ECE 201: Introduction to Signal Analysis

Prof. Paris

Last updated: October 31, 2007

Part I

Sampling of Signals

Lecture: Introduction to Sampling

Sampling and Discrete-Time Signals

- MATLAB, and other digital processing systems, can not process continuous-time signals.
- Instead, MATLAB requires the continuous-time signal to be converted into a discrete-time signal.
- The conversion process is called sampling.
- To sample a continuous-time signal, we evaluate it at a discrete set of times $t_n = nT_s$, where
  - n is a integer,
  - T_s is called the sampling period (time between samples),
  - $f_s = 1/T_s$ is the sampling rate (samples per second).

Sampling and Discrete-Time Signals

- Sampling results in a sequence of samples

$$x(nT_s) = A \cdot \cos(2\pi fnT_s + \phi).$$

- Note that the independent variable is now n, not t.
- To emphasize that this is a discrete-time signal, we write

$$x[n] = A \cdot \cos(2\pi fnT_s + \phi).$$

- Sampling is a straightforward operation.
- We will see that the sampling rate f_s must be chosen with care!

Sampled Signals in MATLAB

- Note that we have worked with sampled signals whenever we have used MATLAB.
- For example, we use the following MATLAB fragment to generate a sinusoidal signal:

```
fs = 100;
tt = 0:1/fs:3;
xx = 5*cos(2*pi*2*tt + pi/4);
```

- The resulting signal `xx` is a discrete-time signal:
  - The vector `xx` contains the samples, and
  - the vector `tt` specifies the sampling instances: $0, 1/f_s, 2/f_s, \ldots, 3$.
- We will now turn our attention to the impact of the sampling rate f_s.

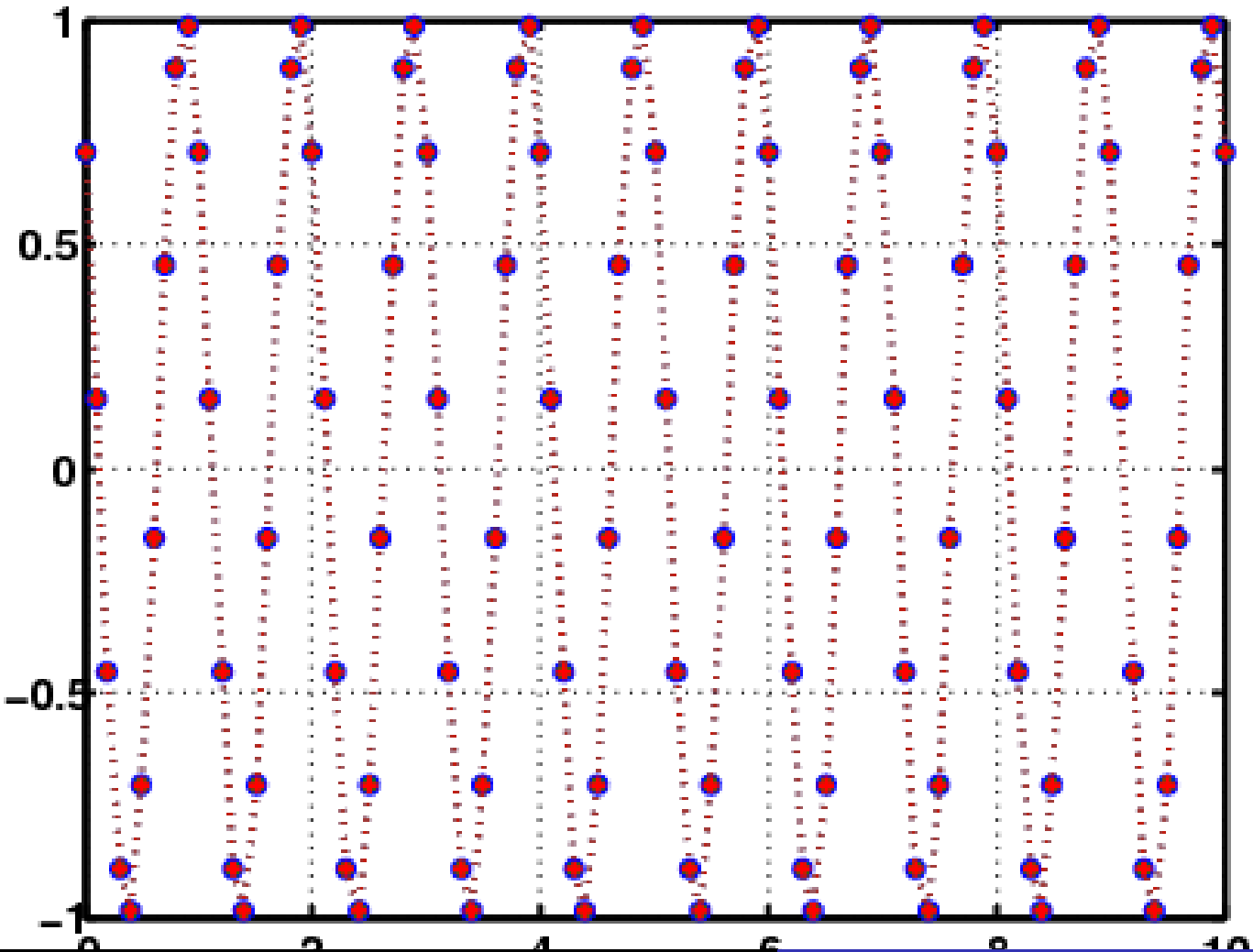
Example: Three Sinuoids

- **Objective:** In MATLAB, compute sampled versions of three sinusoids:
  1. $x(t) = \cos(2\pi t + \pi/4)$
  2. $x(t) = \cos(2\pi 9t - \pi/4)$
  3. $x(t) = \cos(2\pi 11t + \pi/4)$
- The sampling rate for all three signals is $f_s = 10$.

MATLAB code

```
% SamplingDemo - Sample three sinusoidal signals to demonstrate the impact
% of sampling

fs  = 10;
dur = 10;

tt  = 0:1/fs:dur;
xx1 = cos(2*pi*tt+pi/4);
xx2 = cos(2*pi*9*tt-pi/4);
xx3 = cos(2*pi*11*tt+pi/4);

plot(tt,xx1,':o',tt,xx2,':x',tt,xx3,':+');
grid
xlabel('Time (s)')
```

Resulting Plot



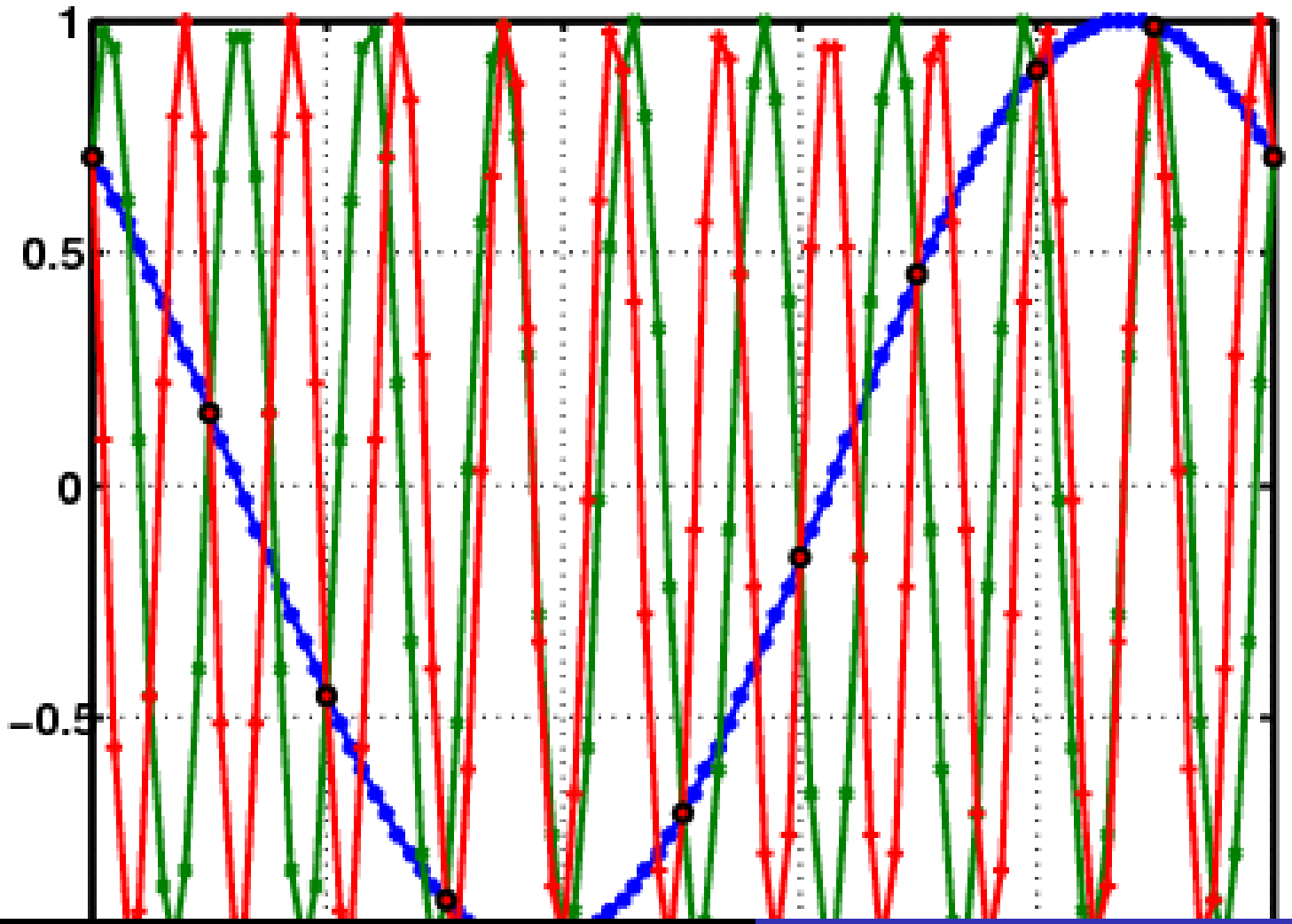
What happened?

- The samples for all three signals are identical: how is that possible?
- Is there a “bug” in the MATLAB code?
  - No, the code is correct.
- **Suspicion:** The problem is related to our choice of sampling rate.
  - To test this suspicion, repeat the experiment with a different sampling rate.
  - We also reduce the duration to keep the number of samples constant - that keeps the plots reasonable.

MATLAB code

```
% SamplingDemoHigh - Sample three sinusoidal signals to demonstrate the impact
% of sampling

fs  = 100;
dur = 1;

tt  = 0:1/fs:dur;
xx1 = cos(2*pi*tt+pi/4);
xx2 = cos(2*pi*9*tt-pi/4);
xx3 = cos(2*pi*11*tt+pi/4);

plot(tt,xx1,'-*',tt,xx2,'-x',tt,xx3,'-+');
hold on
plot(tt(1:10:end), xx1(1:10:end),'ok');
grid
xlabel('Time (s)')

hold off
```

Resulting Plot



The Influence of the Sampling Rate

- Now the three sinusoids are clearly distinguishable and lead to different samples.
- Since the only parameter we changed is the sampling rate f_s, it must be responsible for the ambiguity in the first plot.
- Notice also that every 10-th sample (marked with a black circle) is identical for all three sinusoids.
  - Since the sampling rate was 10 times higher for the second plot, this explains the first plot.
- It is useful to investigate the effect of sampling mathematically, to understand better what impact it has.

Sampling a Sinusoidal Signal

- A continuous-time sinusoid is given by

$$x(t) = A\cos(2\pi ft + \phi).$$

- When this signal is sampled at rate f_s, we obtain the discrete-time signal

$$x[n] = A\cos(2\pi fn/f_s + \phi).$$

- It is useful to define the normalized frequency $f_d = \frac{f}{f_s}$, so that

$$x[n] = A\cos(2\pi f_d n + \phi).$$

Three Cases

- We will distinguish between three cases:
  1. $0 \leq f_d \leq 1/2$ (Oversampling)
  2. $1/2 < f_d \leq 1$ (Undersampling, folding)
  3. $1 < f_d \leq 3/2$ (Undersampling, aliasing)
- This captures the three situations addressed by the first example:
  1. $f = 1, f_s = 10 \Rightarrow f_d = 1/10$
  2. $f = 9, f_s = 10 \Rightarrow f_d = 9/10$
  3. $f = 11, f_s = 10 \Rightarrow f_d = 11/10$
- We will see that all three cases lead to identical samples.

Oversampling

- When the sampling rate is such that $0 \leq f_d \leq 1/2$, then the samples of the sinusoidal signal are given by

$$x[n] = A\cos(2\pi f_d n + \phi).$$

- This cannot be simplified further.
- It provides our base-line.
- Oversampling is the desired behaviour!

Undersampling, Aliasing

- When the sampling rate is such that $1 < f_d \leq 3/2$, then we define the apparent frequency $f_a = f_d - 1$.
- Notice that $0 < f_a \leq 1/2$.
  - For $f = 11, f_s = 10 \Rightarrow f_d = 11/10 \Rightarrow f_a = 1/10$.
- The samples of the sinusoidal signal are given by

$$x[n] = A\cos(2\pi f_d n + \phi) = A\cos(2\pi(1 + f_a)n + \phi).$$

- Expanding the terms inside the cosine,

$$x[n] = A\cos(2\pi f_a n + 2\pi n + \phi) = A\cos(2\pi f_a n + \phi)$$

  - **Interpretation:** The samples are identical to those from a sinusoid with frequency $f_a \cdot f_s$ and phase ϕ.

Undersampling, Folding

- When the sampling rate is such that $1/2 < f_d \leq 1$, then we introduce the apparent frequency $f_a = 1 - f_d$; again $0 < f_a \leq 1/2$.
  - For $f = 9, f_s = 10 \Rightarrow f_d = 9/10 \Rightarrow f_a = 1/10$.
- The samples of the sinusoidal signal are given by

$$x[n] = A\cos(2\pi f_d n + \phi) = A\cos(2\pi(1 - f_a)n + \phi).$$

- Expanding the terms inside the cosine,

$$x[n] = A\cos(-2\pi f_a n + 2\pi n + \phi) = A\cos(-2\pi f_a n + \phi)$$

- Because of the symmetry of the cosine ($\cos(x) = \cos(-x)$) this equals

$$x[n] = A\cos(2\pi f_a n - \phi).$$

Sampling Higher-Frequency Sinusoids

- For sinusoids of even higher frequencies f, either folding or aliasing occurs.
- As before, let f_d be the normalized frequency f/f_s.
- Decompose f_d into an integer part N and fractional part f_p.
  - **Example:** If f_d is 5.7 then N equals 5 and f_p is 0.7.
  - Notice that $0 \leq f_p < 1$, always.
- **Phase Reversal** occurs when the phase of the sampled sinusoid is the negative of the phase of the continuous-time sinusoid.
- We distinguish between
  - **Folding** occurs when $f_p > 1/2$. Then the apparent frequency f_a equals $1 - f_p$ and phase reversal occurs.
  - **Aliasing** occurs when $f_p \leq 1/2$. Then the apparent frequency is $f_a = f_p$; no phase reversal occurs.

Examples

- For the three sinusoids considered earlier:
  1. $f = 1, \phi = \pi/4, f_s = 10 \Rightarrow f_d = 1/10$
  2. $f = 9, \phi = -\pi/4, f_s = 10 \Rightarrow f_d = 9/10$
  3. $f = 11, \phi = \pi/4, f_s = 10 \Rightarrow f_d = 11/10$
- The first case, represents oversampling: The apparent frequency $f_a = f_d$ and no phase reversal occurs.
- The second case, represents folding: The apparent f_a equals $1 - f_d$ and phase reversal occurs.
- In the final example, the fractional part of $f_d = 1/10$. Hence, this case represents alising; no phase reversal occurs.

Exercise

The discrete-time sinusoidal signal

$$x[n] = 5\cos(2\pi 0.2n - \frac{\pi}{4}).$$

was obtained by sampling a continuous-time sinusoid of the form

$$x(t) = A\cos(2\pi ft + \phi)$$

at the sampling rate $f_s = 8000\ Hz$.

1. Provide three different sets of paramters A, f, and ϕ for the continuous-time sinusoid that all yield the discrete-time sinusoid above when sampled at the indicated rate. The parameter f must satisfy $0 < f < 12000\ Hz$ in all three cases.
2. For each case indicate if the signal is undersampled or

Lecture: The Sampling Theorem

The Sampling Theorem

- We have analyzed the relationship between the frequency f of a sinusoid and the sampling rate f_s.
  - We saw that the ratio f/f_s must be less than $1/2$, i.e., $f_s > 2 \cdot f$.
  - Otherwise aliasing or folding occurs.
- This insight provides the first half of the famous sampling theorem

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

- This very import result is attributed to Claude Shannon and

Reconstructing a Signal from Samples

- The sampling theorem suggests that the original continuous-time signal $x(t)$ can be recreated from its samples $x[n]$.
  - Assuming that samples were taken at a high enough rate.
  - This process is referred to as reconstruction or D-to-C conversion.
- In principle, the continous-time signal is reconstructed by placing a suitable pulse at each sample location and adding all pulses.
  - The amplitude of each pulse is given by the sample value.

Suitable Pulses

- Suitable pulses include
  - Rectangular pulse (zero-order hold):

$$p(t) = \begin{cases} 1 & \text{for } -T_s/2 \leq t < T_s/2 \\ 0 & \text{else.} \end{cases}$$

  - Triangular pulse (linear interpolation)

$$p(t) = \begin{cases} 1 + t/T_s & \text{for } -T_s \leq t \leq 0 \\ 1 - t/T_s & \text{for } 0 \leq t \leq T_s \\ 0 & \text{else.} \end{cases}$$

Reconstruction

- The reconstructed signal $\hat{x}(t)$ is computed from the samples and the pulse $p(t)$:

$$\hat{x}(t) = \sum_{n=-\infty}^{\infty} x[n] \cdot p(t - nT_s).$$

- The reconstruction formula says:
  - place a pulse at each sampling instant ($p(t - nT_s)$),
  - scale each pulse to amplitude $x[n]$,
  - add all pulses to obtain the reconstructed signal.

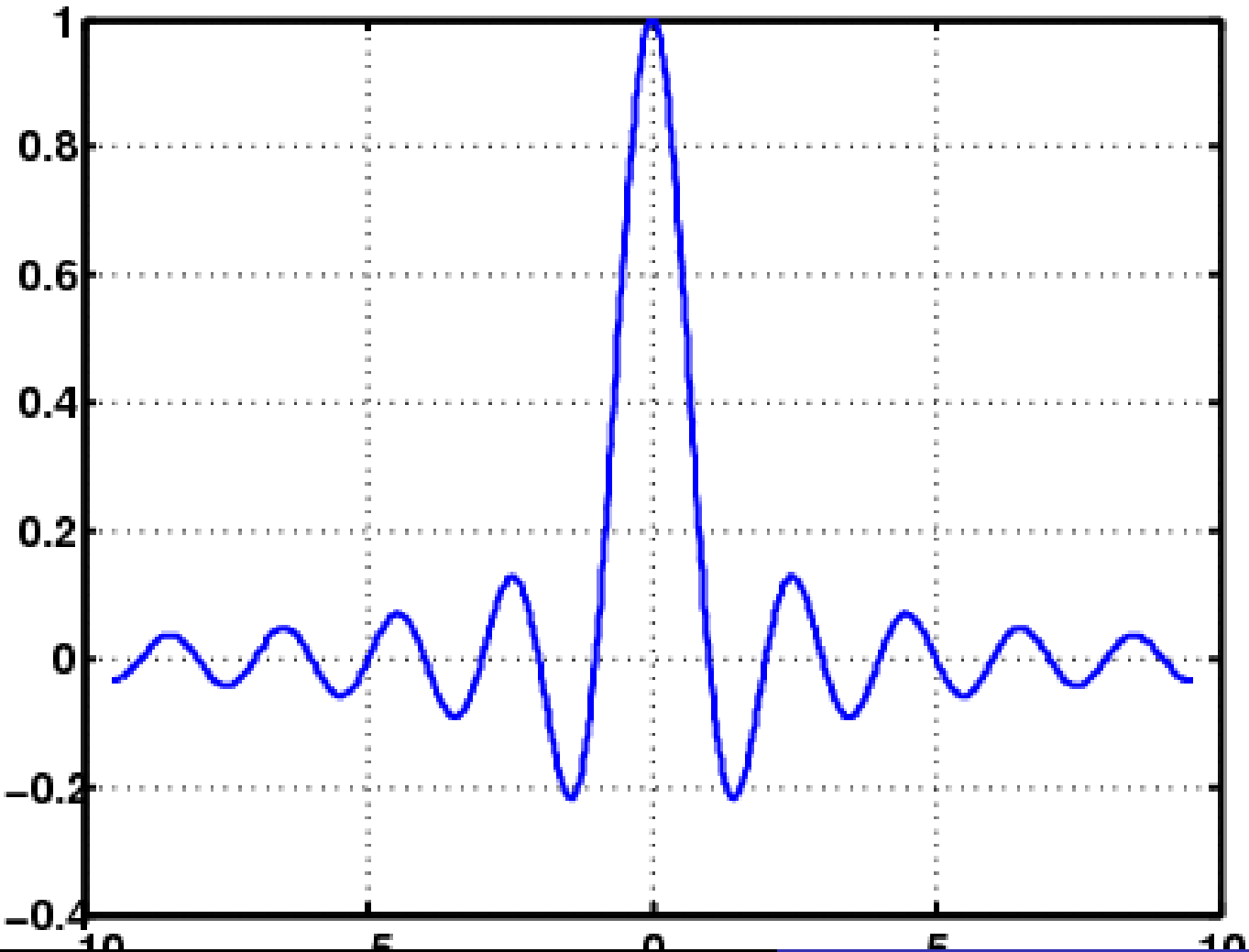
Ideal Reconstruction

- Reconstruction with the above pulses will be pretty good.
  - Particularly, when the sampling rate is much greater than twice the signal frequency (significant oversampling).
- However, reconstruction is not perfect as suggested by the sampling theorem.
- To obtain perfect reconstruction the following pulse must be used:

$$p(t) = \frac{\sin(\pi t/T_s)}{\pi t/T_s}.$$

- This pulse is called the sinc pulse.
- Note, that it is of infinite duration and, therefore, is not practical.
  - In practice a truncated version may be used for excellent reconstruction.

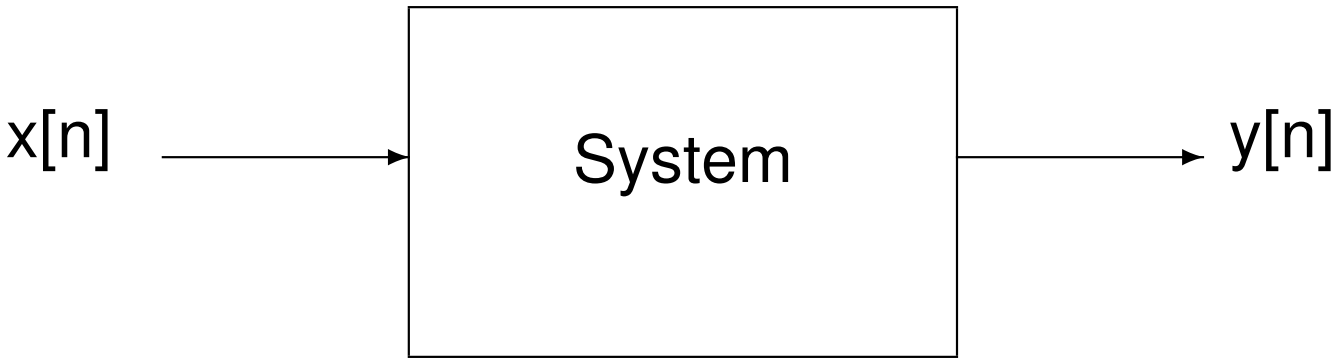
The sinc pulse



Lecture: Introduction to Systems and FIR filters

Systems

- A system is used to process an input signal $x[n]$ and produce the ouput signal $y[n]$.
  - We focus on discrete-time signals and systems;
  - a correspoding theory exists for continuous-time signals and systems.
- Many different systems:
  - Filters: remove undesired signal components,
  - Modulators and demodulators,
  - Detectors.



Representative Examples

- The following are examples of systems:
  - **Squarer:** $y[n] = (x[n])^2$;
  - **Modulator:** $y[n] = x[n] \cdot \cos(2\pi f_d n)$;
  - **Averager:** $y[n] = \frac{1}{M+1} \sum_{k=0}^{M} x[n-k]$;
  - **FIR Filter:** $y[n] = \sum_{k=0}^{M} b_k x[n-k]$

Squarer

- System relationship between input and output signals:

$$y[n] = (x[n])^2.$$

- **Example:** Input signal: $x[n] = \{1, 2, 3, 4, 3, 2, 1\}$
  - **Notation:** $x[n] = \{1, 2, 3, 4, 3, 2, 1\}$ means $x[0] = 1$, $x[1] = 2$, ..., $x[6] = 1$; all other $x[n] = 0$.
- Output signal: $y[n] = \{1, 4, 9, 16, 9, 4, 1\}$.

Modulator

- System relationship between input and output signals:

$$y[n] = (x[n]) \cdot \cos(2\pi f_d n);$$

  assume $f_d = 0.5$, i.e., $\cos(2\pi f_d n) = \{\ldots, 1, -1, 1, -1, \ldots\}$.
- **Example:** Input signal: $x[n] = \{1, 2, 3, 4, 3, 2, 1\}$
- Output signal: $y[n] = \{1, -2, 3, -4, 3, -2, 1\}$.

Averager

- System relationship between input and output signals:

$$\begin{aligned} y[n] &= \frac{1}{M+1}\sum_{k=0}^{M} x[n-k] \\ &= \frac{1}{M+1}(x[n]+x[n-1]+\ldots+x[n-M]) \\ &= \sum_{k=0}^{M}\frac{1}{M+1}x[n-k]. \end{aligned}$$

- This system computes the *sliding* average over the $M+1$ most recent samples.
- **Example:** Input signal: $x[n] = \{1, 2, 3, 4, 3, 2, 1\}$
- For computing the output signal, a table is very useful.
  - synthetic multiplication table.

3-Point Averager ($M = 2$)

n	-1	0	1	2	3	4	5	6	7	8
$x[n]$	0	1	2	3	4	3	2	1	0	0
$\frac{1}{M+1}x[n]$	0	$\frac{1}{3}$	$\frac{2}{3}$	1	$\frac{4}{3}$	1	$\frac{2}{3}$	$\frac{1}{3}$	0	0
$+\frac{1}{M+1}x[n-1]$	0	0	$\frac{1}{3}$	$\frac{2}{3}$	1	$\frac{4}{3}$	1	$\frac{2}{3}$	$\frac{1}{3}$	0
$+\frac{1}{M+1}x[n-2]$	0	0	0	$\frac{1}{3}$	$\frac{2}{3}$	1	$\frac{4}{3}$	1	$\frac{2}{3}$	$\frac{1}{3}$
$y[n]$	0	$\frac{1}{3}$	1	2	3	$\frac{10}{3}$	3	2	1	$\frac{1}{3}$

$$y[n] = \{0, \tfrac{1}{3}, 1, 2, 3, \tfrac{10}{3}, 3, 2, 1, \tfrac{1}{3}\}$$

General FIR Filter

- The M-point averager is a special case of the general FIR filter.
  - FIR stands for Finite Impulse Response; we will see what this means later.
- The system relationship between the input $x[n]$ and the output $y[n]$ is given by

$$y[n] = \sum_{k=0}^{M} b_k \cdot x[n-k].$$

General FIR Filter

- The filter coefficients b_k determine the characteristics of the filter.
- Clearly, with $b_k = \frac{1}{M+1}$ for $k = 0, 1, \ldots, M$ we obtain the M-point averager.
- Computation of the output signal can be done via a synthetic multiplication table.
  - **Example:** $x[n] = \{1, 2, 3, 4, 3, 2, 1\}$ and $b_k = \{1, -2, 1\}$.

FIR Filter ($b_k = \{1, -2, 1\}$)

n	-1	0	1	2	3	4	5	6	7	8
$x[n]$	0	1	2	3	4	3	2	1	0	0
$1 \cdot x[n]$	0	1	2	3	4	3	2	1	0	0
$-2 \cdot x[n-1]$	0	0	-2	-4	-6	-8	-6	-4	-2	0
$+1 \cdot x[n-2]$	0	0	0	1	2	3	4	3	2	1
$y[n]$	0	1	0	0	0	-2	0	0	0	1

$$y[n] = \{0, 1, 0, 0, 0, -2, 0, 0, 0, 1\}$$

Exercise

1. Find the output signal $y[n]$ for an FIR filter

$$y[n] = \sum_{k=0}^{M} b_k \cdot x[n-k]$$

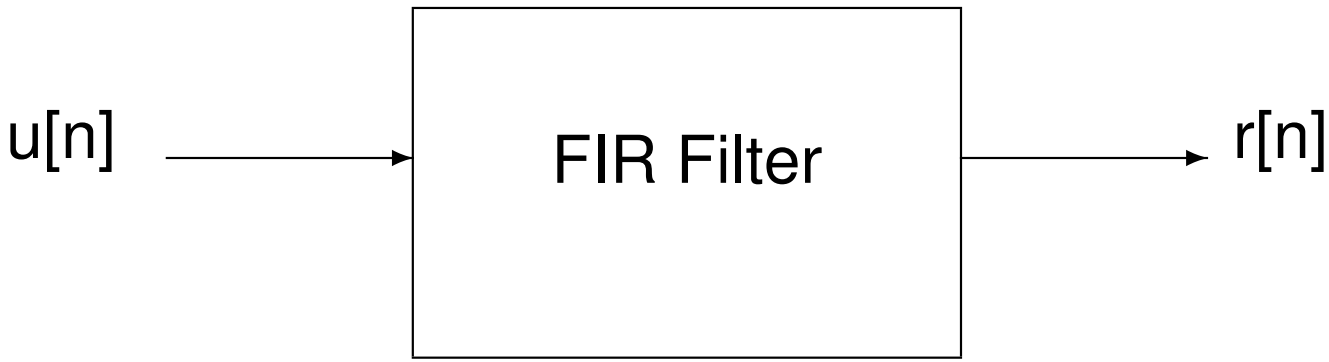
with filter coefficients $b_k = \{1, -1, 2\}$ when the input signal is $x[n] = \{1, 2, 4, 2, 4, 2, 1\}$.

Unit Step Sequence and Unit Step Response

- The signal with samples

$$u[n] = \begin{cases} 1 & \text{for } n \geq 0, \\ 0 & \text{for } n < 0 \end{cases}$$

  is called the unit-step sequence or unit-step signal.
- The output of an FIR filter when the input is the unit-step signal ($x[n] = u[n]$) is called the unit-step response $r[n]$.



Unit-Step Response of 3-Point Averager

- Input signal: $x[n] = u[n]$.
- Output signal: $r[n] = \frac{1}{3}\sum_{k=0}^{2} u[n-k]$.

n	-1	0	1	2	3	...
$u[n]$	0	1	1	1	1	...
$\frac{1}{3}u[n]$	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	...
$+\frac{1}{3}u[n-1]$	0	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	...
$+\frac{1}{3}u[n-2]$	0	0	0	$\frac{1}{3}$	$\frac{1}{3}$	...
$r[n]$	0	$\frac{1}{3}$	$\frac{2}{3}$	1	1	...

Unit-Impulse Sequence and Unit-Impulse Response

- The signal with samples

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

  is called the unit-impulse sequence or unit-impulse signal.
- The output of an FIR filter when the input is the unit-impulse signal ($x[n] = \delta[n]$) is called the unit-impulse response, denoted $h[n]$.
- Frequently, we will simply call the above signals simply impulse signal and impulse response.
- We will see that the impulse-response captures all characteristics of a FIR filter

Unit-Impulse Response of a FIR Filter

- Input signal: $x[n] = \delta[n]$.
- Output signal: $h[n] = \frac{1}{3}\sum_{k=0}^{M} \delta[n-k]$.

n	-1	0	1	2	3	...	M
$\delta[n]$	0	1	0	0	0	...	0
$b_0 \cdot \delta[n]$	0	b_0	0	0	0	...	0
$+b_1 \cdot \delta[n-1]$	0	0	b_1	0	0	...	0
$+b_2 \cdot \delta[n-2]$	0	0	0	b_2	0	...	0
$\vdots$				$\vdots$			
$+b_M \cdot \delta[n-M]$	0	0	0	0	0	...	b_M
$h[n]$	0	b_0	b_1	b_2	b_3	...	b_M

Insights

- For an FIR filter, the impulse response equals the sequence of filter coefficients:

$$h[n] = \begin{cases} b_n & \text{for } n = 0, 1, \ldots, M \\ 0 & \text{else.} \end{cases}$$

- Because of this relationship, the system relationship for an FIR filter can also be written as

$$\begin{aligned} y[n] &= \sum_{k=0}^{M} b_k x[n-k] \\ &= \sum_{k=0}^{M} h[k] x[n-k] \\ &= \sum_{-\infty}^{\infty} h[k] x[n-k]. \end{aligned}$$

- The operation $y[n] = h[n] * x[n] = \sum_{-\infty}^{\infty} h[k]x[n-k]$ is called convolution; it is a **very, very** important operation.

Lecture: Linear, Time-Invariant Systems

Introduction

- We have introduced systems as devices that process an input signal $x[n]$ to produce an output signal $y[n]$.
- **Example Systems:**
  - **Squarer:** $y[n] = (x[n])^2$
  - **Modulator:** $y[n] = x[n] \cdot \cos(2\pi f_d n)$, with $0 < f_d \leq \frac{1}{2}$.
  - **FIR Filter:**

$$y[n] = \sum_{k=0}^{M} h[k] \cdot x[n-k].$$

    Recall that $h[k]$ is the impulse response of the filter and that the above operation is called convolution of $h[n]$ and $x[n]$.
- **Objective:** Define important characteristics of systems and determine which systems possess these characteristics.

Causal Systems

- **Definition:** A system is called causal when it uses only the present and past samples of the input signal to compute the present value of the output signal.
- Causality is usually easy to determine from the system equation:
  - The output $y[n]$ must depend only on input samples $x[n], x[n-1], x[n-2], \ldots$.
  - Input samples $x[n+1], x[n+2], \ldots$ must not be used to find $y[n]$.
- **Examples:**
  - All three systems on the previous slide are causal.
  - The following system is non-causal:

$$y[n] = \frac{1}{3}\sum_{k=-1}^{1} x[n-k] = \frac{1}{3}(x[n+1] + x[n] + x[n-1]).$$

Linear Systems

- The following test procedure defines linearity and establishes how one can determine if a system is linear:
  1. **References:** For $i = 1, 2$, pass input signal $x_i[n]$ through the system to obtain output $y_i[n]$.
  2. **Linear Combination:** Form a new signal $x[n]$ from the linear combination of $x_1[n]$ and $x_2[n]$:

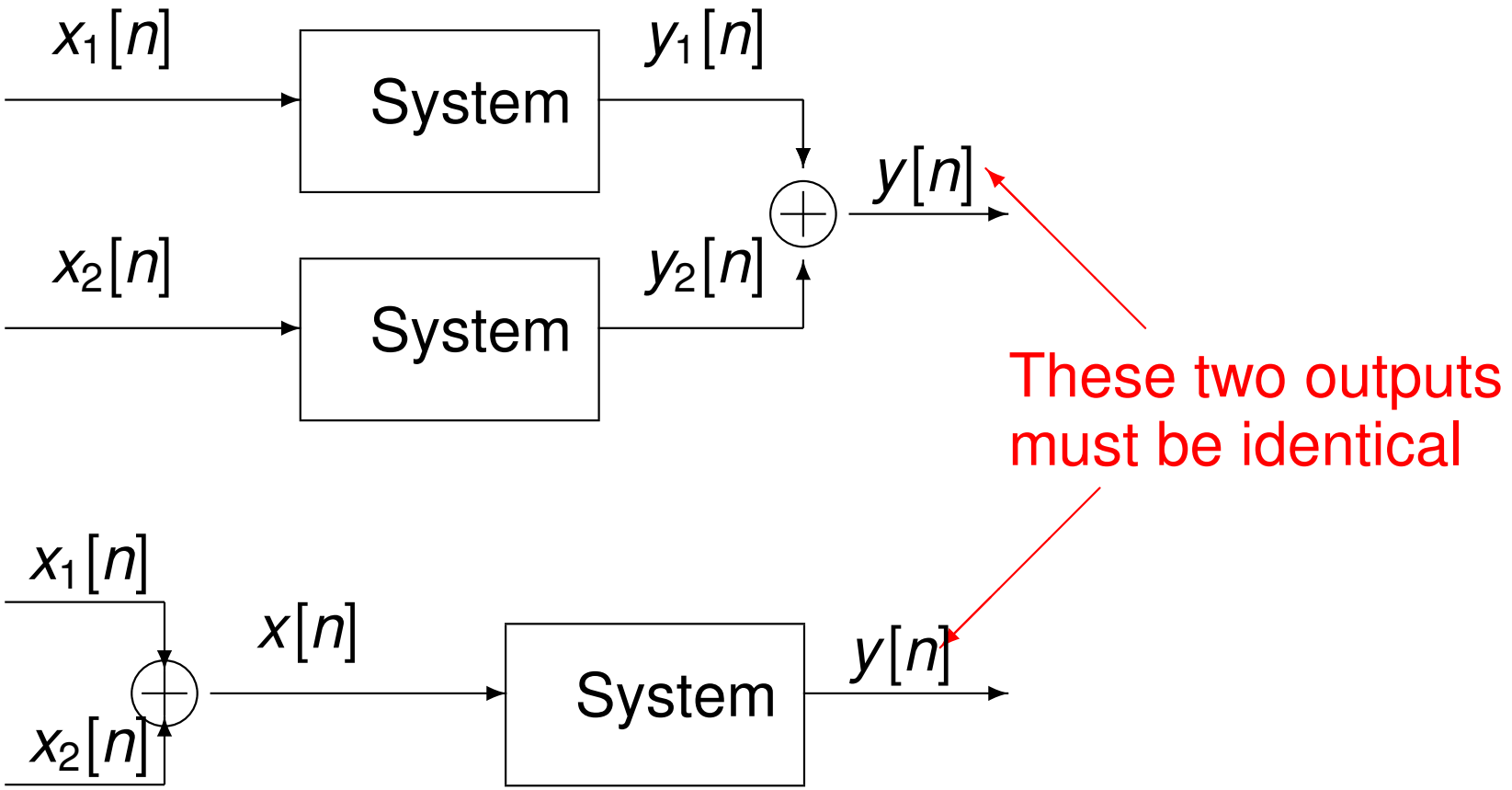
     $$x[n] = x_1[n] + x_2[n].$$

     Then, Pass signal $x[n]$ through the system and obtain $y[n]$.
  3. **Check:** The system is linear if

     $$y[n] = y_1[n] + y_2[n]$$

- The above must hold for **all** inputs $x_1[n]$ and $x_2[n]$.
- For a linear system, the superposition principle holds.

Illustration



Example: Squarer

- **Squarer:** $y[n] = (x[n])^2$
  1. **References:** $y_i[n] = (x_i[n])^2$ for $i = 1, 2$.
  2. **Linear Combination:** $x[n] = x_1[n] + x_2[n]$ and

     $$y[n] = (x[n])^2 = (x_1[n]+x_2[n])^2 = (x_1[n])^2+(x_2[n])^2+2x_1[n]x_2[n].$$

  3. **Check:**

     $$y[n] \neq y_1[n] + y_2[n] = (x_1[n])^2 + (x_2[n])^2.$$

- **Conclusion:** not linear.

Example: Modulator

- **Modulator:** $y[n] = x[n] \cdot \cos(2\pi f_d n)$
  1. **References:** $y_i[n] = x_i[n] \cdot \cos(2\pi f_d n)$ for $i = 1, 2$.
  2. **Linear Combination:** $x[n] = x_1[n] + x_2[n]$ and

$$y[n] = x[n] \cdot \cos(2\pi f_d n) = (x_1[n] + x_2[n]) \cdot \cos(2\pi f_d n).$$

  3. **Check:**

$$y[n] = y_1[n] + y_2[n] = x_1[n] \cdot \cos(2\pi f_d n) + x_2[n] \cdot \cos(2\pi f_d n).$$

- **Conclusion:** linear.

Example: FIR Filter

- **FIR Filter:** $y[n] = \sum_{k=0}^{M} h[k] \cdot x[n-k]$
  1. **References:** $y_i[n] = \sum_{k=0}^{M} h[k] \cdot x_i[n-k]$ for $i = 1, 2$.
  2. **Linear Combination:** $x[n] = x_1[n] + x_2[n]$ and

$$y[n] = \sum_{k=0}^{M} h[k] \cdot x[n-k] = \sum_{k=0}^{M} h[k] \cdot (x_1[n-k] + x_2[n-k]).$$

  3. **Check:**

$$y[n] = y_1[n] + y_2[n] = \sum_{k=0}^{M} h[k] \cdot x_1[n-k] + \sum_{k=0}^{M} h[k] \cdot x_2[n-k].$$

- **Conclusion:** linear.

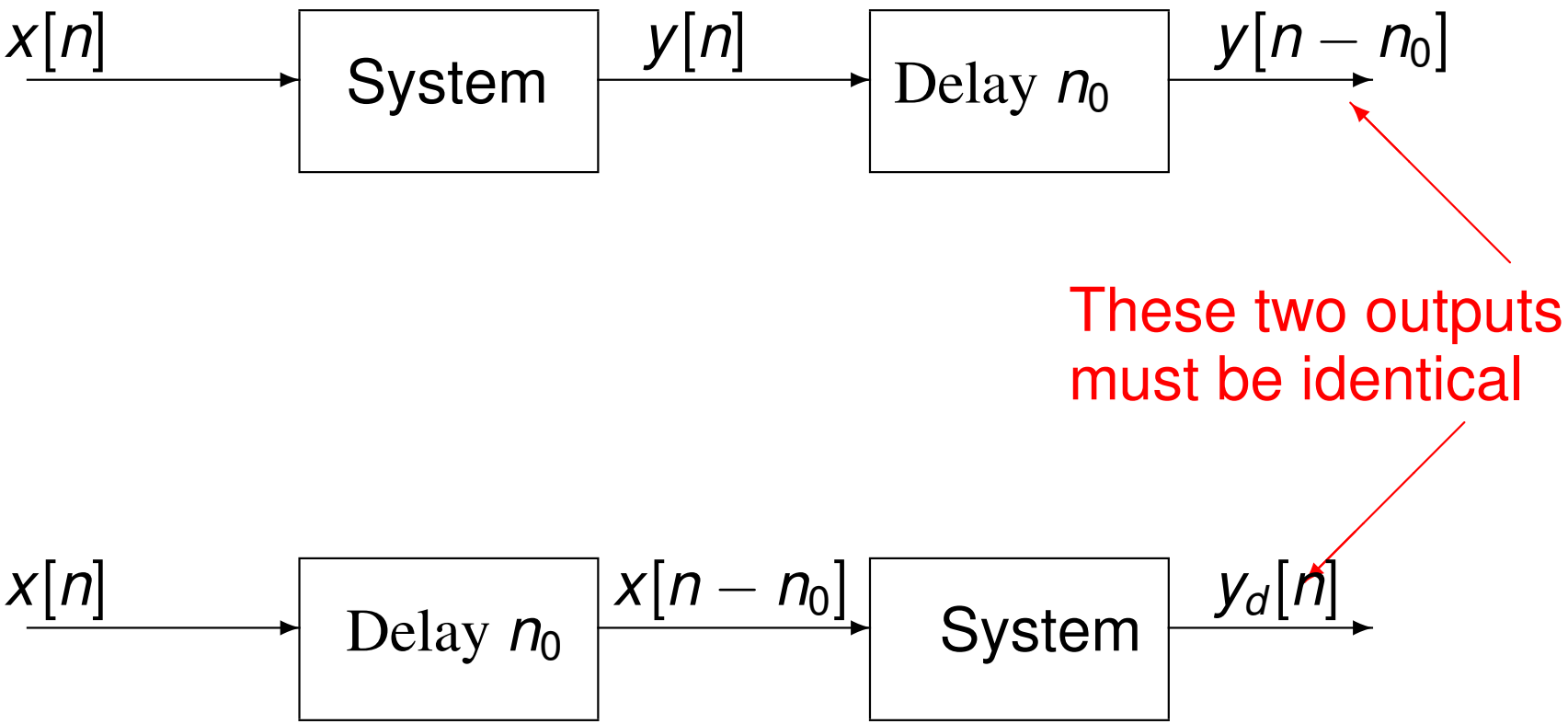
Time-invariance

- The following test procedure defines time-invariance and establishes how one can determine if a system is time-invariant:
  1. **Reference:** Pass input signal $x[n]$ through the system to obtain output $y[n]$.
  2. **Delayed Input:** Form a new signal $x_d[n] = x[n - n_0]$. Then, Pass signal $x_d[n]$ through the system and obtain $y_d[n]$.
  3. **Check:** The system is time-invariant if

$$y[n - n_0] = y_d[n]$$

- The above must hold for **all** inputs $x[n]$ and all delays n_0.
- **Interpretation:** A time-invariant system does not change the way it processes the input signal with time.

Illustration



Example: Squarer

- **Squarer:** $y[n] = (x[n])^2$
  1. **Reference:** $y[n] = (x[n])^2$.
  2. **Delayed Input:** $x_d[n] = x[n - n_0]$ and
  $$y_d[n] = (x_d[n])^2 = (x[n - n_0])^2.$$
  3. **Check:**
  $$y[n - n_0] = (x[n - n_0])^2 = y_d[n].$$
- **Conclusion:** time-invariant.

Example: Modulator

- **Modulator:** $y[n] = x[n] \cdot \cos(2\pi f_d n)$.
  1. **Reference:** $y[n] = x[n] \cdot \cos(2\pi f_d n)$.
  2. **Delayed Input:** $x_d[n] = x[n - n_0]$ and

$$y_d[n] = x_d[n] \cdot \cos(2\pi f_d n) = x[n - n_0] \cdot \cos(2\pi f_d n).$$

  3. **Check:**

$$y[n - n_0] = x[n - n_0] \cdot \cos(2\pi f_d (n - n_0)) \neq y_d[n].$$

- **Conclusion:** not time-invariant.

Example: Modulator

- Alternatively, to show that the modulator is **not** time-invariant, we construct a counter-example.
- Let $x[n] = \{0, 1, 2, 3, \ldots\}$, i.e., $x[n] = n$, for $n \geq 0$.
- Also, let $f_d = \frac{1}{2}$, so that

$$\cos(2\pi f_d n) = \begin{cases} 1 & \text{for } n \text{ even} \\ -1 & \text{for } n \text{ odd} \end{cases}$$

- Then, $y[n] = x[n] \cdot \cos(2\pi f_d n) = \{0, -1, 2, -3, \ldots\}$.
- With $n_0 = 1$, $x_d[n] = x[n-1] = \{0, 0, 1, 2, 3, \ldots\}$ and $y_d[n] = \{0, 0, 1, -2, 3, \ldots\}$.
- Clearly, $y_d[n] \neq y[n-1]$.
- not time-invariant

Example: FIR Filter

- **Reference:** $y[n] = \sum_{k=0}^{M} h[k] \cdot x[n-k]$.
- **Delayed Input:** $x_d[n] = x[n-n_0]$, and

$$y_d[n] = \sum_{k=0}^{M} h[k] \cdot x_d[n-k] = \sum_{k=0}^{M} h[k] \cdot x[n-n_0-k].$$

- **Check:**

$$y[n-n_0] = \sum_{k=0}^{M} h[k] \cdot x[n-n_0-k] = y_d[n]$$

- time-invariant

Exercise

- Let $u[n]$ be the unit-step sequence (i.e., $u[n] = 1$ for $n \geq 0$ and $u[n] = 0$, otherwise).
- The system is a 3-point averager:

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

1. Find the output $y_1[n]$ when the input $x[n] = u[n]$.
2. Find the output $y_2[n]$ when the input $x[n] = u[n-2]$.
3. Find the output $y[n]$ when the input $x[n] = u[n] - u[n-2]$.
4. How are linearity and time-invariance evident in your results?

Lecture: Convolution and Linear, Time-Invariant Systems

Introduction

- **We learned so far:**
  - For FIR filters, input-output relationship

$$y[n] = \sum_{k=0}^{M} b_k x[n-k].$$

  - If $x[n] = \delta[n]$, then $y[n] = h[n]$ is called the impulse response of the system.
  - For FIR filters:

$$h[n] = \begin{cases} b_n & \text{for } 0 \leq n \leq M \\ 0 & \text{else.} \end{cases}$$

  - **Convolution:** input-output relationship

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

Linearity and Time-Invariance

- **Linearity:**
  - For arbitrary input signals $x_1[n]$ and $x_2[n]$, let the ouputs be denoted $y_1[n]$ and $y_2[n]$.
  - Further, for the input signal $x[n] = x_1[n] + x_2[n]$, let the output signal be $y[n]$.
  - The system is linear if $y[n] = y_1[n] + y_2[n]$.
- **Time-Invariance:**
  - For an arbitrary input signal $x[n]$, let the output be $y[n]$.
  - For the delayed input $x_d[n] = x[n - n_0]$, let the output be $y_d[n]$.
  - The system is time-invariant if $y_d[n] = y[n - n_0]$.
- **Today:** For any linear, time-invariant system: input-output relationship is $y[n] = x[n] * h[n]$.

Reminders

- We need a few more facts and relationships for the impulse signal $\delta[n]$.
- Recall:
  - If input to a system is the impulse signal $\delta[n]$,
  - then, the output is called the impulse response,
  - and is denoted by $h[n]$.
- We showed that for FIR filters,
  - the output signal $y[n]$
  - is computed from the input signal $x[n]$
  - and the impulse response $y[n]$:

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

  - This operation is called convolution.

Identity System

- **Question:** How can one characterize a LTI system for which the output $y[n]$ is the same as the input $x[n]$.
  - Such a system is called the identity system.
- Specifically, we want the impulse response $h[n]$ of such a system.
- As always, one finds the impulse response $h[n]$ as the output of the LTI system when the impulse $\delta[n]$ is the input.
- Since the ouput is the same as the input for an identity system, we find

$$h[n] = \delta[n].$$

Ideal Delay Systems

- **Closely Related Question:** How can one characterize a LTI system for which the output $y[n]$ is a delayed version of the input $x[n]$:

$$y[n] = x[n - n_0]$$

where n_0 is the delay introduced by the system
  - Such a system is called an ideal delay system.
- Again, we want the impulse response $h[n]$ of such a system.
- As before, one finds the impulse response $h[n]$ as the output of the LTI system when the impulse $\delta[n]$ is the input.
- Since the ouput is merely a delayed version of the input, we find

$$h[n] = \delta[n - n_0].$$

Decomposing a Signal with Impulses

- **Objective:** Express a signal $x[n]$ as a sum of (scaled and delayed) impulses:

$$x[n] = \sum_{k=-\infty}^{\infty} x_k \delta[n-k].$$

- Recall, for the identity system:
  - impulse response: $\delta[n]$
  - input equals ouput.
- Hence:

$$x[n] = \delta[n] * x[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k].$$

Illustration

n	$\ldots$	-1	0	1	2	$\ldots$
$x[n]$	$\ldots$	x[-1]	x[0]	x[1]	x[2]	$\ldots$
$\delta[n]$	$\ldots$	0	1	0	0	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x[-1] \cdot \delta[n+1]$	$\ldots$	x[-1]	0	0	0	$\ldots$
$x[0] \cdot \delta[n]$	$\ldots$	0	x[0]	0	0	$\ldots$
$x[1] \cdot \delta[n-1]$	$\ldots$	0	0	x[1]	0	$\ldots$
$x[2] \cdot \delta[n-2]$	$\ldots$	0	0	0	x[2]	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k]$	$\ldots$	x[-1]	x[0]	x[1]	x[2]	$\ldots$

A few more Pre-requisistes

- A few more considerations:
  - We know already that input $\delta[n]$ produces output $h[n]$ (impulse repsonse). We write:

$$\delta[n] \mapsto h[n].$$

  - For a time-invariant system:

$$\delta[n-k] \mapsto h[n-k].$$

  - And for a linear system:

$$x[k] \cdot \delta[n-k] \mapsto x[k] \cdot h[n-k].$$

- Next, we use linearity to derive the convolution sum.

Derivation of the Convolution Sum

- Linearity implies that a linear combination of input signals produces an output signal that is equal to the linear combination of individual output signals.

Input	$\mapsto$	Output
$\vdots$	$\vdots$	$\vdots$
$x[-1] \cdot \delta[n+1]$	$\mapsto$	$x[-1] \cdot h[n+1]$
$x[0] \cdot \delta[n]$	$\mapsto$	$x[0] \cdot h[n]$
$x[1] \cdot \delta[n-1]$	$\mapsto$	$x[1] \cdot h[n-1]$
$x[2] \cdot \delta[n-1]$	$\mapsto$	$x[2] \cdot h[n-2]$
$\vdots$	$\vdots$	$\vdots$
$\sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k] = x[n]$	$\mapsto$	$y[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot h[n-k]$

Summary and Conclusions

- We just derived the convolution sum formula:

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

- We only assumed that the system is linear and time-invariant.
- Therefore, we can conclude that *for any* linear, time-invariant system, the output is the convolution of input and impulse response.
  - Needless to say: convolution and impulse response are enormously important concepts.

Exercise

- Show that convolution is a commutative operation, i.e., that $x[n] * h[n]$ equals $h[n] * x[n]$.

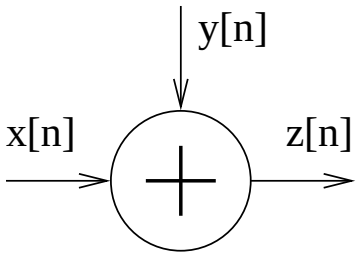
Lecture: Convolution and Linear, Time-Invariant Systems

Building Blocks

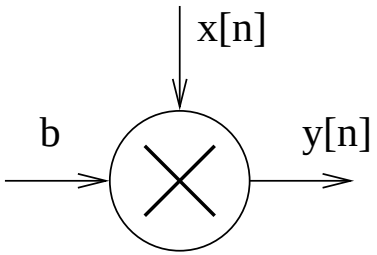
- Recall that the input-output relationship for an FIR filter is given by

$$y[n] = \sum_{k=0}^{M} b_k x[n-k].$$

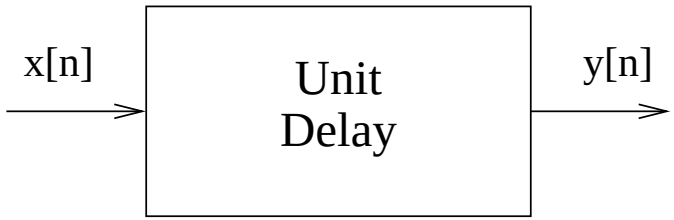
- Digital systems implementing this relationships are easily constructed from simple building blocks:



Adder

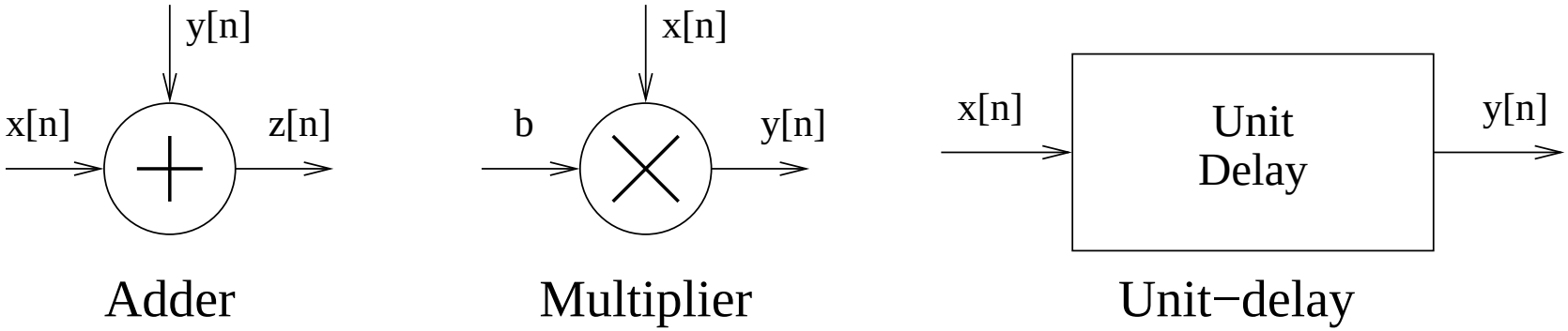


Multiplier



Unit−delay

Operation of Building Blocks



- **Adder:** sum of two signals

$$z[n] = x[n] + y[n].$$

- **Multiplier:** product of signal with a scalar

$$y[n] = b \cdot x[n]$$

- **Unit-delay:** delays input by one sample:

$$y[n] = x[n-1]$$

Block Diagrams