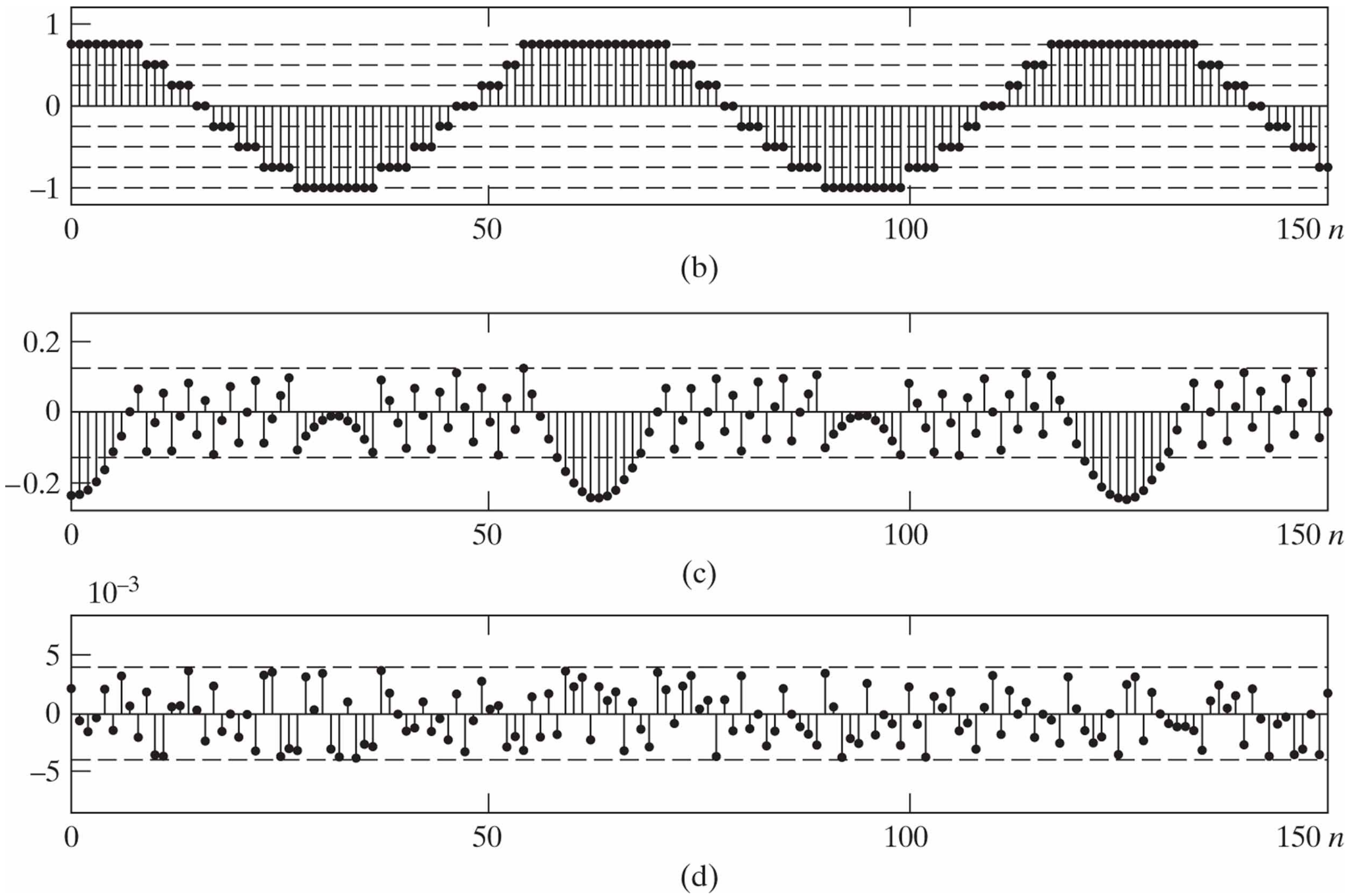
$$(-X_m - \Delta/2) < x[n] \leq (X_m - \Delta/2)$$

Noise Model for Quantization Error

- Assumptions:
  - Model $e[n]$ as a sample sequence of a stationary random process
  - $e[n]$ is not correlated with $x[n]$, *e.g.*, $\mathrm{E}\, e[n]\, x[n] = 0$
  - $e[n]$ not correlated with e[m], e.g., $\mathrm{E}\, e[n]\, x[m] = 0 \;\; | \; \mathrm{m \neq n}$ (white noise)
  - $e[n] \sim U[\text{-}\Delta/2, \Delta/2]$
- Result:
  - Variance is: $\sigma_e^2 = \dfrac{\Delta^2}{12}$, or $\sigma_e^2 = \dfrac{2^{-2B} X_m^2}{12}$ since $\Delta = 2^{-B} X_m$
  - Assumptions work well for signals that change rapidly, are not clipped and for small Δ

Quantization Noise

Figure 4.57 (*continued*) (b) Quantized samples of the cosine waveform in part (a) with a 3-bit quantizer. (c) Quantization error sequence for 3-bit quantization of the signal in (a). (d) Quantization error sequence for 8-bit quantization of the signal in (a).



SNR of Quantization Noise

- For uniform B+1 bits quantizer: $\sigma_e^2 = \frac{2^{-2B} X_m^2}{12}$

$$\begin{aligned} SNR_Q &= 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_e^2}\right) \\ &= 10\log_{10}\left(\frac{12\cdot 2^{2B}\sigma_x^2}{X_m^2}\right) \end{aligned}$$

$$\mathrm{SNR}_Q = 6.02B + 10.8 - 20\log_{10}\left(\frac{X_m}{\sigma_x}\right)$$

Quantizer range

rms of amp

SNR of Quantization Noise

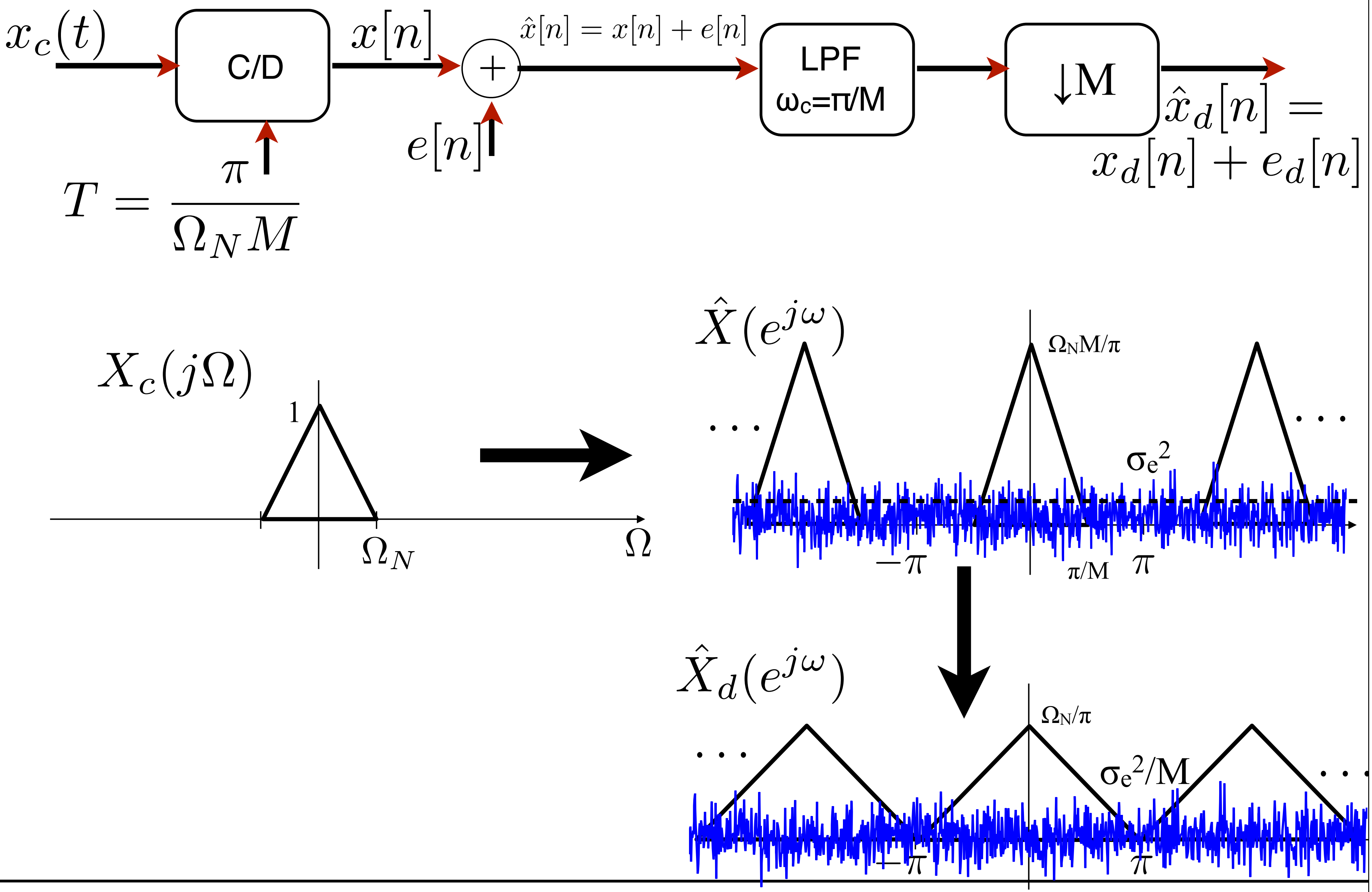
$$\mathrm{SNR}_Q = 6.02B + 10.8 - 20\log_{10}\left(\frac{X_m}{\sigma_x}\right)$$

Quantizer range

rms of amp

- Improvement of 6dB with every bit
- The range of the quantization must be adapted to the rms amplitude of the signal
  - Tradeoff between clipping and noise!
  - Often use pre-amp
  - Sometimes use analog auto gain controller (AGC)
  - If $\sigma_x = X_m/4$ then $\mathrm{SNR}_Q \approx 6B - 1.25$dB so SNR of 90-96 dB requires 16-bits (audio)

Quantization noise in Oversampled ADC



Quantization noise in Oversampled ADC

- Energy of $x_d[n]$ equals energy of $x[n]$
  - No filtering of signal!
- Noise var is reduced by factor of M

$$\mathrm{SNR}_Q = 6.02B + 10.8 - 20\log_{10}\left(\frac{X_m}{\sigma_x}\right) + 10\log_{10} M$$

- For doubling of M we get 3dB improvement, which is the same as 1/2 a bit of accuracy
  - With oversampling of 16 with 8bit ADC we get the same quantization noise as 10bit ADC!

Practical ADC (Ch. 4.8.4)

D.T

$$x[n] = x(t)|_{t=nT} \rightarrow \boxed{\text{sinc pulse generator}} \rightarrow x_r(t) = \sum_{n=-\infty}^{\infty} x[n]\text{sinc}\left(\frac{t-nT}{T}\right)$$

C.T

- Scaled train of sinc pulses
- Difficult to generate sinc ⇒ Too long!

Practical ADC

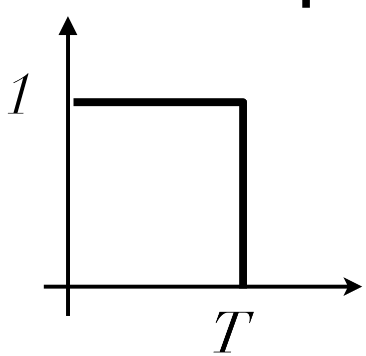
D.T

C.T analog processing

$x[n] = x(t)|_{t=nT} \rightarrow$ Interp. Filter $h_0(t) \Leftrightarrow H_0(j\Omega) \rightarrow$ Recon. Filter $h_r(t) \Leftrightarrow H_r(j\Omega) \rightarrow x_r(t)$

$= \sum x[n]h_0(t - nT)$

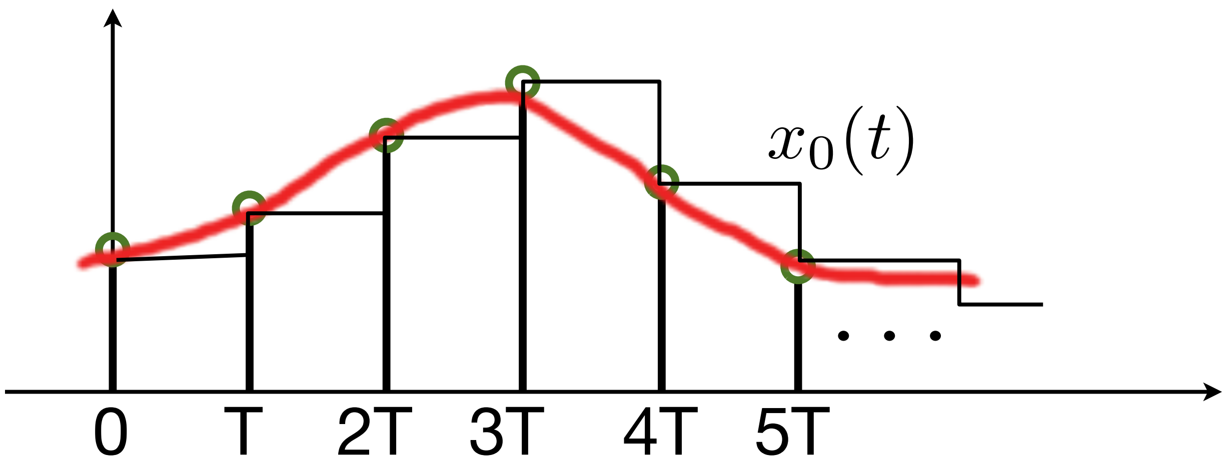
- $h_0(t)$ is finite length pulse ⇒ easy to implement
- For example: zero-order hold



$$H_0(j\Omega) = Te^{-j\Omega\frac{T}{2}}\operatorname{sinc}(\frac{\Omega}{\Omega_s})$$

Practical ADC

Zero-Order-Hold interpolation



$$x_0(t) = \sum_{n=-\infty}^{\infty} x[n]h_0(t-nT) = h_0(t) * x_s(t)$$

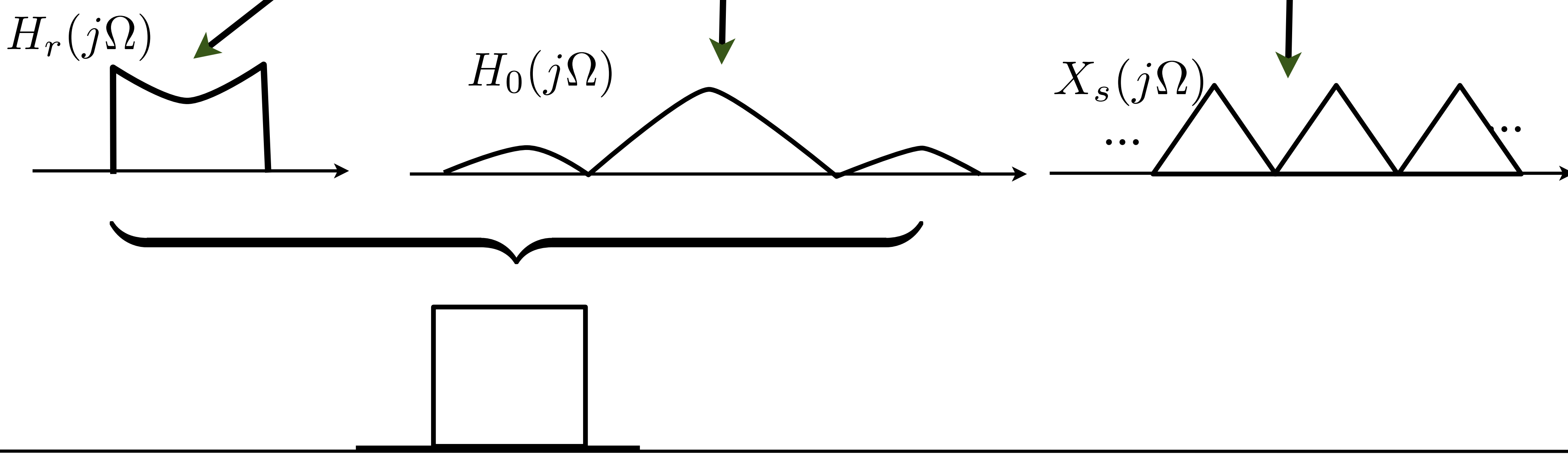
Taking a FT:

$$\begin{aligned} X_(j\Omega) &= H_0(j\Omega)X_s(j\Omega) \\ &= H_0(j\Omega)\frac{1}{T}\sum_{k=-\infty}^{\infty} X(j(\Omega - k\Omega_s)) \end{aligned}$$

Practical ADC

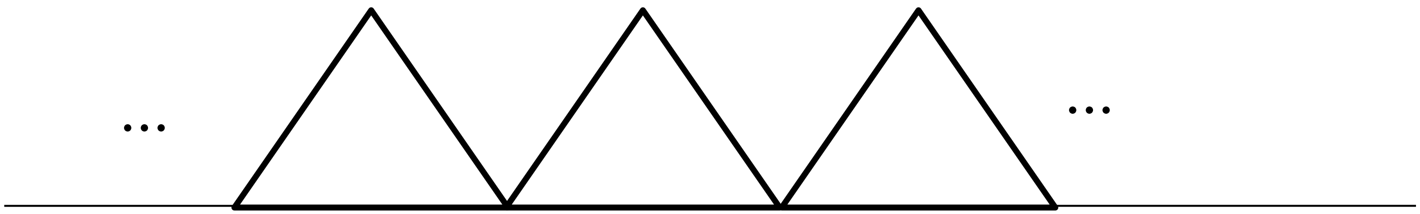
Output of the reconstruction filter:

$$
\begin{aligned}
X_r(j\Omega) &= H_r(j\Omega)\cdot H_0(j\Omega)\cdot X_s(j\Omega) \\
&= \underbrace{H_r(j\Omega)}_{\text{recon filter}}\cdot \underbrace{Te^{-j\Omega\frac{T}{2}}\operatorname{sinc}(\frac{\Omega}{\Omega_s})}_{\text{from zero-order hold}}\cdot \underbrace{\frac{1}{T}\sum_{k=-\infty}^{\infty} X(j(\Omega-k\Omega_s))}_{\text{Shifted copies from sampling}}
\end{aligned}
$$



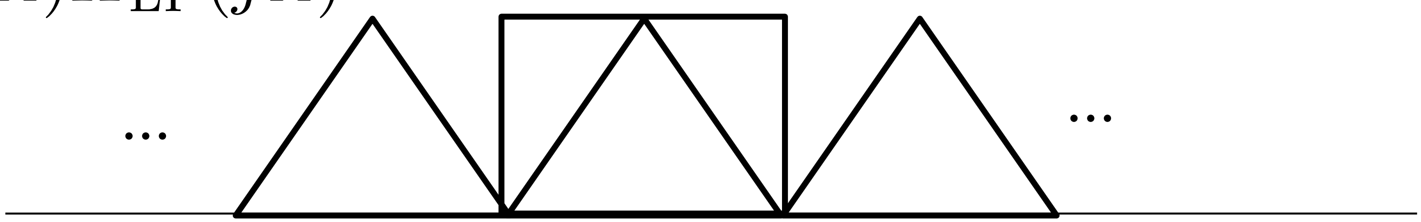
Practical ADC

$X_s(j\Omega)$



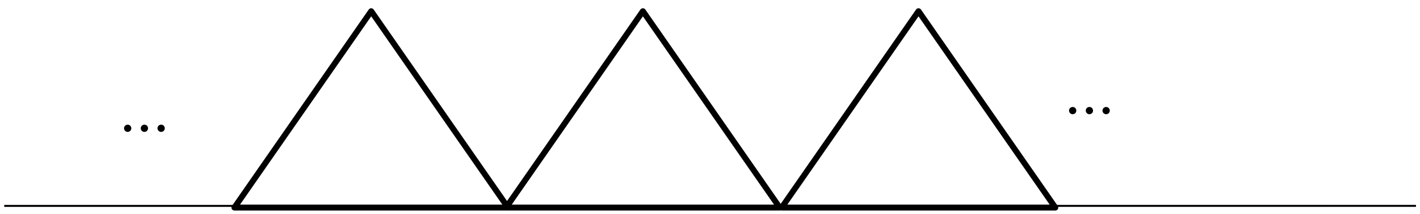
Ideally:

$X_s(j\Omega)H_{\mathrm{LP}}(j\Omega)$



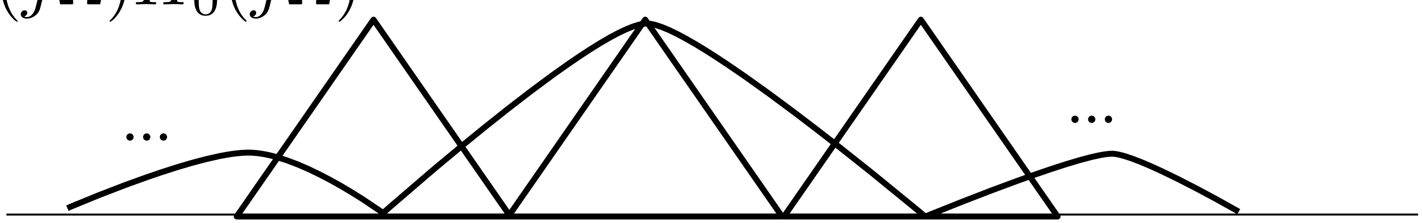
Practical ADC

$X_s(j\Omega)$



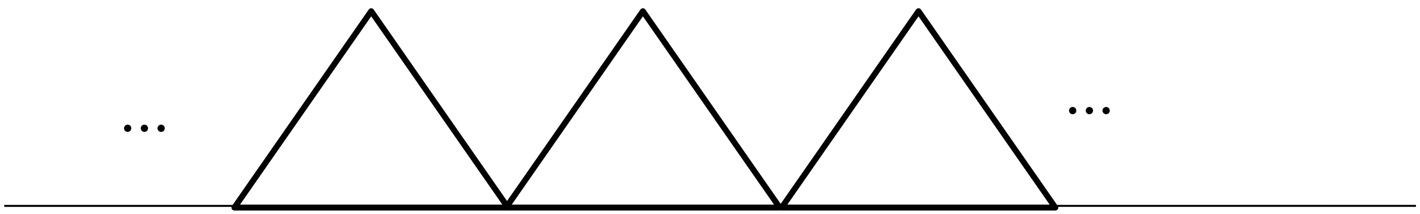
Practically:

$X_s(j\Omega)H_0(j\Omega)$



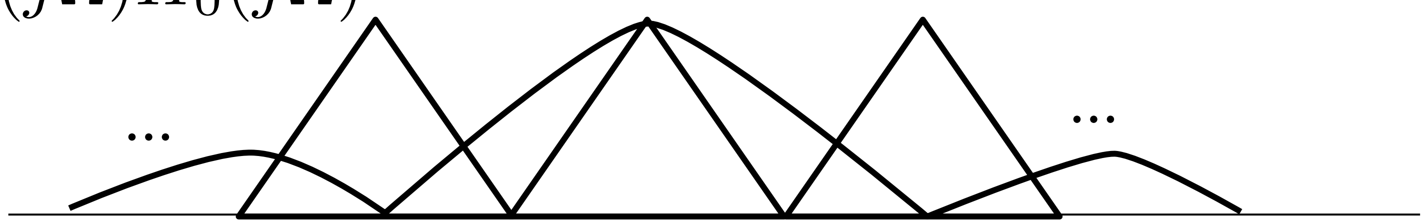
Practical ADC

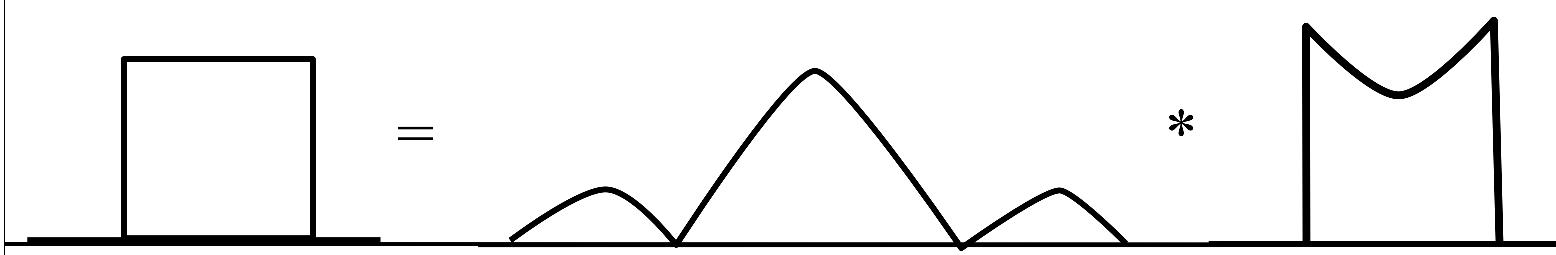
$X_s(j\Omega)$



Practically:

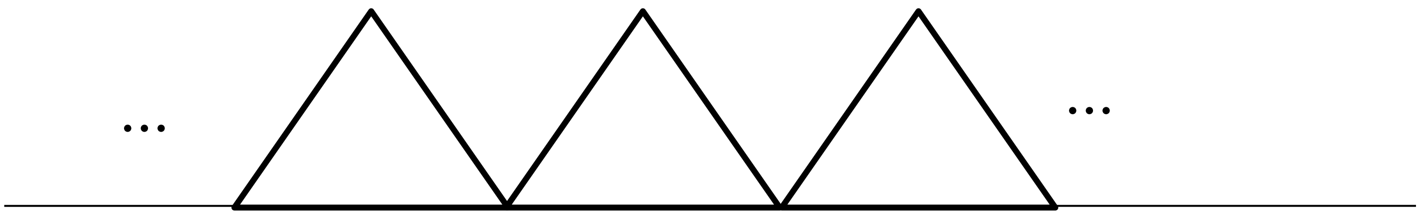
$X_s(j\Omega)H_0(j\Omega)$





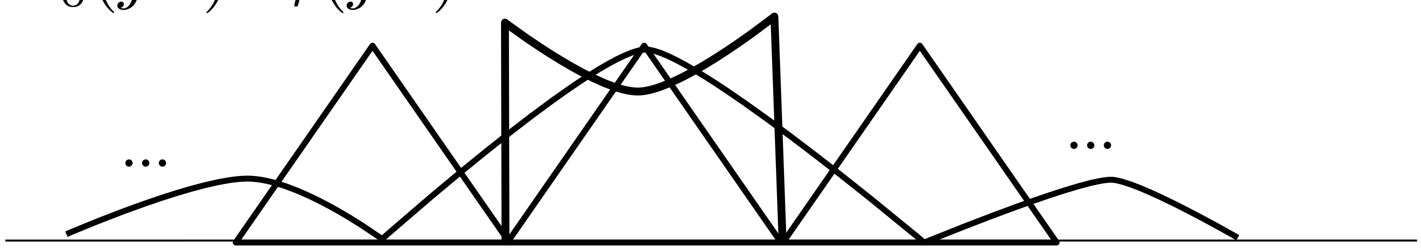
Practical ADC

$X_s(j\Omega)$

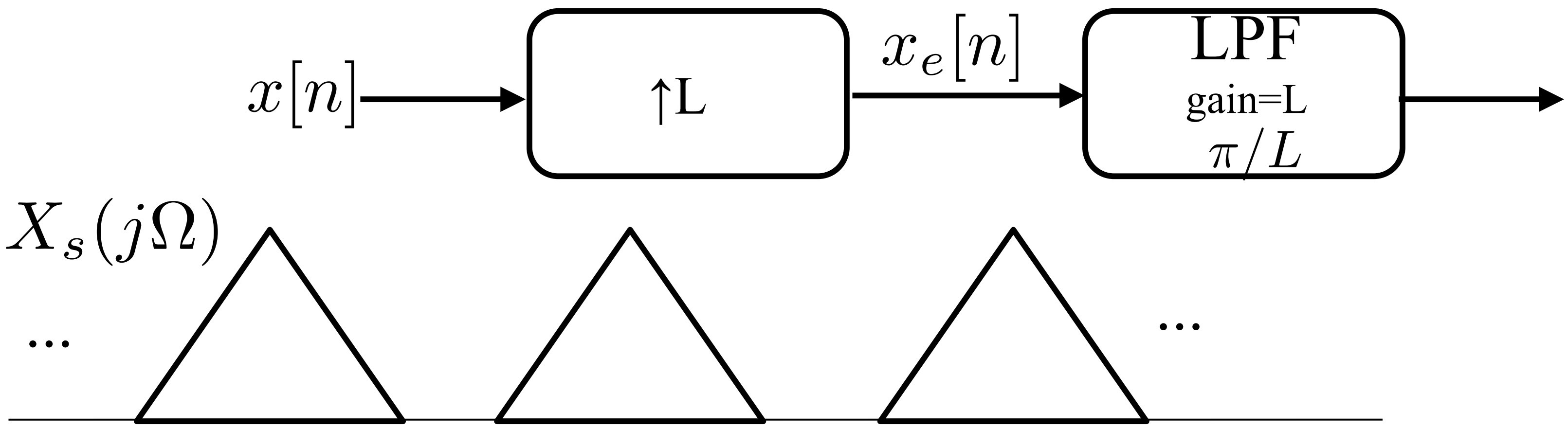


Practically:

$X_s(j\Omega)H_0(j\Omega)H_r(j\Omega)$

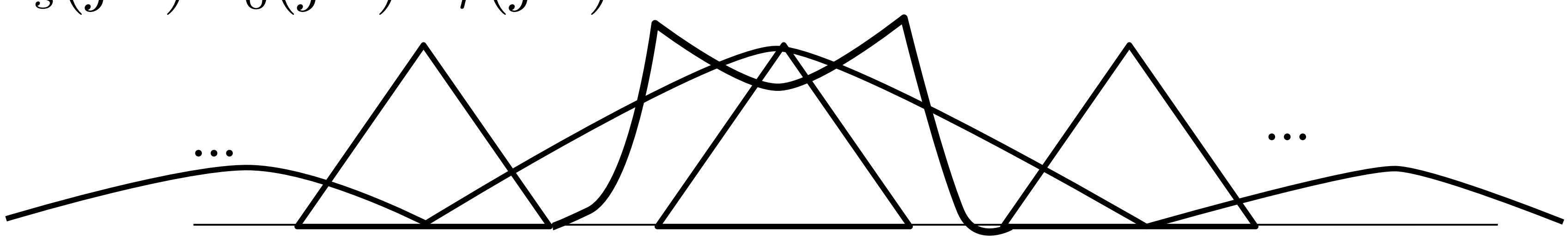


Easier Implementation with Digital upsampling



Practically:

$X_s(j\Omega)H_0(j\Omega)H_r(j\Omega)$



Easier Implementation with Digital upsampling

easier implementing
with analog components

Need analog components
made of low-loss
unobtainium transistors