Enabling Technologies for Sports (5XSF0), Module 02

Basics of Signals, Sampling, Fourier series, Aliasing, and FIR Filtering

Peter H.N. de With & Piet C.W. Sommen

(p.h.n.de.with@tue.nl)

slides version 1.0

Overview Module 02

* P1: Time-discrete sampling and Spectrum basics
  - Sinusoidal signals
  - time-discrete sampling, repetition
* P2: Sampling & Aliasing, and D/A conversion
  - Aliasing, concept, aspects
  - Discrete-to-Continuous conversion and vice versa
* P3: Finite Impulse Response Filters
  - Time domain and Convolution
  - FIR filters, architectures and implementation
* P4: Introduction to Signal Transformation
  - Rotation, affine transform, orthogonal transforms

Module 02 – Part 1 Time-discrete sampling

Fundamentals of signals and sampling and the Delta function

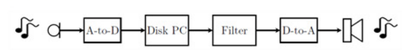
Introduction / Signals – (1) Definition

What is a signal?

Audio example:

Toilet.wav Speech in.wav Speech out.wav

How does this work?:



- Picture represents main content of this course
- Why go to digital domain?
- Many useful demos at: http://users.ece.gatech.edu/mcclella/matlabGUIs/index.html and http://www.jhu.edu/signals/

Introduction / Signals – (2) Applications

What can we do with signals in real life? – Nearly everything!

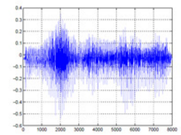
- Audio, Video: Signal enhancement, compression (e.g. MP3), surround sound, HDTV, surveillance, ...
- Biomedical: Electrocardiogram (heart), electroencephalogram (brain), activity detection, ...
- Automatic control: Process control (e.g. food industry), ...
- Broadcasting and Telecommunication: Radio, TV, Mobile phone, ...
- Navigation: Sonar , Radar, ...
- Automotive: Safety and security, (e.g. navigation for monitoring and control, situation awareness, target tracking, road prediction for collision avoidance), Infotainment and audio, Wheel speed analysis, ...
- . . .

Introduction / Signals – (3) Representation

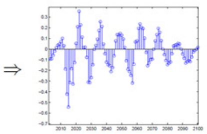
How do we express/represent signals, in what form?

Many different representations:

E.g. speech: acoustic, electrical, *string of numbers*



(zoom in range 2000 - 2100) ⇒



Many other signals:

Photo: ; Film: ; Video, radar, seismic, etc.

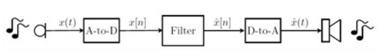
Introduction / Signals – (4) System building

We can build systems that process signals, to add/extract information value

What is a system? (in context of signals)

Something that can manipulate, change, record, or transmit signals

Audio example:



Symbols: Continuous-time (analog) → $(\cdot)$
Discrete-time → $[\cdot]$

Modeling (= mathematical description) appropriate for describing and understanding signals and systems

Start with mathematical description of sinusoidal signal

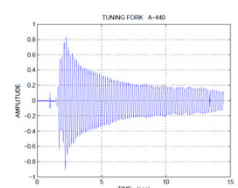
Introduction / Sinusoids – (1) Motivation

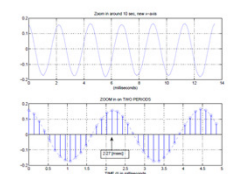
Why is sine/ cosine so important?

Sinusoids most basic in theory of signals and systems.

Example: Tuning fork

Sinusoidal signals are cornerstones of signal models and used in many real life applications…





440 Hz? From detailed figure ⇒ $f_0 = \frac{1}{2.27 \cdot 10^{-3}} = 440$ Hz

Introduction / Sinusoids – (2) Basic prop's

Sine and cosine functions are periodic, even/odd, are zero sometimes…

Property	Equation
Equivalency	$\sin(\theta) = \cos(\theta - \pi/2)$ or $\cos(\theta) = \sin(\theta + \pi/2)$
Periodicity	$\cos(\theta + k \cdot 2\pi) = \cos(\theta)$, for k integer
Evenness cos	$\cos(-\theta) = \cos(\theta)$
Oddness sin	$\sin(-\theta) = -\sin(\theta)$
Zeros sin	$\sin(k \cdot \pi) = 0$ for k integer
Ones cos	$\cos(k \cdot 2\pi) = 1$ for k integer
Minus ones cos	$\cos((k + \frac{1}{2}) \cdot 2\pi) = -1$ for k integer
Cos: slope sin	$\frac{d \sin(\theta)}{d\theta} = \cos(\theta)$
Sin: negative slope cos	$\frac{d \cos(\theta)}{d\theta} = -\sin(\theta)$

Some basic trigonometric identities (Table 2-2 book):
See: "Signal Processing First", James H. McClellan et. al.

Introduction / Sinusoids – (3) Review cosine

What is the amplitude, frequency and phase of a sinusoid?

Mathematical formula continuous-time cosine:

$$x(t) = A\cos(\omega_0 \cdot t + \phi) = A\cos(2\pi \cdot f_0 \cdot t + \phi)$$

Symbol	Name	Dimension
A	Amplitude	-
ω_0	Radian frequency	rad/sec
ϕ	phase	rad
f_0	(cyclic) frequency	sec^{-1}= Hz

Relation period - frequency:

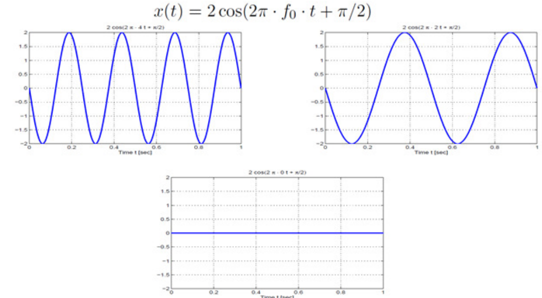
$$x(t + T_0) = x(t)$$
$$A\cos(\omega_0 \cdot (t + T_0) + \phi) = A\cos(\omega_0 \cdot t + \phi)$$
$$\omega_0 T_0 = 2\pi \Rightarrow T_0 = \frac{2\pi}{\omega_0} \text{ or } (2\pi f_0)T_0 = 2\pi \Rightarrow T_0 = \frac{1}{f_0}$$

Introduction / Sinusoids – (4) Example

Cosine function for several values of the frequency

Amplitude $A = 2$, Phase $\phi = \pi/2$, Frequency f_0 varies $4 \rightarrow 0$ [Hz]

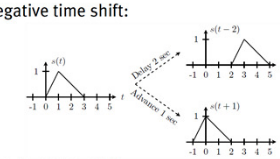
$$x(t) = 2\cos(2\pi \cdot f_0 \cdot t + \pi/2)$$



Introduction / Sinusoids – (5) Phase/time

Positive and negative time shift:

Phase can be shifted to time and vice versa



Conversion time- to phase shift:

$$x(t) = \cos(\omega_0 t) \Rightarrow x(t - t_1) = \cos(\omega_0(t - t_1)) = \cos(\omega_0 t + \phi)$$
$$\phi = -\omega_0 t_1 \Leftrightarrow t_1 = -\frac{\phi}{\omega_0} = -\frac{\phi}{2\pi f_0}$$

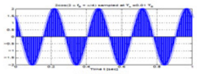
Notes:

- Phase negative for positive time shift (delay)
- Phase can always be chosen in $-\pi < \phi \leq \pi$. Why?

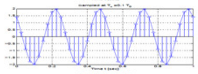
Intro / Sinusoids – (6) Plotting & sampling

$x(t) = 2\cos(2\pi \cdot 4 \cdot t + \pi/4) \rightarrow$ One period? T_0=1/4 =0.25 [sec]

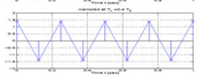
$T_s = 0.0025$ [sec] $\rightarrow$ 100 samples in T_0



$T_s = 0.025$ [sec] $\rightarrow$ 10 samples in T_0



$T_s = 0.1$ [sec] $\rightarrow$ 2.5 samples in T_0

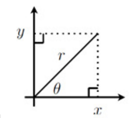


Notes:

- Choice of T_s depends on frequency cosine
- String of numbers at sampling space T_s (`stem`)
- Reconstruction: Matlab (`plot`) connects samples by lines
- How large T_s for accurate reconstruction?

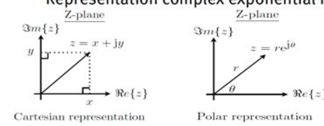
Intro / Sinusoids – (7) Complex exponents

Complex exponentials and phasors



$$\sin(\theta) = \frac{y}{r} \rightarrow y = r\sin(\theta)$$
$$\cos(\theta) = \frac{x}{r} \rightarrow x = r\cos(\theta)$$
$$r = \sqrt{x^2 + y^2} \quad \theta = \arctan\left(\frac{y}{x}\right)$$
$$\text{if } x < 0 \quad \theta = \arctan\left(\frac{y}{x}\right) + \pi$$

Representation complex exponential in Z-plane ($\mathrm{j} = \sqrt{-1}$)



$$\Re e\{r e^{\mathrm{j}\theta}\} = r\cos(\theta)$$
$$\Im m\{r e^{\mathrm{j}\theta}\} = r\sin(\theta)$$

Euler : $e^{\mathrm{j}\theta} = \cos(\theta) + \mathrm{j}\sin(\theta)$

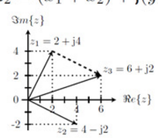
Intro / Sinusoids – (8) Complex addition

Complex addition of numbers and conversions

$$z_1 = x_1 + \mathrm{j}y_1 \text{ ; } z_2 = x_2 + \mathrm{j}y_2 \rightarrow z_3 = z_1 + z_2 = (x_1 + x_2) + \mathrm{j}(y_1 + y_2)$$

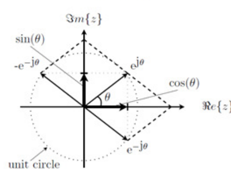
Example:

$$z_1 = 2 + \mathrm{j}4$$
$$z_2 = 4 - \mathrm{j}2$$
$$z_3 = 6 + \mathrm{j}2$$



Inverse Euler:

$$\cos(\theta) = \frac{e^{\mathrm{j}\theta} + e^{-\mathrm{j}\theta}}{2}$$
$$\sin(\theta) = \frac{e^{\mathrm{j}\theta} - e^{-\mathrm{j}\theta}}{2\mathrm{j}}$$



Intro / Sinusoids – (9) Complex multiplication

Best way via polar representation:

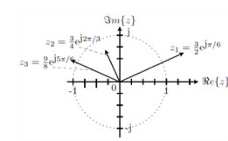
Complex multiplication multiplies radii and adds angles

$$z_1 = r_1 e^{\mathrm{j}\theta_1} \qquad z_2 = r_2 e^{\mathrm{j}\theta_2}$$
$$z_3 = z_1 \cdot z_2 = r_1 e^{\mathrm{j}\theta_1} \cdot r_2 e^{\mathrm{j}\theta_2} = r_1 r_2 e^{\mathrm{j}(\theta_1 + \theta_2)} = r e^{\mathrm{j}\theta}$$

$r = r_1 \cdot r_2$ and $\theta = \theta_1 + \theta_2$

Example:

$$z_1 = \frac{3}{2} e^{\mathrm{j}\pi/6}$$
$$z_2 = \frac{3}{4} e^{\mathrm{j}2\pi/3}$$
$$z_3 = \frac{9}{8} e^{\mathrm{j}5\pi/6}$$



Intro / Sinusoids – (10) Complex exp. signals

Complex exponential signals are easier to handle for sinusoidal signals

$$z(t) = A\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi)}$$

Magnitude: $|z(t)| = A$ and Angle: $\arg\{z(t)\} = \omega_0 t + \phi$

Use Euler $\rightarrow z(t) = A\cos(\omega_0 t+\phi) + \mathrm{j}A\sin(\omega_0 t+\phi)$

Amplitude A; Phase shift ϕ and Frequency ω_0

Complex exponential is alternative representation of real cosine (sine):

$$\Re e\{A\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi)}\} = A\cos(\omega_0 t+\phi) \quad ; \quad \Im m\{A\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi)}\} = A\sin(\omega_0 t+\phi)$$

Rotating phaser:

Complex exponentials: Simplifies manipulation with sinusoidal signals

Intro / Sinusoids – (11) Basic trigonometrics

Complex exponential signals are easier to handle for sinusoidal signals

1	$\sin^2(\theta) + \cos^2(\theta) = 1$
2	$\cos(2\theta) = \cos^2(\theta) - \sin^2(\theta)$
3	$\sin(2\theta) = 2\sin(\theta)\cos(\theta)$
4	$\sin(\alpha \pm \beta) = \sin(\alpha)\cos(\beta) \pm \cos(\alpha)\sin(\beta)$
5	$\cos(\alpha \pm \beta) = \cos(\alpha)\cos(\beta) \mp \sin(\alpha)\sin(\beta)$

Simple proof via Euler, e.g. property 1:

$$\begin{aligned}\sin^2(\theta) + \cos^2(\theta) &= \left(\frac{\mathrm{e}^{\mathrm{j}\theta} - \mathrm{e}^{-\mathrm{j}\theta}}{2\mathrm{j}}\right)^2 + \left(\frac{\mathrm{e}^{\mathrm{j}\theta} + \mathrm{e}^{-\mathrm{j}\theta}}{2}\right)^2 \\ &= \left(\frac{\mathrm{e}^{\mathrm{j}2\theta} + \mathrm{e}^{-\mathrm{j}2\theta} - 2}{4(\mathrm{j})^2}\right) + \left(\frac{\mathrm{e}^{\mathrm{j}2\theta} + \mathrm{e}^{-\mathrm{j}2\theta} + 2}{4}\right) \\ &= \left(\frac{-\mathrm{e}^{\mathrm{j}2\theta} - \mathrm{e}^{-\mathrm{j}2\theta} + 2}{4}\right) + \left(\frac{\mathrm{e}^{\mathrm{j}2\theta} + \mathrm{e}^{-\mathrm{j}2\theta} + 2}{4}\right) = 1\end{aligned}$$

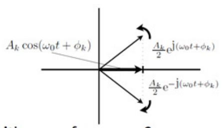
Intro / Sinusoids – (12) Phasor addition rule

Sum cosine signals, same frequency $\rightarrow$ Single cosine, same frequency:

$$x(t) = \sum_{k=1}^{N} A_k\cos(\omega_0 t + \phi_k) = A\cos(\omega_0 t + \phi)$$

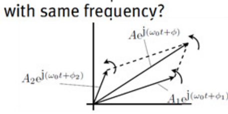
Main lines of 'proof':

$$A_k\cos(\omega_0 t + \phi_k) = A_k\left(\frac{\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi_k)} + \mathrm{e}^{-\mathrm{j}(\omega_0 t+\phi_k)}}{2}\right)$$



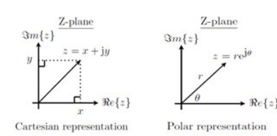
Sum of different complex exponentials with same frequency?

$$A_1\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi_1)} + A_2\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi_2)} = A\mathrm{e}^{\mathrm{j}(\omega_0 t+\phi)}$$



Introduction / Sinusoids – (13) Summary

- Periodic sine (cosine) wave:
  $x(t) = A\cos(\omega_0 \cdot t + \phi) = A\cos(2\pi \cdot f_0 \cdot t + \phi)$
- Relation period sine wave and frequency: $T_0 = 1/f_0$
- Relation phase- and time shift (delay):
  $x(t) = \cos(\omega_0 t)\,;\, x(t-\tau) = \cos(\omega_0 t + \phi) \Rightarrow \tau = -\frac{\phi}{\omega_0} = -\frac{\phi}{2\pi f_0}$
- Representation complex exponential in Z-plane ($\mathrm{j} = \sqrt{-1}$)



$$\begin{aligned}\Re e\{r\mathrm{e}^{\mathrm{j}\theta}\} &= x = r\cos(\theta) \\ \Im m\{r\mathrm{e}^{\mathrm{j}\theta}\} &= y = r\sin(\theta) \\ \theta &= \arctan\left(\frac{y}{x}\right) \\ \text{if } x < 0 \quad &= \arctan\left(\frac{y}{x}\right) + \pi\end{aligned}$$

- Euler and inverse Euler:
  $\mathrm{e}^{\mathrm{j}\theta} = \cos(\theta) + \mathrm{j}\sin(\theta) \leftrightarrow \cos(\theta) = \frac{\mathrm{e}^{\mathrm{j}\theta} + \mathrm{e}^{-\mathrm{j}\theta}}{2}\,;\, \sin(\theta) = \frac{\mathrm{e}^{\mathrm{j}\theta} - \mathrm{e}^{-\mathrm{j}\theta}}{2\mathrm{j}}$

Introduction / Sinusoids – (14) Summary

- Addition rule complex numbers: $z_1 = x_1 + \mathrm{j}y_1$; $z_2 = x_2 + \mathrm{j}y_2$

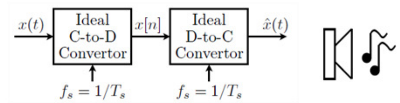
  $\Rightarrow z_3 = z_1 + z_2 = (x_1 + x_2) + \mathrm{j}(y_1 + y_2)$
- Multiplication rule complex numbers: $z_1 = r_1 e^{\mathrm{j}\theta_1}$; $z_2 = r_2 e^{\mathrm{j}\theta_2}$

  $\Rightarrow z_3 = z_1 \cdot z_2 = r e^{\mathrm{j}\theta}$ with $r = r_1 \cdot r_2$ and $\theta = \theta_1 + \theta_2$
- Complex exponential (phasor): $z(t) = A e^{\mathrm{j}(\omega_0 t + \phi)} = A e^{\mathrm{j}\phi} \cdot e^{\mathrm{j}\omega_0 t}$

  $\Re e\{z(t)\} = A\cos(\omega_0 t + \phi)$; $\Im m\{z(t)\} = A\sin(\omega_0 t + \phi)$
- Sum cosine signals with same frequency (phasor addition rule):

  $\sum_{k=1}^{N} A_k \cos(\omega_0 t + \phi_k) = A\cos(\omega_0 t + \phi) = \Re e\{A e^{\mathrm{j}\phi} e^{\mathrm{j}\omega_0 t}\}$

  with $A e^{\mathrm{j}\phi} = \sum_{k=1}^{N} A_k e^{\mathrm{j}\phi_k}$



Module 02 – Part 3
Sampling and aliasing

Sampling frequency aspects using the cosine waveform as a basis

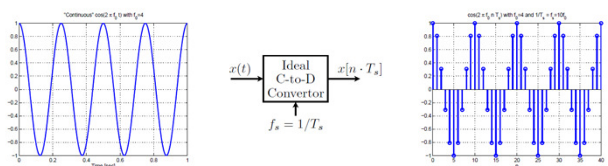
Sampling and aliasing – (1)

Conversion analog (continuous-time) $\leftrightarrow$ digital (discrete-time) domain?

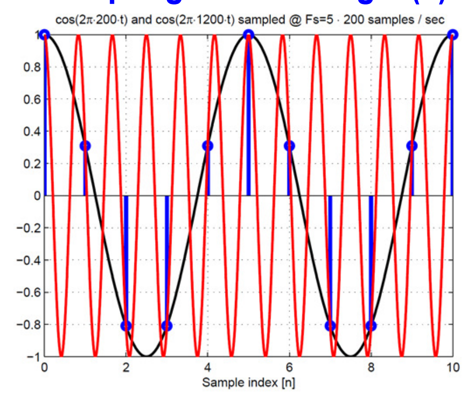
Analog (continuous-time) signal: $x(t) = A\cos(\omega \cdot t + \phi)$

Computer (e.g. Matlab) represents **discrete-time** data (samples):

$$"x[n]" = x[n \cdot T_s] = x(t)|_{t=n \cdot T_s} = A\cos(\omega \cdot n \cdot T_s + \phi) \qquad -\infty < n < \infty$$



Sampling and aliasing – (2)



Note that the phase changes the sampling values!

Absolute and relative frequency – (1)

With continuous signal $x(t) = A\cos(\omega \cdot t + \phi)$

$$\begin{aligned} x[n \cdot T_s] &= x(t)|_{t=n \cdot T_s} = A\cos(\omega \cdot n \cdot T_s + \phi) \\ &= A\cos((\omega \cdot T_s) \cdot n + \phi) = A\cos(\theta \cdot n + \phi) \end{aligned}$$

Absolute frequency: $\omega = 2\pi f$ [rad/sec] (f [Hz])

Relative (normalized) frequency: $\theta = \omega \cdot T_s$ [rad] (dimensionless)

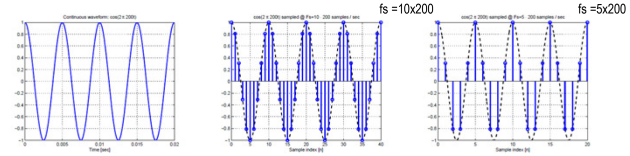
Note: After sampling time scale information lost

$\Rightarrow$ Discrete-time signal $x[n \cdot T_s]$ (or $x[n]$) is sequence of numbers

$\Rightarrow$ **No unique relation between analog and discrete-time domain, e.g.:**

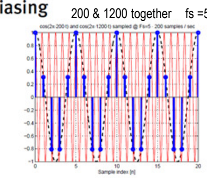
$\omega = 200\pi$ [rad/sec] $T_s = 1/2000$ [sec] $\rightarrow \theta = 0.1\pi$ [rad]
$\omega = 1000\pi$ [rad/sec] $T_s = 1/10000$ [sec] $\rightarrow \theta = 0.1\pi$ [rad]

Concept of aliasing – (1)



Other frequencies resulting in same samples (e.g. as in right figure)?
Use property $\cos(\theta \cdot n) = \cos((\theta + 2\pi) \cdot n) \Rightarrow$ **Aliasing**

$$\begin{aligned} x_1[n] &= \cos(2\pi \cdot \underline{200} \cdot n \cdot \frac{1}{1000}) \\ &= \cos(0.4\pi \cdot n) \\ &= \cos((0.4\pi + 2\pi) \cdot n) = \cos(2.4\pi \cdot n) \\ &= \cos(2\pi \cdot ?\underline{1200} \cdot n \cdot \frac{1}{1000}) \end{aligned}$$



Concept of aliasing – (2)

Other possibility by using $\cos(\theta n) = \cos((2\pi - \theta)n)$

Note: Phase must be opposite sign $\cos(\theta n + \phi) = \cos((2\pi - \theta)n - \phi)$

$\Rightarrow$ Sinus with frequency θ_0 has following (infinite) alias frequencies:

$\theta_0 + 2\pi l \, ; 2\pi l - \theta_0$ for integer l

since:

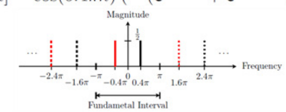
$$A\cos(\theta_0 n + \phi) = A\cos((\theta_0 + 2\pi l)n + \phi) = A\cos((2\pi l - \theta_0)n - \phi)$$

Conclusion:

When making a stem plot of all these signals, with specific A, θ_0, ϕ and l, we see no difference

Spectrum of discrete-time signal – (1)

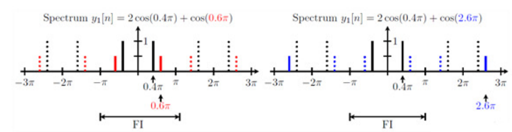
Spectrum of $x_1[n] = \cos(0.4\pi n)$ $(= (e^{j0.4\pi n} + e^{-j0.4\pi n})/2)$



Spectrum analog cosine two components , here ∞ components

One interval length 2π relevant: **Fundamental Interval (FI)** $\{-\pi, \pi\}$

Example: Two different discrete-time signals, same spectra



Spectrum of discrete-time signal – (2)

Spectral lines of $x(t) = A\cos(2\pi f_0 t + \phi)$ at $\pm f_0$ with amplitude $\frac{1}{2}A\mathrm{e}^{\pm \mathrm{j}\phi}$

What about spectral lines of discrete-time signal? ($\theta_0 = f_0/f_s$)

$$\begin{aligned} x[n] &= A\cos(\theta_0 n + \phi) \\ &= \frac{1}{2}A\mathrm{e}^{\mathrm{j}\phi}\mathrm{e}^{\mathrm{j}\theta_0 n} + \frac{1}{2}A\mathrm{e}^{-\mathrm{j}\phi}\mathrm{e}^{-\mathrm{j}\theta_0 n} \end{aligned}$$

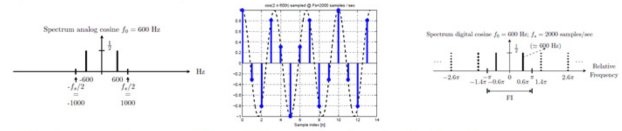
Due to aliasing spectral lines of $x[n]$ at:

$$\theta = \theta_0 + 2\pi l \quad \text{for} \quad l = 0, \pm 1, \pm 2, \cdots$$
$$\theta = -\theta_0 + 2\pi l \quad \text{for} \quad l = 0, \pm 1, \pm 2, \cdots$$

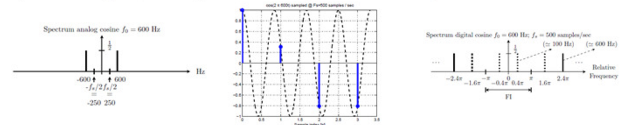
Spectrum of discrete-time signal – (3)

Example: $x(t) = \cos(2\pi(600)t)$ ($f_{max} = 600$), different sample rates

Oversampling case: $f_s = 2000$ samples/sec ($> 2 \cdot f_{max}$)

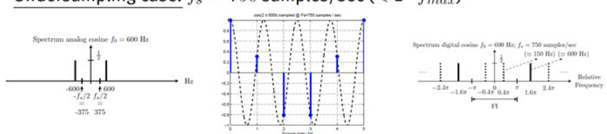


Undersampling case: $f_s = 500$ samples/sec ($< 2 \cdot f_{max}$)



Spectrum of discrete-time signal – (4)

Undersampling case: $f_s = 750$ samples/sec ($< 2 \cdot f_{max}$)



Same results in movies:

Analog cosine with frequency $f_0 = 600$ Hz samples at different rates:

f_s [Hz]	500	750	2000
Film	🎞	🎞	🎞

What happens when $f_s = 2 \cdot 600$ Hz = 1200 Hz?

Reconstruction – (1)

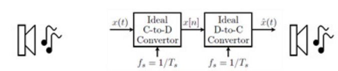
Relation between analog frequency (ω) and discrete-time frequency (θ):

$$\omega = 2\pi f \quad \text{while} \quad \theta = 2\pi(f/f_s) \quad \text{thus} \quad \omega = \theta \cdot f_s$$

$\Rightarrow$ Each discrete-time frequency mapped to continuous-time frequency

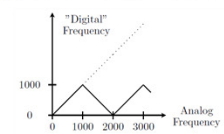
Take care alias terms $\rightarrow$ Select (e.g.) FI=$\{-\pi, \pi\} \simeq \{-\frac{f_s}{2}, \frac{f_s}{2}\}$

Chirp



How does $x(t)$ and $\hat{x}(t)$ sound when $f_s = 2000$ samples/sec?

Explanation:



The Shannon Sampling Theorem

How often sample for enough information to reconstruct original continuous (analog) signal?

Shannon Sampling Theorem

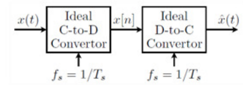
Continuous-time signal $x(t)$ with frequencies no higher than f_{max} can be reconstructed exactly from its samples $x[n] = x(t)|_{t=n \cdot T_s}$, if samples are taken at a rate $f_s = 1/T_s$, that is greater than $2f_{max}$

Nyquist rate: Minimum sampling rate of $2f_{max}$

Two issues:

- Minimum sample rate depends on frequency content (f_{max})of $x(t)$
- How reconstruct continuous-time signal $x(t)$ from the samples?

Ideal reconstruction (Shannon's theorem)



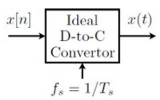
Conclusion from Shannon Sampling Theorem

If $x(t)$ contains no frequencies higher than f_{max} and $f_s > 2 \cdot f_{max}$ output $\hat{x}(t)$ of ideal D-to-C convertor equal to input $x(t)$ ideal C-to-D convertor

Why sample rate audio CD 44.1 kHz?
$\rightarrow$ 20 kHz upper limit human hearing/ perception musical signals

What happens if we don't sample fast enough? $\rightarrow$ Aliasing

Discrete-to-Continous conversion – (1)



$x(t) = x[n]|_{n=f_s t}$ correct?

For samples sinusoidal signals with $f_0 < \frac{1}{2} f_s$ only:

Ideal D-to-C in effect replaces n by $f_s \cdot t$, thus

$$x[n] = A\cos(2\pi f_o n T_s + \phi) \quad \text{with} \quad f_0 < \frac{f_s}{2} = \frac{1}{2T_s}$$

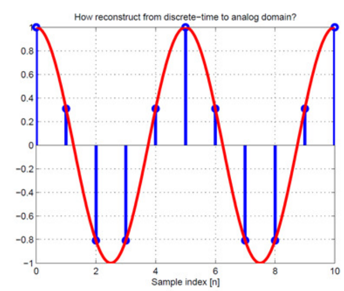
$$\Rightarrow \quad \hat{x}(t) = x[n]|_{n=f_s t} = A\cos(2\pi f_0 t + \phi)$$

What if $f_0 > \frac{1}{2} f_s$?

Ideal D-to-C converts to alias frequency less than $\frac{f_s}{2}$

Discrete-to-Continous conversion – (2)

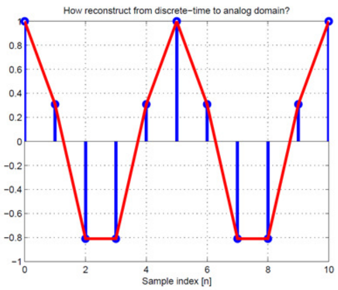
How interpolate smooth continuous-time function through samples $x[n]$?



For sinusoidal signals (mathematical expression of samples $\equiv \cos(\cdots)$)

Discrete-to-Continous conversion – (3)

How interpolate smooth continuous-time function through samples $x[n]$?



In general: Interpolation by D-to-A convertor (= **approximation**)

Different pulses for non-ideal D-to-C conv.

Square: $p(t) = \begin{cases} 1 & -\frac{1}{2}T_s < t \leq \frac{1}{2}T_s \\ 0 & \text{otherwise} \end{cases}$

Triangular: $p(t) = \begin{cases} 1 - \frac{|t|}{T_s} & -T_s < t \leq T_s \\ 0 & \text{otherwise} \end{cases}$

Cubic: Third order polynomial with $p(t)$ for $t = \pm T_s, \pm 2T_s$ and derivative smooth at sample locations

How make smooth reconstruction?
Use smooth pulse with long duration or use oversampling $\rightarrow$ original waveform does not vary much over duration of $p(t)$.

Note practical consequences of all different choises

Ideal bandlimited interpolation for D-to-C

Pulse shape ideal D-to-C conversion

$$p(t) = \frac{\sin(\frac{\pi}{T_s}t)}{\frac{\pi}{T_s}t} \quad \text{for} \quad -\infty < t < \infty$$

Note consequences of following properties of this pulse $p(t)$:

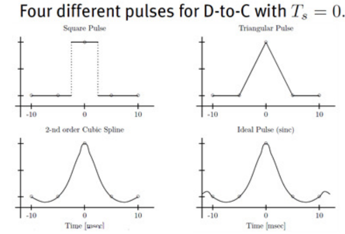
- Pulse has zeros at multiples of T_s
- Pulse is infinite long
- Reconstruct cosine wave with $f_0 < f_s/2$

Discrete-to-Continuous Conversion rev.

How does D-to-C work in general?

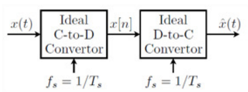
Adding, possible overlapping, shifted pulses
$\hat{x}(t) = \sum_{n=-\infty}^{\infty} x[n]p(t - nT_s)$ with $p(t)$ characteristic pulse shape convertor. Thus each $t_n = n \cdot T_s$ a pulse $p(t - nT_s)$ is emitted with amplitude $x[n]$

Four different pulses for D-to-C with $T_s = 0.005$ [sec] (i.e. $f_s = 200$ [Hz])



Pulse	Duration
Square	T_s
Triangle	$2T_s$
Cubic	$4T_s$
Ideal	∞

Sampling Theorem recapitulated



Recap: Shannon Sampling Theorem

Continuous-time signal $x(t)$ with frequencies no higher than f_{max} can be reconstructed exactly from its samples $x[n] = x(t)|_{t=n \cdot T_s}$, if samples are taken at a rate $f_s = 1/T_s$, that is greater than $2f_{max}$

Summary Sampling and Aliasing

- Sampling sinusoid: With $x(t) = A\cos(\omega t + \phi)$ and $T_s = 1/f_s$
  $x[n \cdot T_s] = x(t)|_{t=n \cdot T_s} = A\cos((\omega \cdot T_s) \cdot n + \phi) = A\cos(\theta \cdot n + \phi)$
  - Absolute frequency: $\omega = 2\pi f$ [rad/sec] (f [Hz])
  - Relative frequency: $\theta = \omega \cdot T_s = 2\pi(f/f_s)$ [rad] (dimensionless)
- Aliasing frequencies $\cos(\theta_0 n + \phi)$ with relative frequency θ_0:
  $\Rightarrow (\theta_0 + 2\pi l)n + \phi$ or $(2\pi l - \theta_0)n - \phi$ with integer l
- Spectrum discrete-time signal: Fundamental Interval (FI)
  FI $=\{-\pi, \pi\} \simeq \{-\frac{f_s}{2}, \frac{f_s}{2}\}$, periodic with ∞ components
- Ideal reconstruction sinusoidal signal: Each discrete-time frequency θ mapped via $\omega = \theta \cdot f_s$ to continuous-time frequency. Take care of alias term by selecting one period $\Rightarrow$ Choose one FI
- Sampling theorem:
  Continuous-time signal $x(t)$ with frequencies no higher than f_{max} can be reconstructed exactly from its samples $x[n] = x(t)|_{t=n \cdot T_s}$, if samples are taken at a rate $f_s = 1/T_s$, that is greater than $2f_{max}$

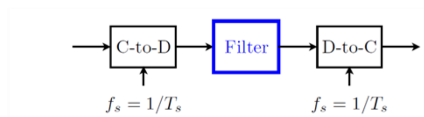
Enabling Technologies for Sports (5XSF0), Module 02, Part 3-4

Signals, Sampling, Fourier series, Aliasing, and FIR Filtering

Peter H.N. de With & Piet C.W. Sommen

(p.h.n.de.with@tue.nl)

slides version 1.0

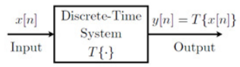


Module 02 – Part 3 Finite Impulse Response (FIR) Filters

FIR filters, convolution processing in time domain, FIR architecture

FIR filter definition

A filter is a system designed to remove some components or modify some characteristics of a signal.
Finite Impulse Response (FIR) filters: Each output sample is sum of finite number of weighted samples of input sequence



Both input and output are discrete-time samples (in contrast to e.g. D-to-C and C-to-D) $\rightarrow$ Operator $T\{\cdot\}$ described by formula

Examples:

$$y[n] = x^2[n]$$

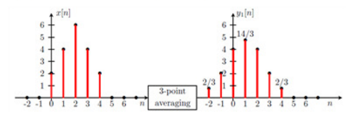
$$y[n] = \max\{x[n], x[n-1], x[n-2]\}$$

Running Average Filter – (1)

Compute a moving (running) average of two or more consecutive samples, forming a new sequence of the average values

Note: FIR is generalization of running average

Averaging used when data fluctuates, thus smoothing prior interpretation (e.g. view trends). E.g. Stock-market prices, credit-card balances, etc.
Example:
Input-output relation (=**difference equation**) 3-point averaging
$\rightarrow y_1[n] = \frac{1}{3}(x[n] + x[n+1] + x[n+2])$

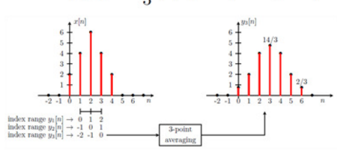


Running Average Filter – (2)

Observations output signal samples:
Smooths input , longer than input , finite sequence ,starts (becomes nonzero) before input starts (=noncausal filter)
Last issue undesirable if input comes directly from A-to-D (e.g. audio).
Causal filter uses only present and past values

Noncausal $y_2[n] = \frac{1}{3}(x[n+1] + x[n] + x[n-1])$

Causal $y_3[n] = \frac{1}{3}(x[n] + x[n-1] + x[n-2])$



General FIR filter

Difference equation general FIR: $y[n] = \sum_{k=0}^{M} b_k x[n-k]$

- Weighted running average of $M+1$ samples
- Causal filter
- M is order of FIR filter
- $L = M+1$ is length of FIR filter

Example: FIR of order $M = 3$ with coefficients $b_k = \{3, -1, 2, 1\}$

$$\rightarrow \quad y[n] = 3 \cdot x[n] - 1 \cdot x[n-1] + 2 \cdot x[n-2] + 1 \cdot x[n-3]$$

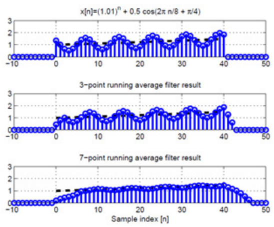
n	$n<0$	0	1	2	3	4	5	6	7	8	$n>8$
$x[n]$	0	2	4	6	4	2	0	0	0	0	0
$y[n]$	0	6	10	18	?	?	?	8	2	0	0

Illustration of FIR filtering

Find trend in the following signal:

$$x[n] = \begin{cases} (1.01)^n + \frac{1}{2}\cos(2\pi n/8 + \pi/4) & 0 \le n \le 40 \\ 0 & \text{otherwise} \end{cases}$$

Use L-point averaging filter $\rightarrow$ $y[n] = \frac{1}{L}\sum_{k=0}^{L-1} x[n-k]$

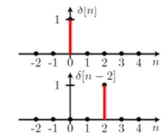


Observations:
- Run-in and -out area $y[n]$
- Length $y[n]\uparrow$
- For $L\uparrow \Rightarrow$ Less fluctuations
- Fluctuations reduced not eliminated

Unit Impulse Sequence

$$\delta[n] = \begin{cases} 1 & \text{for } n = 0 \\ 0 & \text{for } n \ne 0 \end{cases}$$

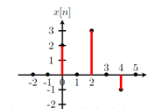
$$\delta[n-2] = \begin{cases} 1 & \text{for } n = 2 \\ 0 & \text{for } n \ne 2 \end{cases}$$



n	$\cdots$	-2	-1	0	1	2	3	4	5	6	$\cdots$
$\delta[n]$	0	0	0	1	0	0	0	0	0	0	0
$\delta[n-2]$	0	0	0	0	0	1	0	0	0	0	0

Concept usefull to represent signals and system, e.g.

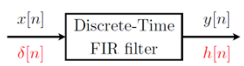
$$x[n] = 2\delta[n] + 3\delta[n-2] - \delta[n-4]$$



$$x[n] = \sum_k x[k]\delta[n-k] = \cdots + x[-1]\delta[n+1] + x[0]\delta[0] + x[1]\delta[n-1] + \cdots$$

Unit Impulse Response Sequence

Response FIR for $x[n] = \delta[n]$



Impulse response $h[n]$ of order M FIR:

$$h[n] = \sum_{k=0}^{M} b_k \delta[n-k] = \begin{cases} b_n & n = 0, 1, \cdots, M \\ 0 & \text{otherwise} \end{cases}$$

In tabular form response to FIR (Finite Impulse Response):

n	$n<0$	0	1	2	3	$\cdots$	M	$M+1$	$n>M$
$x[n]=\delta[n]$	0	1	0	0	0	0	0	0	0
$y[n]=h[n]$	0	b_0	b_1	b_2	b_3	$\cdots$	b_M	0	0

Unit Delay System

Delay or shift input $x[n]$ by n_0 samples, thus $y[n] = x[n - n_0]$

For order M FIR:

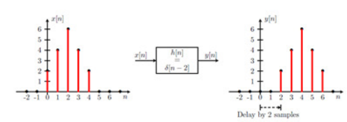
$$y[n] = b_0 x[n] + b_1 x[n-1] + \cdots b_{n_0} x[n-n_0] + \cdots + b_M x[n-M]$$

Now $y[n] = x[n-n_0]$ when $b_i = 0$ except for $i = n_0$, thus $b_{n_0} = 1$

$\Rightarrow$ Impulse response of delay by n_0 samples: $h[n] = \delta[n - n_0]$

Example:

Delay by 2 samples, thus $h[n] = \delta[n-2]$ and $y[n] = x[n-2]$



Convolution and FIR Filters

With impulse response $h[n]$ of order M FIR:

$$h[n] = \sum_{k=0}^{M} b_k \delta[n-k] = \begin{cases} b_n & n = 0, 1, \cdots, M \\ 0 & \text{otherwise} \end{cases}$$

the FIR output $y[n] = \sum_{k=0}^{M} b_k x[n-k]$ writes as Convolution sum:

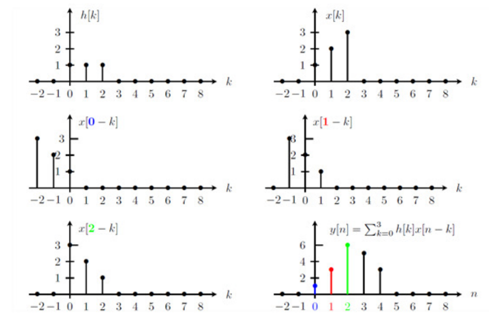
$$\begin{aligned} y[n] &= \sum_{k=0}^{M} h[k]x[n-k] \\ &= h[0]x[n] + h[1]x[n-1] + \cdots + h[M-1]x[n-M+1] \end{aligned}$$

Convolution is fundamental input-output algorithm for large class of filters (including FIR)

Later we will show that convolution also holds for infinite-lengths:

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k]$$

Convolution procedure via plot



Note for FIR : Length(y) = Length(x) + Length(h) -1

Convolution procedure via table

Convolve $x[n] = \{1, 2, 3\}$ with $h[n] = \{1, 1, 1\}$

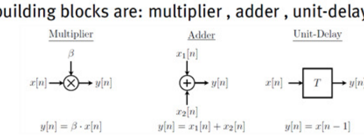
n	$n<0$	0	1	2	3	4	5	6	7	$n>7$
$x[n]$	0	1	2	3	0	0	0	0	0	0
$h[n]$	0	1	1	1						
$h[0] \cdot x[n]$	0	1	2	3	0	0	0	0	0	0
$h[1] \cdot x[n-1]$	0	0	1	2	3	0	0	0	0	0
$h[2] \cdot x[n-2]$	0	0	0	1	2	3	0	0	0	0
$y[n]$	0	1	3	6	5	3	0	0	0	0

In Matlab use command conv

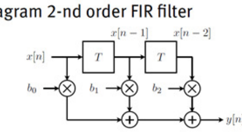
FIR Filter basic building blocks implem. –(1)

Recall FIR output writes as: $y[n] = \sum_{k=0}^{M} b_k x[n-k]$

⇒ Basic building blocks are: multiplier , adder , unit-delay operator



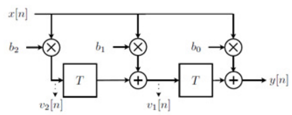
Example: Block diagram 2-nd order FIR filter



Structure shows why FIR also called feed-forward difference equation

FIR Filter basic building blocks implem. – (2)

How derive difference equation from block diagram?



$$\begin{aligned} y[n] &= b_0 x[n] + v_1[n-1] \\ v_1[n] &= b_1 x[n] + v_2[n-1] \\ v_2[n] &= b_2 x[n] \end{aligned}$$

$$\Rightarrow \quad v_1[n] = b_1 x[n] + b_2 x[n-1]$$
$$\Rightarrow \quad y[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2]$$

Same result as previous structure → **Transpose** system

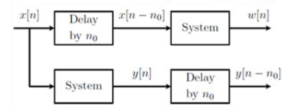
Implementation issues:
Constraints with VLSI or DSP architectures; Finite word-length effects; Memory length; etc

Linear Time Invariant (LTI) Systems

Why LTI: Simplified mathematics and leads to greater insight

Input-output notation: $x[n] \mapsto y[n]$

Time Invariance: $x[n-n_0] \mapsto y[n-n_0]$



System Time-Invariant
$w[n] = y[n-n_0]$

Example: $y[n] = (x[n])^2$

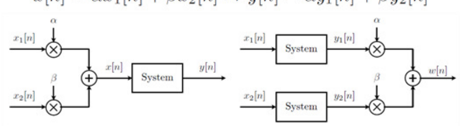
$x[n] \mapsto y[n] = (x[n])^2 \Rightarrow$ Delay by one $\Rightarrow y[n-1] = (x[n-1])^2$
$x[n] \Rightarrow$ Delay by one $\Rightarrow x[n-1] \mapsto (x[n-1])^2 = y[n-1]$

Not time-invariant: $y[n] = x[-n]$

Linearity property

Linear system: If $x_1[n] \mapsto y_1[n]$ and $x_2[n] \mapsto y_2[n]$, then

$$x[n] = \alpha x_1[n] + \beta x_2[n] \mapsto y[n] = \alpha y_1[n] + \beta y_2[n]$$



System Linear iff $w[n] = y[n]$

Example: System $y[n] = (x[n])^2$ is Nonlinear since

$$y[n] = (\alpha x_1[n] + \beta x_2[n])^2 \neq \alpha (x_1[n])^2 + \beta (x_2[n])^2$$

Is $y[n] = x[-n]$ linear?

FIR Filter is an LTI

FIR both linear and time-invariant, proof:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \quad \Rightarrow \quad y[n-n_0] = \sum_{k=0}^{M} b_k x[(n-n_0)-k]$$

On the other hand with $v[n] = x[n-n_0]$

$$w[n] = \sum_{k=0}^{M} b_k v[n-k] = \sum_{k=0}^{M} b_k x[(n-k)-n_0] = \sum_{k=0}^{M} b_k x[(n-n_0)-k]$$

FIR is Linear Time-Invariant (LTI) system

Main issues of LTI system:

- Impulse response is complete characterization
- Convolution is general formula to compute output from input

Convolution and the LTI system

Recall that we can write $x[n] = \sum_l x[l]\delta[n-l]$

$$(\cdots + x[-2]\delta[n+2] + x[-1]\delta[n+1] + x[0]\delta[n] + x[1]\delta[n-1] + x[2]\delta[n-2] + \cdots)$$

For LTI system:

$$\begin{aligned} \delta[n] \mapsto h[n] \quad &\Rightarrow \quad \delta[n-l] \mapsto h[n-l] \quad \text{for any } l \\ &\Rightarrow \quad x[l]\delta[n-l] \mapsto x[l]h[n-l] \quad \text{for any } l \\ &\Rightarrow \quad x[n] = \sum_l x[l]\delta[n-l] \mapsto y[n] = \sum_l x[l]h[n-l] \end{aligned}$$

No assumptions made about length $x[n]$ or $h[n]$, $\Rightarrow$

The convolution sum formula: $$y[n] = \sum_{l=-\infty}^{\infty} x[l]h[n-l]$$

Thus all LTI systems can be represented by a convolution sum

FIR Filter Convolution

Impulse response $h[n]$ of FIR only nonzero for $0 \leq n \leq M$. Thus

$$y[n] = \sum_{l=-\infty}^{\infty} x[l]h[n-l]$$

with index $h[n-l] \in \{0, M\}$ results in

$$y[n] = \sum_{l=n-M}^{n} x[l]h[n-l]$$

Simple example: Convolution of two blocks

See John Hopkins University site:

http://www.jhu.edu/signals/ ("Joy of convolution (discrete time)")

Some important properties LTI system - (1)

Convolution as operator:

$$y[n] = x[n]*h[n] = \sum_{l=-\infty}^{\infty} x[l]h[n-l]$$

Convolution with shifted impulse: $\Leftrightarrow$ Delay with n_0

$$x[n] * \delta[n-n_0] = x[n-n_0]$$

Commutative property of $*$:

$$x[n] * h[n] = h[n] * x[n]$$

Proof:

$$\begin{aligned} y[n] &= x[n] * h[n] = \sum_{l=-\infty}^{\infty} x[l]h[n-l] \quad \text{with} \quad k = n-l \\ &= \sum_{k=+\infty}^{-\infty} x[n-k]h[k] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] = h[n] * x[n] \end{aligned}$$

Some important properties LTI system – (2)

Associative property:

$$(x_1[n] * x_2[n]) * x_3[n] = x_1[n] * (x_2[n] * x_3[n])$$

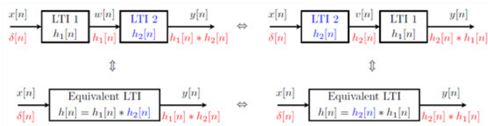
Proof:

$$x_1[n] * (x_2[n] * x_3[n]) = \sum_{l=-\infty}^{\infty} x_1[l] \left(\sum_{k=-\infty}^{\infty} x_2[k]x_3[(n-l)-k] \right)$$

with $k = q - l \Rightarrow$ $$= \sum_{l=-\infty}^{\infty} x_1[l] \sum_{q=-\infty}^{\infty} x_2[q-l]x_3[n-q]$$

$$= \sum_{q=-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} x_1[l]x_2[q-l] \right) x_3[n-q] = (x_1[n] * x_2[n]) * x_3[n]$$

Cascade of LTI systems – (1)



$$
\begin{aligned}
y[n] &= (x[n] * h_1[n]) * h_2[n] = x[n] * (h_1[n] * h_2[n]) \\
&= x[n] * (h_2[n] * h_1[n]) = (x[n] * h_2[n]) * h_1[n]
\end{aligned}
$$

Example: Evaluate $h[n] = h_1[n] * h_2[n]$ with

$$h_1[n] = \begin{cases} 1 & 0 \leq n \leq 3 \\ 0 & \text{otherwise} \end{cases} \qquad h_2[n] = \begin{cases} 1 & 1 \leq n \leq 3 \\ 0 & \text{otherwise} \end{cases}$$

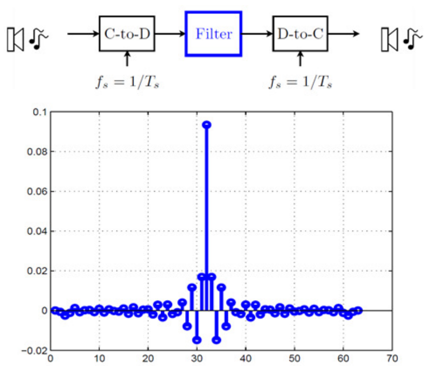
Cascade of LTI systems – (2)

n	$n<0$	0	1	2	3	4	5	6	$n>6$
$h_1[n]$	0	1	1	1	1	0	0	0	0
$h_2[n]$	0	0	1	1	1				
$h_2[0]h_1[n]$	0	0	0	0	0	0	0	0	0
$h_2[1]h_1[n-1]$	0	0	1	1	1	1	0	0	0
$h_2[2]h_1[n-2]$	0	0	0	1	1	1	1	0	0
$h_2[3]h_1[n-3]$	0	0	0	0	1	1	1	1	0
$h[n]$	0	0	1	2	3	3	2	1	0

Equivalent impulse response:
$h[n] = \sum_{k=0}^{6} b_k \delta[n-k]$ with b_k the sequence $\{0, 1, 2, 3, 3, 2, 1\}$

What would be preferred from implementation point of view?
$y[n] = \sum_{k=0}^{6} b_k x[n-k]$ or
$y[n] = \sum_{k=1}^{3} w[n-k]$ with $w[n] = \sum_{k=0}^{3} x[n-k]$

Example FIR filtering

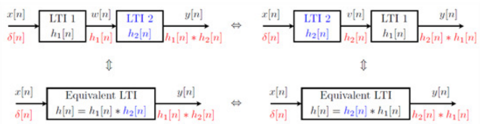


FIR filters summary – (1)

- Finite Impulse Response (FIR) filters: Each output sample is sum of finite number of weighted samples of input sequence.
- Difference equation FIR: $y[n] = \sum_{k=0}^{M} b_k x[n-k]$
  Weighted running average of $M+1$ samples; Causal filter; M is order of FIR filter; $L = M+1$ is length of FIR filter
- Unit Impulse Sequence: $\delta[n] = 1$ for $n = 0$ and zero elsewhere.
  Representation signal by unit impulses: $x[n] = \sum_k x[k]\delta[n-k]$
- Impulse response $h[n]$ of order M FIR: $h[n] = \sum_{k=0}^{M} b_k \delta[n-k] = b_n$
- Time-Invariance: If $x[n] \mapsto y[n]$ then $x[n-n_0] \mapsto y[n-n_0]$
- Linear system: If $x_1[n] \mapsto y_1[n]$ and $x_2[n] \mapsto y_2[n]$ then
  $x[n] = \alpha x_1[n] + \beta x_2[n] \mapsto y[n] = \alpha y_1[n] + \beta y_2[n]$

FIR filters summary – (2)

- Main issues Linear Time-Invariance (LTI) system: (FIR is LTI)
  Completely characterized by impulse response; Convolution is general formula to compute output from input.
- Convolution sum FIR: $y[n] = \sum_{k=0}^{M} h[k]x[n-k]$
- Convolution operator: $y[n] = x[n]*h[n] = \sum_{l=-\infty}^{\infty} x[l]h[n-l]$
  - Commutative property: $x[n]*h[n] = h[n]*x[n]$
  - Associative property: $(x_1[n]*x_2[n])*x_3[n] = x_1[n]*(x_2[n]*x_3[n])$
- Cascade of LTI systems:



Module 02 – Part 4 Introduction to signal transformation

Rotation transformation, generalization to orthogonal transforms

Module 02 – Part 4 The meaning of a transformation

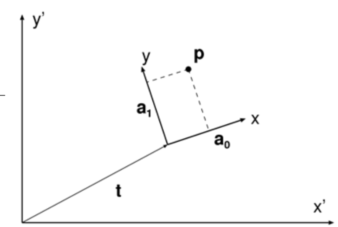
Affine transformation, basic principles, matrix notation

Affine 2D Coordinate Transform – (1)

- **Express coordinate frame** (x,y) **in coordinate frame** (x',y').
- **In motion estimation:**
  - (x,y) could be the coordinate in the last frame, and
  - (x',y') the corresponding coordinate in the current frame.

$$\mathbf{p} = (x \;\; y)^\top$$
$$\mathbf{p}' = (x' \;\; y')^\top$$

$$\mathbf{p}' = \mathbf{t} + x \cdot \mathbf{a}_0 + y \cdot \mathbf{a}_1$$
$$= (\mathbf{a}_0 \;\; \mathbf{a}_1)\,\mathbf{p} + \mathbf{t}$$
$$= \mathbf{A}\mathbf{p} + \mathbf{t}$$



Affine 2D Coordinate Transform – (2)

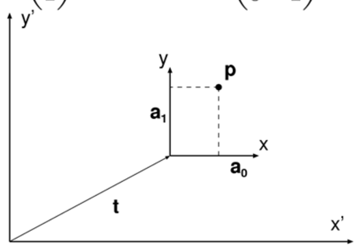
73

* **Special (simple) case 1: translation only**

$$\mathbf{a_0} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \mathbf{a_1} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \longrightarrow \mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

* **A is identity matrix $\mathbf{I}$**

$$\mathbf{p}' = \mathbf{A}\mathbf{p} + \mathbf{t} = \mathbf{p} + \mathbf{t}$$



Affine 2D Coordinate Transform – (3)

74

* **Special case 2: scaling only**
  - scaling in x direction

$$\mathbf{a_0} = \begin{pmatrix} s_x \\ 0 \end{pmatrix}$$

scaling in y direction

$$\mathbf{a_1} = \begin{pmatrix} 0 \\ s_y \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}$$

* **Isotropic scaling (rigid motion)**

$$s_x = s_y = s \qquad \mathbf{A} = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} \qquad \mathbf{p}' = \mathbf{A}\mathbf{p}$$

Affine 2D Coordinate Transform – (4)

75

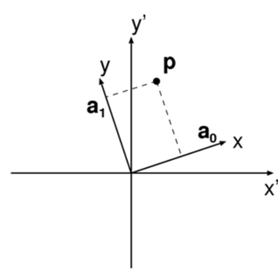
* **Special case 3: rotation only**
  - constraint 1: size stays constant

$$||\mathbf{a_0}|| = ||\mathbf{a_1}|| = 1$$

  - constraint 2: coordinate axes are perpendicular

$$\mathbf{a_0}^\top \cdot \mathbf{a_1} = 0$$

* **A is an orthonormal matrix (rotation matrix R)**



Affine 2D Coordinate Transform – (5)

76

* **Rotation matrix for rotation by an angle α**

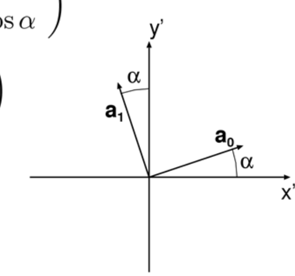
$$\mathbf{a_0} = \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \quad \mathbf{a_1} = \begin{pmatrix} -\sin\alpha \\ \cos\alpha \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}$$

* **rotation by -α**

$$\mathbf{A}^{-1} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$$

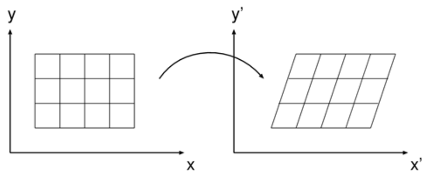
$$\mathbf{A}^{-1} = \mathbf{A}^\top$$



Affine 2D Coordinate Transform – (6)

* Image skew (horizontal)

$$\mathbf{A} = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}$$



$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{A} \begin{pmatrix} x \\ y \end{pmatrix}$$

Affine 2D Coordinate Transform – (7)

* Concatenation of transforms

$$\mathbf{p}' = \mathbf{A}_1\mathbf{p} + \mathbf{t}_1$$

$$\mathbf{p}'' = \mathbf{A}_2\mathbf{p}' + \mathbf{t}_2$$

* gives $$\mathbf{p}'' = \underbrace{\mathbf{A}_2\mathbf{A}_1}_{\mathbf{A}_3}\mathbf{p} + \underbrace{\mathbf{A}_2\mathbf{t}_1 + \mathbf{t}_2}_{\mathbf{t}_3}$$

(Shortly we will introduce a trick to make this much easier)

Affine 2D Coordinate Transform – (8)

* Computing the inverse of

$$\mathbf{p}' = \mathbf{A}\mathbf{p} + \mathbf{t}$$

* gives

$$\mathbf{p} = \mathbf{A}^{-1}\mathbf{p}' - \mathbf{A}^{-1}\mathbf{t}$$

(the mathematical trick will make this much easier, too)

* **Inverse is possible if A is nonsingular.**

Set of nonsingular affine transforms constitutes a ***group***.

Affine 2D Coordinate Transform – (9)

* Here the trick: augment each point-coordinate with a dummy '1':

$$\mathbf{p} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

* Now, write affine transform as simple matrix multiplication:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{00} & a_{01} & t_x \\ a_{10} & a_{11} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\mathbf{p}' = \mathbf{A}\mathbf{p} + \mathbf{t}$$

Affine 2D Coordinate Transform – (10)

81

- Concatenation is now very easy

$$\mathbf{A_3} = \mathbf{A_2 A_1}$$ (A_i are 3x3 matrices)

- And the inverse transform is simply the 3x3 matrix inverse.

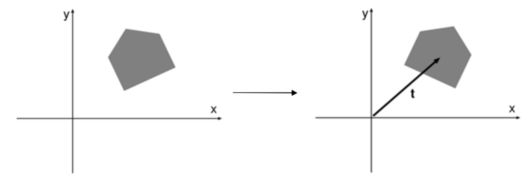
$$\mathbf{p}' = \mathbf{A}\mathbf{p} \qquad \mathbf{p} = \mathbf{A}^{-1}\mathbf{p}$$

- This useful "trick" will reappear later...

Affine 2D Coordinate Transform
Example: Transform Sequence – (1)

82

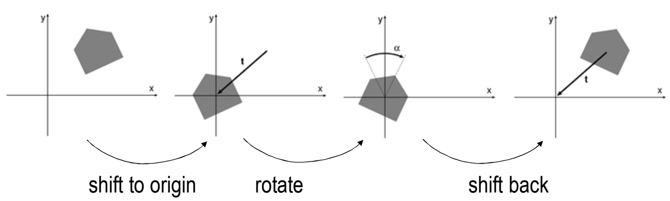
- What is the transformation between these two images?



Affine 2D Coordinate Transform
Example: Transform Sequence – (2)

83

- Break the transformation into sequence of elementary steps.



$$\mathbf{A_1} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{A_2} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{A_3} = \begin{pmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{pmatrix}$$

Affine 2D Coordinate Transform
Example: Transform Sequence – (3)

84

- Successive steps:

$$\mathbf{A_1} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{A_2} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{A_3} = \begin{pmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{pmatrix}$$

- Combined transform:

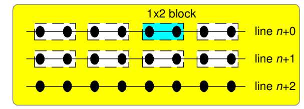
$$\mathbf{p}' = \mathbf{A_3 A_2 A_1 p}$$

- Multiplied together: $\begin{pmatrix} t_x\cos\alpha & -t_y\sin\alpha & -t_x \\ t_x\sin\alpha & t_y\cos\alpha & -t_y \\ 0 & 0 & 1 \end{pmatrix}$

Basic principles of transformation
A. Meaning of transform

Transform / Basic principles – (1)

* **Consider the simplest transform of 1x2 samples**

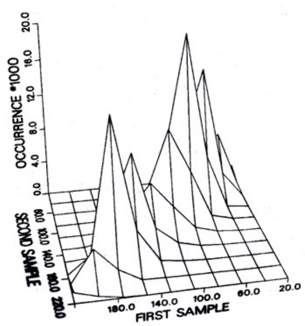


– Rewrite sample combination *(x0, x1)* as:
(x0, x1) = x0.(1, 0) + *x1*.(0, 1)
– Sample sequence *(x0, x1)* can be *approximated* by
(x0, x1) „= +/-“((x0+x1)/2 , (x0+x1)/2)
– The sample sequence *(x0, x1)* is *exactly* described by taking the approximation and adding a detailed modification
$(x0, x1) = (x0+x1)/2\ (1, 1)^T + (-x0+x1)/2\ (-1, 1)^T$

Transform / Basic principles – (2)

Example 1x2 blocks:
„Car1.y“ sequence,
f = 13.5 MHz

Note that
* **all grey values occur**
* **extends over both dimensions**



Transform / Basic principles – (3)

Linear algebra

* **Rewrite the sample combination (x0, x1) in vector form**

$$\begin{pmatrix} x0 \\ x1 \end{pmatrix} = x0.\begin{pmatrix} 1 \\ 0 \end{pmatrix} + x1.\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

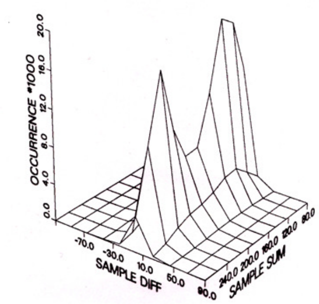
* **Or, alternatively**

$$\begin{pmatrix} x0 \\ x1 \end{pmatrix} = \frac{x0+x1}{\sqrt{2}.\sqrt{2}}.\begin{pmatrix} 1 \\ 1 \end{pmatrix} + \frac{-x0+x1}{\sqrt{2}.\sqrt{2}}.\begin{pmatrix} -1 \\ 1 \end{pmatrix} = \frac{x0+x1}{\sqrt{2}}\underline{b}0 + \frac{-x0+x1}{\sqrt{2}}.\underline{b}1$$

* **The sample sequence (x0, x1) is a linear combination with**
  – $(x0+x1)/\sqrt{2}$ as the weight or low-frequency transform coefficient
  – $(-x0+x1)/\sqrt{2}$ as the weight or high-frequency transform coefficient
  – the vectors $\underline{b}0$ and $\underline{b}1$ are the basis vectors

Transform / Basic principles – (4)

Example 1x2 blocks:
„Car1.y" sequence, f = 13.5 MHz, based on ***sample sum and difference***

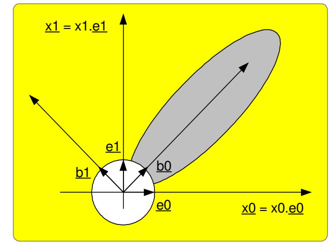


Note that
- **same landscape occurs**
- **extends over sample sum only!**
- **Concentration to one dimension**

Transform / Basic principles – (5)

A geometric interpretation shows rotation

- **($\underline{e}0$, $\underline{e}1$) and ($\underline{b}0$, $\underline{b}1$) are orthogonal vectors**
- **these vectors have normalized length (norm)**
- **($\underline{b}0$, $\underline{b}1$) are along axis (projected) $x0$=$x1$ and axis $x0$= -- $x1$**
- **Thus: a *rotated* coordinate system**



Transform / Basic principles – (6)

Results of geometric interpretation

- **Transformation is rotation**
  - Rotation to a different coodinate system for the input samples. The amount of samples gives the amount of dimensions.
- **Transformation concentrates energy**
  - Coordinates are chosen such that global approximations are found in the total signal and additions that make the description exact.
- **Transformation to perform ... decorrelation**
  - Statistical dependence occurs between adjacent samples. This is exploited by global approximation. New coordinates are decorrelated.

Basic principles of transformation
B. Fundamentals in matrix notation

Transform / Basic principles – (7)

Matrix notation

* **Rewrite in vector / matrix form**

$$\underline{y}=\begin{pmatrix}y0\\y1\end{pmatrix}=\frac{x0+x1}{\sqrt{2}}.\underline{b}0+\frac{-x0+x1}{\sqrt{2}}.\underline{b}1=\frac{1}{\sqrt{2}}.\begin{bmatrix}+1&+1\\-1&+1\end{bmatrix}.\begin{pmatrix}x0\\x1\end{pmatrix}=A.\underline{x}$$

* **Hence, $\underline{y}$ = A.$\underline{x}$, and similarly, for the backward transform**

$$\underline{x_r}=\begin{pmatrix}x_r0\\x_r1\end{pmatrix}=\frac{1}{\sqrt{2}}.\begin{bmatrix}+1&-1\\+1&+1\end{bmatrix}.\begin{pmatrix}y0\\y1\end{pmatrix}=A^{-1}.\underline{y}$$

* **Some properties**
  - Matrices **A** and **A**$^{-1}$ are the *transform* and *inverse transform matrix*. Mostly, they are not equal and they are orthogonal.
  - Basis vectors $\underline{b}0$ and $\underline{b}1$ are the rows in the transform matrix
  - Reconstructed samples x_r0 and x_r1 are linear combination of the basis vectors.
  - The weight of each vector is the *transform coefficient*

Transform / Basic principles – (8)

Matrix notation for generalized 2-D transforms

* **In matrix notation**

$$y(u,v)=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}x(i,j).A(u,v,i,j)=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}x(i,j).Av(u,i).Ah(v,j)$$

* **Perform 2-D transform as two times a 1-D transform, thus**

$$y(u,v)=\sum_{i=0}^{N-1}\{\sum_{j=0}^{N-1}x(i,j).Av(u,i).Ah(v,j)\}=\sum_{i=0}^{N-1}Av(u,i).\{\sum_{j=0}^{N-1}x(i,j).Ah(v,j)\}$$

* **Geometric meaning is: the reconstructed sample data is a linear combination of basis images B_{uv}, being an outer product of basis vectors**

$$x_r(i,j)=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}y(u,v).\underline{\underline{b}}_{uv}\ldots with.\underline{\underline{b}}_{uv}=\underline{b}_u\otimes\underline{b}_v^T$$

Example: 2-D Discrete Fourier Transform

* **2-D Discrete Fourier Transform** of image signal $f(x, y)$ for $0<=u<=M$-1 and $0<=v<=N$-1 is

$$F(u,v)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

* 2. **2-D Inverse Discrete Fourier Transform**

$$f(x,y)=\frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)e^{+j2\pi(ux/M+vy/N)}$$

* 3. The above equations hold for an image of size MxN pixels and together they form a transform pair