DSP

Chapter-2 : Signals & Systems Review

Marc Moonen & Toon van Waterschoot
Dept. E.E./ESAT-STADIUS, KU Leuven
marc.moonen@esat.kuleuven.be
www.esat.kuleuven.be/stadius/

Chapter-2 : Signals & Systems Review

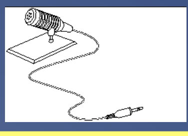
- **Discrete-Time/Digital Signals** (10 slides)
  Sampling, quantization, reconstruction
- **Discrete-Time Systems** (13 slides)
  LTI, impulse response, convolution, z-transform, frequency response, frequency spectrum, IIR/FIR
- **Discrete Fourier Transform** (4 slides)
  DFT-IDFT, FFT
- **Multi-Rate Systems** (11 slides)

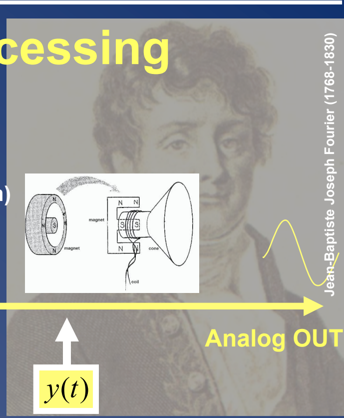
Discrete-Time/Digital Signals 1/10

Analog signal processing

Analog domain

(continuous-time domain)





Analog IN → Analog Signal Processing Circuit → Analog OUT

$u(t)$ $y(t)$

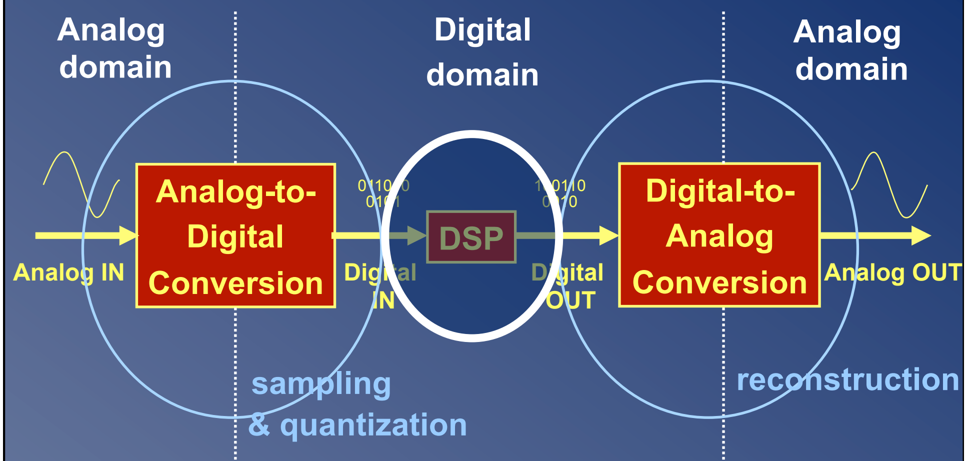
$$U(f)=\mathbb{F}\{u(t)\}=\int_{-\infty}^{+\infty}u(t).e^{-j2\pi.f.t}\,dt$$

$$Y(f)=\mathbb{F}\{y(t)\}=\int_{-\infty}^{+\infty}y(t).e^{-j2\pi.f.t}\,dt$$

(Fourier Transform / spectrum, where *f* = frequency)

Discrete-Time/Digital Signals 2/10

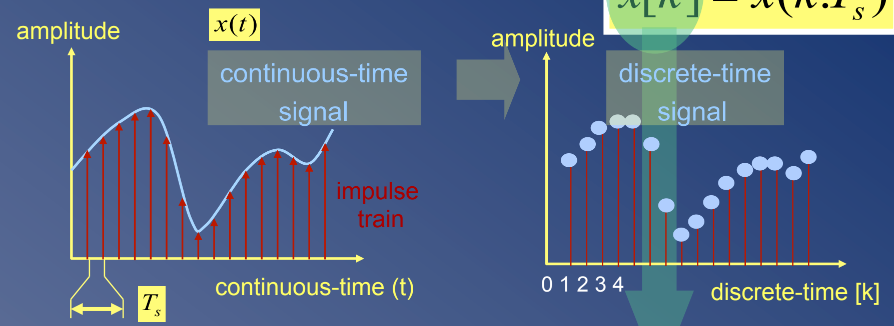
Digital signal processing



Discrete-Time/Digital Signals 3/10

1. Sampling

$$x[k] = x(k.T_s)$$



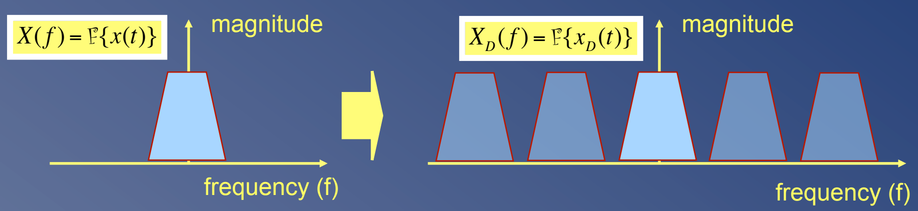
It will turn out (p.24-25) that a **spectrum** can be computed from x[k] (=discrete-time), which (remarkably) will be equal to the **spectrum** (=Fourier transform) of the (continuous-time) sequence of impulses……

$$x_D(t) = x(t).\sum_{k=-\infty}^{+\infty} \delta(t - k.T_s)$$

Discrete-Time/Digital Signals 4/10

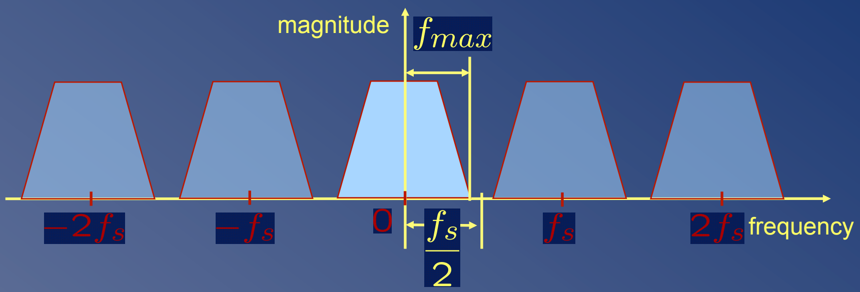
So what does this spectrum of $x_D(t)$ look like…

- **Spectrum replication**
  - Time domain: $$x_D(t) = x(t).\sum_{k=-\infty}^{+\infty} \delta(t - k.T_s)$$
  - Frequency domain: $$X_D(f) = \frac{1}{T_s}.\sum_{k=-\infty}^{+\infty} X(f - \frac{k}{T_s})$$



Discrete-Time/Digital Signals 5/10

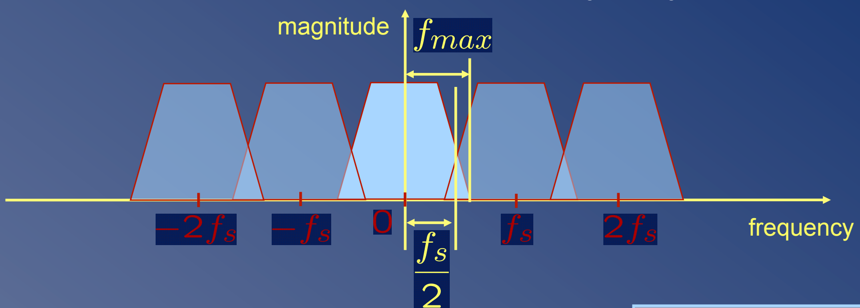
- **Sampling theorem**
  - Analog signal spectrum X(f) runs up to f_{max} Hz
  - Spectrum replicas are separated by $f_s = 1/T_s$ Hz



  - No spectral overlap if and only if **$f_s > 2.f_{max}$**

Discrete-Time/Digital Signals 6/10

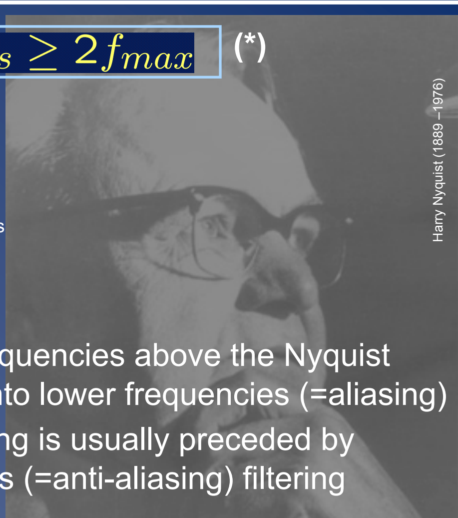
- **Sampling theorem**
  - Analog signal spectrum X(f) runs up to f_{max} Hz
  - Spectrum replicas are separated by $f_s = 1/T_s$ Hz



  - Spectral overlap (='folding', 'aliasing') if **$f_s < 2.f_{max}$**

Discrete-Time/Digital Signals 7/10

- **Sampling theorem** $f_s \geq 2f_{max}$ (*)
  - Terminology:
    - sampling frequency/rate f_s
    - Nyquist frequency $f_s/2$
    - sampling interval/period T_s
  - E.g. CD audio: f_s = 44,1 kHz
- **Anti-aliasing prefilters**
  - If $f_s < 2f_{max}$ then frequencies above the Nyquist frequency are 'folded' into lower frequencies (=aliasing)
  - To avoid aliasing, sampling is usually preceded by (analog-domain) low-pass (=anti-aliasing) filtering

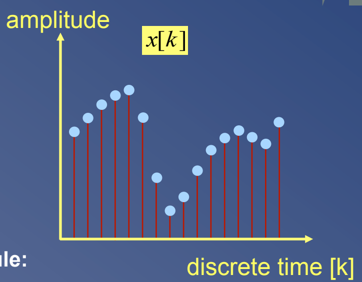

Harry Nyquist (1889 –1976)

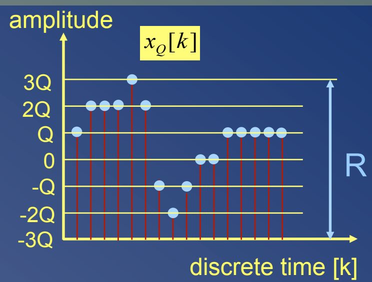
(*) An equivalent formulation is $f_s > f_{max}-(-f_{max}) = f_{max}-f_{min}$ = 'bandwidth'…will use this in p.36

Discrete-Time/Digital Signals 8/10

2. B-bit quantization

discrete-time signal → quantized discrete-time signal =discrete-amplitude&time signal =digital signal





6dB per bit rule:

$$\text{Number of bits } B = \log_2\left(\frac{\text{range } R}{\text{quantization step } Q}+1\right)$$

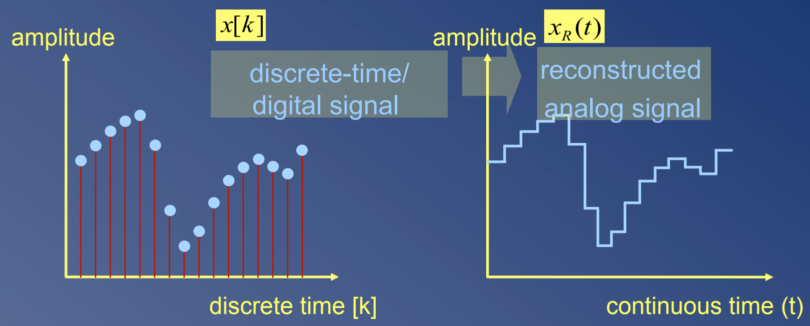
$$\text{Signal-to-QuantizationNoise-Ratio} = 20\log_{10}\left(\frac{\text{range } R}{\text{quantization step } Q}\right) = 6B \text{ dB}$$

Ex: CD audio = 16bits ~ 96dB SNR (LP's: 60dB SNR)

Discrete-Time/Digital Signals 9/10

3. Reconstruction

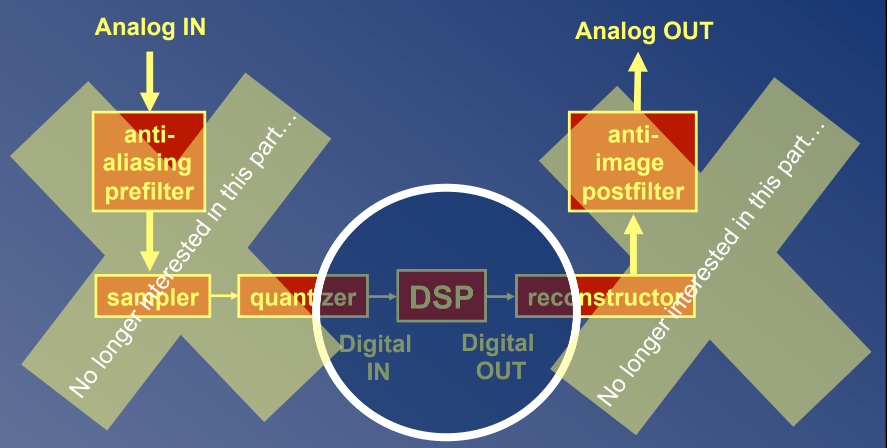
– Reconstruction = ‘fill the gaps’ between adjacent samples
– Example: staircase reconstructor



– In a practical realization $x_D(t)$ is generated first as an intermediate signal by means of a D-to-A & sampler, which is then followed by (analog domain) filtering (details omitted)

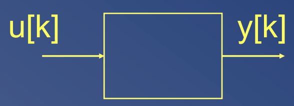
Discrete-Time/Digital Signals 10/10

- **Complete scheme is…**



Discrete-Time Systems 1/13

Discrete-time system is `sampled data' system



Input signal u[k] is a sequence of samples (=numbers)

..,u[-2],u[-1],u[0], u[1],u[2],…

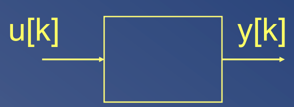
System then produces a sequence of output samples y[k]

..,y[-2],y[-1],y[0], y[1],y[2],…

Example: `DSP' block in previous slide

Discrete-Time Systems 2/13

Will consider linear time-invariant (LTI) systems



Linear :

input u1[k] -> output y1[k]

input u2[k] -> output y2[k]

hence a.u1[k]+b.u2[k]-> a.y1[k]+b.y2[k]

Time-invariant (shift-invariant)

input u[k] -> output y[k]

hence input u[k-T] -> output y[k-T]

Discrete-Time Systems 3/13

Will consider causal systems

iff for all input signals with u[k]=0,k<0 -> output y[k]=0,k<0

Impulse response

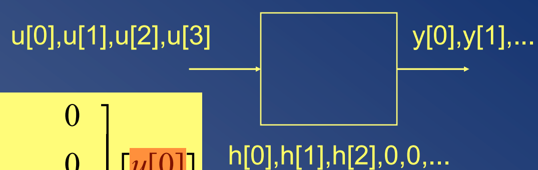
input …,0,0, 1 (K=0) ,0,0,0,…-> output …,0,0, h[0] (K=0) ,h[1],h[2],h[3],…

General input u[0],u[1],u[2],u[3]   (cfr. linearity & shift-invariance!)

$$\begin{bmatrix} y[0] \\ y[1] \\ y[2] \\ y[3] \\ y[4] \\ y[5] \end{bmatrix} = \begin{bmatrix} h[0] & 0 & 0 & 0 \\ h[1] & h[0] & 0 & 0 \\ h[2] & h[1] & h[0] & 0 \\ 0 & h[2] & h[1] & h[0] \\ 0 & 0 & h[2] & h[1] \\ 0 & 0 & 0 & h[2] \end{bmatrix} . \begin{bmatrix} u[0] \\ u[1] \\ u[2] \\ u[3] \end{bmatrix}$$

Discrete-Time Systems 4/13

Convolution



$$\begin{bmatrix} y[0] \\ y[1] \\ y[2] \\ y[3] \\ y[4] \\ y[5] \end{bmatrix} = \begin{bmatrix} h[0] & 0 & 0 & 0 \\ h[1] & h[0] & 0 & 0 \\ h[2] & h[1] & h[0] & 0 \\ 0 & h[2] & h[1] & h[0] \\ 0 & 0 & h[2] & h[1] \\ 0 & 0 & 0 & h[2] \end{bmatrix} . \begin{bmatrix} u[0] \\ u[1] \\ u[2] \\ u[3] \end{bmatrix}$$

$$y[k] = \sum_{\bar{k}} h[k-\bar{k}].u[\bar{k}] \stackrel{\Delta}{=} h[k] * u[k]$$

= `convolution sum '

(=more convenient than Toeplitz matrix notation when considering (infinitely) long input and impulse response sequences)

Discrete-Time Systems 5/13

Z-Transform of system h[k] and signals u[k],y[k]

Definition: $H(z)\overset{\Delta}{=}\sum_k h[k].z^{-k}$ $U(z)\overset{\Delta}{=}\sum_k u[k].z^{-k}$ $Y(z)\overset{\Delta}{=}\sum_k y[k].z^{-k}$

Input/output relation:

$$\underbrace{\begin{bmatrix}1 & z^{-1} & z^{-2} & z^{-3} & z^{-4} & z^{-5}\end{bmatrix}\begin{bmatrix}y[0]\\y[1]\\y[2]\\y[3]\\y[4]\\y[5]\end{bmatrix}}_{Y(z)}=\underbrace{\begin{bmatrix}1 & z^{-1} & z^{-2} & z^{-3} & z^{-4} & z^{-5}\end{bmatrix}\begin{bmatrix}h[0]&0&0&0\\h[1]&h[0]&0&0\\h[2]&h[1]&h[0]&0\\0&h[2]&h[1]&h[0]\\0&0&h[2]&h[1]\\0&0&0&h[2]\end{bmatrix}}_{H(z).\begin{bmatrix}1 & z^{-1} & z^{-2} & z^{-3}\end{bmatrix}}.\begin{bmatrix}u[0]\\u[1]\\u[2]\\u[3]\end{bmatrix}$$

$$\Rightarrow Y(z) = H(z).U(z)$$

H(z) is `transfer function'

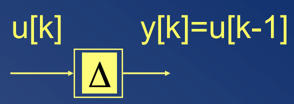
Discrete-Time Systems 6/13

Z-Transform

- Easy input-output relation: $Y(z) = H(z).U(z)$
- May be viewed as `shorthand' notation
  (for convolution operation/Toeplitz-vector product)
- Stability
  =bounded input u[k] leads to bounded output y[k]
  --iff $\sum_k |h[k]| \prec \infty$
  --iff all the poles of H(z) lie inside the unit circle
  (now z=complex variable)
  (for causal, rational systems, see below)

Discrete-Time Systems 7/13

Example-1 : `Delay operator'

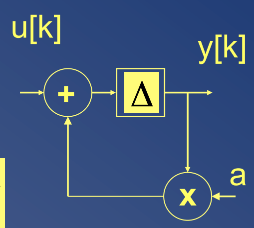


Impulse response is …,0,0, 0 ,1,0,0,0,… (K=0)

Transfer function is $H(z) = z^{-1} = \frac{1}{z}$

Pole at z=0

Example-2 : Delay + feedback



Impulse response is …,0,0, 0 ,1,a,a^2,a^3… (K=0)

Transfer function is

$$H(z) = z^{-1} + a.z^{-2} + a^2.z^{-3} + a^3.z^{-4} + ...$$

$$\Rightarrow H(z) - a.z^{-1}H(z) = z^{-1}$$

$$\Rightarrow H(z) = \frac{z^{-1}}{1 - a.z^{-1}} = \frac{1}{z - a}$$

Pole at z=a

=simple rational function

realized with a delay element, a multiplier and an adder

Discrete-Time Systems 8/13

Will consider only rational transfer functions:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 z^L + b_1 z^{L-1} + ... + b_L}{z^L + a_1 z^{L-1} + ... + a_L} = \frac{b_0 + b_1 z^{-1} + ... + b_L z^{-L}}{1 + a_1 z^{-1} + ... + a_L z^{-L}}$$

- L poles (zeros of A(z)) , L zeros (zeros of B(z))
- Corresponds to **difference equation**

$$Y(z) = H(z).U(z) \Rightarrow A(z).Y(z) = B(z).U(z) \Rightarrow ...$$

$$y[k] = b_0.u[k] + b_1.u[k-1] + ... + b_L.u[k-L] - a_1.y[k-1] - ... - a_L.y[k-L]$$

- Hence rational H(z) can be realized with finite number of delay elements, multipliers and adders
- In general, this is a `infinitely long impulse response' (`**IIR**') system (as in example-2)

Discrete-Time Systems 9/13

Special case is

$$H(z) = \frac{B(z)}{z^L} = b_0 + b_1 z^{-1} + \ldots + b_L z^{-L}$$

- L poles at the origin z=0 (hence guaranteed stability)
- L zeros (zeros of B(z)) = `all zero' filter
- Corresponds to **difference equation**

$$Y(z) = H(z).U(z) \Rightarrow y[k] = b_0.u[k] + b_1.u[k-1] + \ldots + b_L.u[k-L]$$

=`moving average' (MA) filter

- Impulse response h[k] is

$$0,0,0,b_0,b_1,\ldots,b_{L-1},b_L,0,0,0,\ldots$$

= `finite impulse response' (`**FIR**') filter

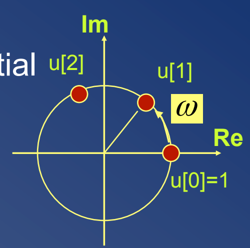
Discrete-Time Systems 10/13

H(z) & frequency response:

- Given a system H(z)
- Given an input signal = complex exponential

$$u[k] = e^{j\omega k} \qquad -\infty \prec k \prec \infty$$
$$= \cos(\omega k) + j.\sin(\omega k)$$

(where ω=radial frequency)



- Output signal :

$$y[k] = \sum_{\bar{k}} h[\bar{k}].u[k-\bar{k}] = \sum_{\bar{k}} h[\bar{k}].e^{j\omega(k-\bar{k})} = e^{j\omega k} \sum_{\bar{k}} h[\bar{k}].e^{-j\omega\bar{k}} = u[k].H(e^{j\omega})$$

$H(e^{j\omega})$ = `frequency response'
= complex function of radial frequency ω
= H(z) evaluated on the unit circle

Discrete-Time Systems 11/13

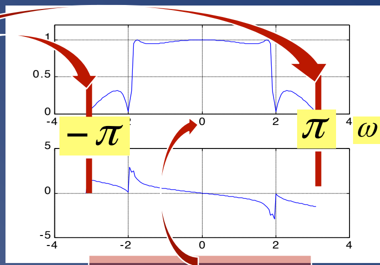
H(z) & frequency response: $H(e^{j\omega})$

- Periodic with period = 2π
- For a real-valued impulse response h[k]
  - magnitude response $|H(e^{j\omega})|$ is even function
  - phase response $\angle H(e^{j\omega})$ is odd function
- Example-1: Low-pass filter

Nyquist frequency

$$e^{j\pi k} = \ldots,1,-1,1,-1,1,\ldots$$

(=2 samples/period)



DC

$$e^{j0k} = \ldots,1,1,1,1,1,\ldots$$

- Example-2: All-pass filter $|H(e^{j\omega})| \equiv 1$

Discrete-Time Systems 12/13

- **Z-Transform & Discrete-Time Fourier Transform**

$$z = e^{j\omega}$$

$$Y(z) = H(z).U(z) \rightarrow Y(e^{j\omega}) = H(e^{j\omega}).U(e^{j\omega})$$

- $H(e^{j\omega})$ is frequency response of the LTI system
- $U(e^{j\omega})$ is frequency spectrum ('Discrete-Time Fourier Transform') of input signal

$$U(e^{j\omega}) = U(z)\Big|_{z=e^{j\omega}} = \sum_{k=-\infty}^{+\infty} u[k].z^{-k}\Big|_{z=e^{j\omega}} = \sum_{k=-\infty}^{+\infty} u[k].e^{-j\omega.k}$$

(compare to Fourier Transform, see p.3)

- $Y(e^{j\omega})$ is frequency spectrum of the output signal

Discrete-Time Systems 13/13

- **Z-Transform & Fourier Transform** It is proved that…
  - The **frequency response** $H(e^{j\omega})$ of an LTI system is equal to the Fourier transform of the continuous-time impulse sequence (see p.5) constructed with h[k]

e.g. $f = \frac{f_s}{2} \Rightarrow \omega = \pi$

$$H(e^{j\omega}) = \ldots = \mathcal{F}\{h_D(t)\} = \mathcal{F}\{\sum_k h[k].\delta(t-k.T_s)\} \quad , \quad \omega = 2\pi.\frac{f}{f_s}$$

  - The **frequency spectrum** $U(e^{j\omega})$ or $Y(e^{j\omega})$ of a discrete-time signal is equal to the Fourier transform of the continuous-time impulse sequence constructed with u[k] or y[k]

$$U(e^{j\omega}) = \ldots = \mathcal{F}\{u_D(t)\} = \mathcal{F}\{\sum_k u[k].\delta(t-k.T_s)\} \quad , \quad \omega = 2\pi.\frac{f}{f_s}$$

  - $\mathcal{F}\{y_D(t)\} = \mathcal{F}\{h_D(t)\}.\mathcal{F}\{u_D(t)\}$ corresponds to continuous-time $Y(f) = H(f).U(f)$ iff $U(f), Y(f), H(f)$ are bandlimited (→no aliasing)

Discrete/Fast Fourier Transform 1/4

- **DFT definition:**
  - The `Discrete-time Fourier Transform' of a discrete-time system/signal x[k] is a (periodic) continuous function of ω

$$X(e^{j\omega}) = \left[\sum_{k=-\infty}^{+\infty} x[k].z^{-k} \right]_{z=e^{j\omega}}$$

(see p.28)

  - The `Discrete Fourier Transform' (DFT) is a discretized version of this, obtained by sampling ω at N uniformly spaced frequencies $\omega_n = 2\pi.n/N$ (n=0,1,..,N-1) and by truncating x[k] to N samples (k=0,1,..,N-1)

$$X[e^{j\frac{2\pi.n}{N}}] = \sum_{k=0}^{N-1} x[k].\, e^{-j\frac{2\pi.n}{N}k}$$

Discrete/Fast Fourier Transform 2/4

- **DFT & Inverse DFT (IDFT):**
  - An *N*-point DFT sequence can be calculated from an *N*-point time sequence:

$$X[e^{j\frac{2\pi.n}{N}}] = \sum_{k=0}^{N-1} x[k].\, e^{-j\frac{2\pi.n}{N}k}$$ **= DFT**

  - Conversely, an *N*-point time sequence can be calculated from an *N*-point DFT sequence:

$$x[k] = \frac{1}{N}\sum_{n=0}^{N-1} X[e^{j\frac{2\pi.n}{N}}].\, e^{j\frac{2\pi.n}{N}k}$$ **= IDFT**

Discrete/Fast Fourier Transform 3/4

- **DFT/IDFT in matrix form**
  - Using shorthand notation.. $\begin{cases} X[n] = X\left[e^{j\frac{2\pi}{N}n}\right] \\ W_N = e^{-j\frac{2\pi}{N}} \end{cases}$
  - ..the **DFT** can be rewritten as

$$\begin{bmatrix} X[0] \\ X[1] \\ \cdots \\ X[N-1] \end{bmatrix} = \begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & \cdots & W_N^{(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ W_N^0 & W_N^{(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}$$

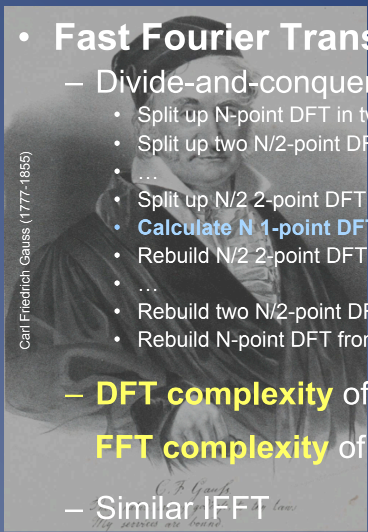
$$\mathbf{X} = \mathbf{F}.\mathbf{x}$$

  - ..the **IDFT** can be rewritten as

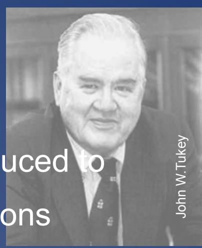
$$\begin{bmatrix} x[0] \\ x[1] \\ \cdots \\ x[N-1] \end{bmatrix} = \frac{1}{N} \begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^{-1} & \cdots & W_N^{-(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ W_N^0 & W_N^{-(N-1)} & \cdots & W_N^{-(N-1)^2} \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix}$$

$$\mathbf{x} = \mathbf{F}^{-1}.\mathbf{X} = \frac{1}{N}\mathbf{F}^*.\mathbf{X}$$

Discrete/Fast Fourier Transform 4/4



- **Fast Fourier Transform (FFT)** (1805/1965)
  - Divide-and-conquer approach:
    - Split up N-point DFT in two N/2-point DFT's
    - Split up two N/2-point DFT's in four N/4-point DFT's
    - …
    - Split up N/2 2-point DFT's in N 1-point DFT's
    - **Calculate N 1-point DFT's**
    - Rebuild N/2 2-point DFT's from N 1-point DFT's
    - …
    - Rebuild two N/2-point DFT's from four N/4-point DFT's
    - Rebuild N-point DFT from two N/2-point DFT's
  - **DFT complexity** of N^2 multiplications is reduced to **FFT complexity** of $O(N.\log_2(N))$ multiplications
  - Similar IFFT



Multi-Rate Systems 1/11

- **Decimation : decimator (=downsampler)**

  u[0],u[1],u[2]... → ↓D → u[0], u[D], u[2D]...

  Example : u[k]: 1,2,3,4,5,6,7,8,9,…
  2-fold downsampling: 1,3,5,7,9,...

- **Interpolation : expander (=upsampler)**

  u[0], u[1], u[2],... → ↑D → u[0],0,..0,u[1],0,…,0,u[2]...

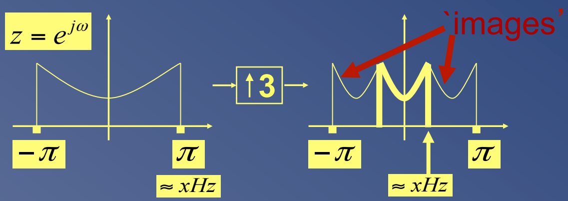
  Example : u[k]: 1,2,3,4,5,6,7,8,9,…
  2-fold upsampling: 1,0,2,0,3,0,4,0,5,0...

Multi-Rate Systems 2/11

- **Z-transform & frequency domain analysis of expander**

u[0], u[1], u[2],... → ↑D → u[0],0,..0,u[1],0,…,0,u[2]...

$U(z)$ → ↑D → $U(z^D)$



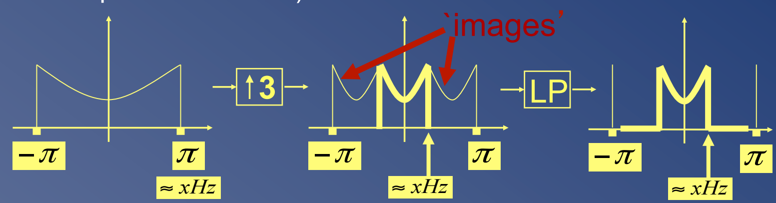
`Expansion in time domain ~ compression in frequency domain'

Multi-Rate Systems 3/11

- **Z-transform & frequency domain analysis of expander**

u[0], u[1], u[2],... → ↑D → u[0],0,..0,u[1],0,…,0,u[2]...

Expander mostly followed by `interpolation filter' to remove images (and `interpolate the zeros')



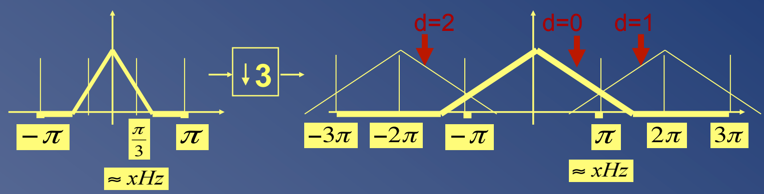
Interpolation filter can be low-/band-/high-pass (see p.35-36 and Chapter-10)

Multi-Rate Systems 4/11

- **Z-transform & frequency domain analysis of decimator**

u[0],u[1],u[2]... → ↓D → u[0], u[D], u[2D]...

$U(z)$ → ↓D → $\frac{1}{D}\cdot\sum_{d=0}^{D-1} U(z^{1/D}.e^{-j2\pi d/D})$



`Compression in time domain ~ expansion in frequency domain'

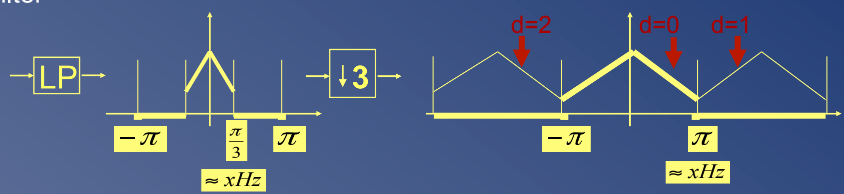
PS: Note that $U(e^{j\omega})$ is periodic with period 2π while $U(e^{j\omega/D})$ is periodic with period $2D\pi$
The summation with d=0…D-1 restores the periodicity with period 2π !

Multi-Rate Systems 5/11

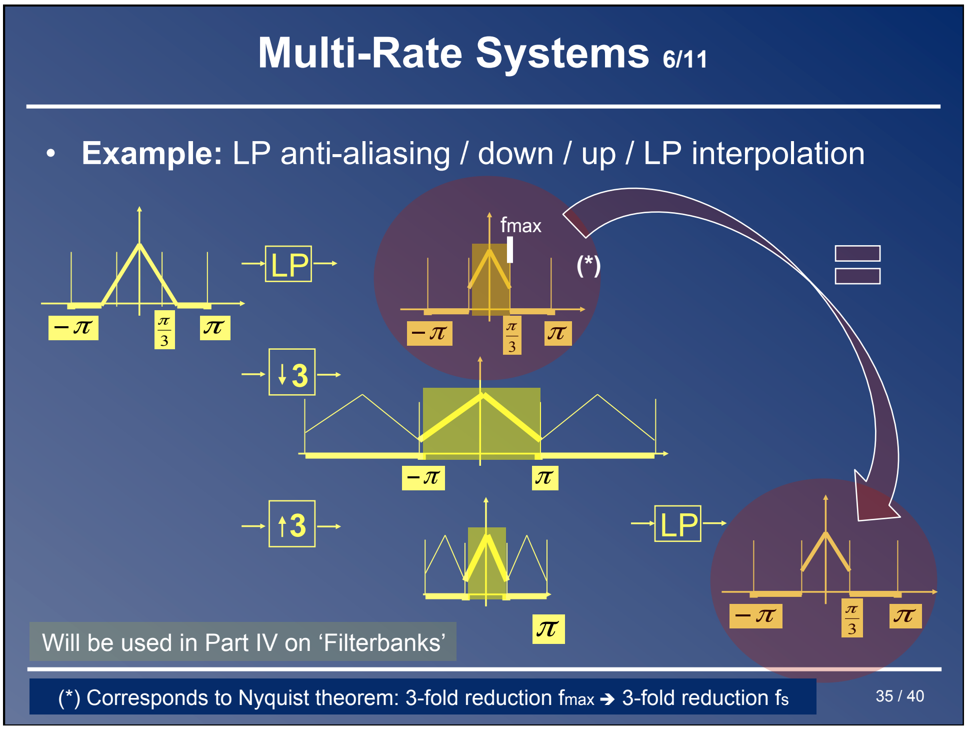
- **Z-transform & frequency domain analysis of decimator**

u[0],u[1],u[2]... → ↓D → u[0], u[D], u[2D]...

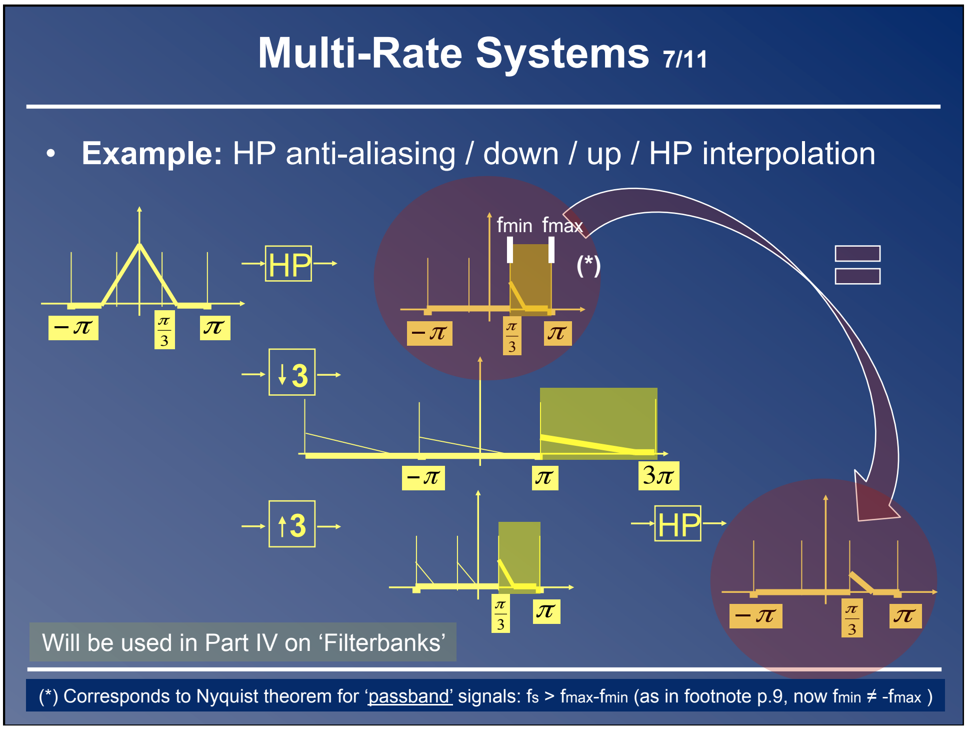
Decimation introduces <u>ALIASING</u> if input signal occupies frequency band larger than $2\pi/D$, hence mostly preceded by anti-aliasing (decimation) filter



Anti-aliasing filter can be low-/band-/high-pass (see p.35-36 and Chapter-10)



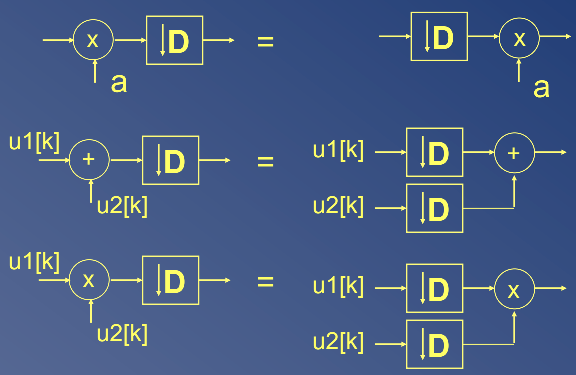
Multi-Rate Systems 6/11
• Example: LP anti-aliasing / down / up / LP interpolation
LP
fmax
(*)
↓3
↑3
LP
Will be used in Part IV on 'Filterbanks'
(*) Corresponds to Nyquist theorem: 3-fold reduction fmax → 3-fold reduction fs
35 / 40



Multi-Rate Systems 7/11
• Example: HP anti-aliasing / down / up / HP interpolation
HP
fmin fmax
(*)
↓3
↑3
HP
Will be used in Part IV on 'Filterbanks'
(*) Corresponds to Nyquist theorem for 'passband' signals: fs > fmax-fmin (as in footnote p.9, now fmin ≠ -fmax)

Multi-Rate Systems 8/10

- **Interconnection of multi-rate building blocks**

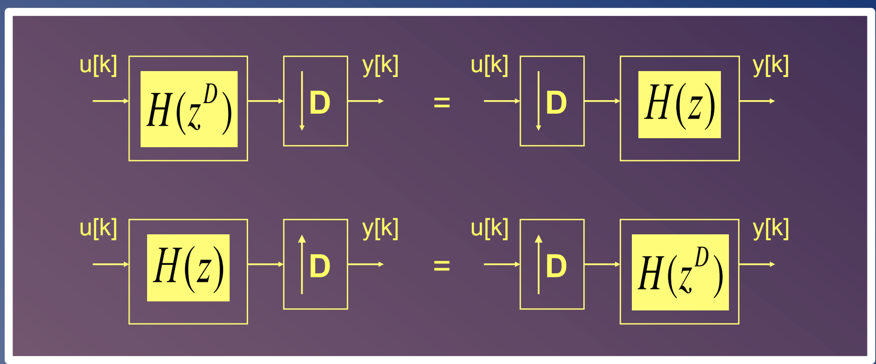


i.e. all filter operations can be performed at the lowest rate!

Identities also hold if decimators are replaced by expanders

Multi-Rate Systems 9/11

- **`Noble identities ‘**(only for rational functions)



Multi-Rate Systems 10/10

Application of `noble identities : efficient multi-rate realizations of FIR filters through…

- **Polyphase decomposition:**

Example : (2-fold decomposition)

$$H(z) = h[0] + h[1].z^{-1} + h[2].z^{-2} + h[3].z^{-3} + h[4].z^{-4} + h[5].z^{-5} + h[6].z^{-6}$$
$$= \underbrace{(h[0] + h[2].z^{-2} + h[4].z^{-4} + h[6].z^{-6})}_{E_0(z^2)} + z^{-1}.\underbrace{(h[1] + h[3].z^{-2} + h[5].z^{-4})}_{E_1(z^2)}$$

Example : (3-fold decomposition)

$$H(z) = h[0] + h[1].z^{-1} + h[2].z^{-2} + h[3].z^{-3} + h[4].z^{-4} + h[5].z^{-5} + h[6].z^{-6}$$
$$= \underbrace{(h[0] + h[3].z^{-3} + h[6].z^{-6})}_{E_0(z^3)} + z^{-1}.\underbrace{(h[1] + h[4].z^{-3})}_{E_1(z^3)} + z^{-2}.\underbrace{(h[2] + h[5].z^{-3})}_{E_2(z^3)}$$

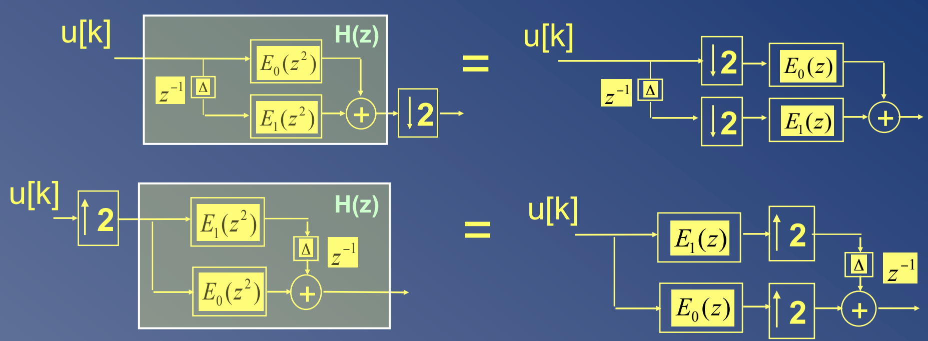
General: (D-fold decomposition)

$$H(z) = \sum_{k=-\infty}^{\infty} h[k].z^{-k} = \sum_{d=0}^{D-1} z^{-d}.E_d(z^D) \quad , \quad E_d(z) = \sum_{k=-\infty}^{\infty} h[D.k+d].z^{-k}$$

Multi-Rate Systems 11/11

- **Polyphase decomposition:**

Example : efficient realization of FIR decimation/interpolation filter



i.e. all filter operations can be performed at the lowest rate!