Signal Processing Systems

- System Abstraction
- Superposition and Convolution
- Filtering
Many applications of signal processing can be thought of as systems that convert an input signal into an output signal.

Examples:
- audio enhancement: equalization, noise reduction, reverberation, echo cancellation, pitch shift (auto-tune)
- image enhancement: smoothing, edge enhancement, unsharp masking, feature detection
- video enhancement: image stabilization, motion magnification
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} (x[n-1] + x[n] + x[n+1])$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
**Example: Three-Point Averaging**

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} (x[n-1] + x[n] + x[n+1])$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3}(x[n-1] + x[n] + x[n+1])$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

$$y[n] = \frac{1}{3} \left( x[n-1] + x[n] + x[n+1] \right)$$
Example: Three-Point Averaging

The output at time $n$ is average of inputs at times $n-1$, $n$, and $n+1$.

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

Think of this process as a system with input $x[n]$ and output $y[n]$.
Multiple Representations of a System

We will consider three different representations for such systems.

Difference Equation: algebraic constraint on samples

Convolution: represent system by its unit-sample response

Filter: as amplification/attenuation of frequency components
Properties of Systems

Arbitrary systems are arbitrarily difficult to describe.

Fortunately, many useful systems have two important properties:

- **linearity**: additive and homogeneous
- **time invariance**

The presence or absence of these properties directly determine the usefulness of each type of system representation:

- as a set of difference equations
- as a convolution
- as a filter
Additivity

A system is additive if its response to a **sum of inputs** is equal to the **sum of its responses** to each input taken one at a time.

Given

\[ x_1[n] \rightarrow \text{system} \rightarrow y_1[n] \]

and

\[ x_2[n] \rightarrow \text{system} \rightarrow y_2[n] \]

the **system is additive** if

\[ x_1[n] + x_2[n] \rightarrow \text{system} \rightarrow y_1[n] + y_2[n] \]

is true for all possible inputs.

Example: a recording of two musical instruments is (approximately) the sum of separate recordings of each.
Homogeneity

A system is homogeneous if multiplying its input by a constant multiplies its output by the same constant.

Given

\[ x_1[n] \xrightarrow{\text{system}} y_1[n] \]

the system is homogeneous if

\[ \alpha x_1[n] \xrightarrow{\text{system}} \alpha y_1[n] \]

is true for all \( \alpha \) and all possible inputs.

Example: doubling loudness of a musical instrument (approximately) doubles the loudness of a recording of this instrument.
Linearity

A system is linear if its response to a **weighted sum of inputs** is equal to the **weighted sum of its responses** to each of the inputs.

Given

\[ x_1[n] \xrightarrow{\text{system}} y_1[n] \]

and

\[ x_2[n] \xrightarrow{\text{system}} y_2[n] \]

the **system is linear** if

\[ \alpha x_1[n] + \beta x_2[n] \xrightarrow{\text{system}} \alpha y_1[n] + \beta y_2[n] \]

is true for all \( \alpha \) and \( \beta \) and all possible inputs.

A system is linear if it is both additive and homogeneous.
**Time-Invariance**

A system is time-invariant if delaying the input to the system simply delays the output by the same amount of time.

Given

\[ x[n] \rightarrow \text{system} \rightarrow y[n] \]

the **system is time invariant** if

\[ x[n-n_0] \rightarrow \text{system} \rightarrow y[n-n_0] \]

is true for all \( n_0 \) and for all possible inputs.

Example: Shifting the input to a 3-pt averager simply shifts the output by that same amount.
Consider a system represented by the following difference equation:

\[ y[n] = x[n] + x[n-1] \quad (\text{for all } n) \]

Is this system linear?

Assume that \( x_1[n] \to y_1[n] \). Then \( y_1[n] = x_1[n] + x_1[n-1] \).

Assume that \( x_2[n] \to y_2[n] \). Then \( y_2[n] = x_2[n] + x_2[n-1] \).

Multiply \( \alpha \) times equation 1 and add \( \beta \) times equation 2:

\[ \alpha y_1[n] + \beta y_2[n] = \alpha x_1[n] + \beta x_2[n] + \alpha x_1[n-1] + \beta x_2[n-1] \]

This equation shows that \( \alpha x_1[n] + \beta x_2[n] \to \alpha y_1[n] + \beta y_2[n] \).

Therefore the system is linear.
Determining linearity from a difference equation representation.

Example 2.

\[ y[n] = x[n] \times x[n-1] \quad \text{(for all } n) \]

Is this system \textbf{linear}?

Assume that \( x_1[n] \rightarrow y_1[n] \). Then \( y_1[n] = x_1[n] \times x_1[n-1] \).

Find the response \( y_2[n] \) when \( x_2[n] = \alpha x_1[n] \):

\[
\begin{align*}
y_2[n] &= x_2[n] \times x_2[n-1] \\
&= \alpha x_1[n] \times \alpha x_1[n-1] \\
&= \alpha^2 x_1[n] \times x_1[n-1] \\
&= \alpha^2 y_1[n]
\end{align*}
\]

Multiplying input \( x_1[n] \) by \( \alpha \) does \textbf{not} multiply the output \( y_1[n] \) by \( \alpha \). It multiplies \( y_1[n] \) by \( \alpha^2 \)!

Therefore the system is \textbf{neither homogeneous nor linear}. 


Representing Systems with Difference Equations

Determining linearity from a difference equation representation.

Example 3:

\[ y[n] = nx[n] \quad \text{(for all } n) \]

Is the system **linear**?

---

Let \( x[n] = \alpha x_1[n] + \beta x_2[n] \). Then

\[
\begin{align*}
y[n] &= n(\alpha x_1[n] + \beta x_2[n]) \\
    &= \alpha nx_1[n] + \beta nx_2[n] \\
    &= \alpha y_1[n] + \beta y_2[n]
\end{align*}
\]

Therefore the system is **linear**.
Representing Systems with Difference Equations

Determining time invariance from a difference equation.

Example 3.

\[ y[n] = nx[n] \quad \text{(for all } n) \]

Is the system **time-invariant**?

---

If time-invariant, delaying input by 1 should delay output by 1. Let \( x_1[n] \) represent a delayed version of the input.

\[ x_1[n] = x[n-1] \]

The corresponding output \( y_1[n] \) is given by

\[ y_1[n] = nx_1[n] = nx[n-1] \]

This is not the same as delaying the original output:

\[ y[n-1] = (n-1)x[n-1] \]

Since \( y_1[n] \neq y[n-1] \), the system is **not time-invariant**.
Linear Difference Equations with Constant Coefficients

A system is linear and time-invariant if it can be expressed in terms of a linear difference equation with constant coefficients.

General form:
\[\sum_m c_m y[n-m] = \sum_k d_k x[n-k]\]

**Additivity:** output of sum is sum of outputs
\[\sum_m c_m(y_1[n-m] + y_2[n-m]) = \sum_k d_k(x_1[n-k] + x_2[n-k])\]  √

**Homogeneity:** scaling an input scales its output
\[\sum_m \alpha c_m y_1[n-m] = \sum_k \alpha d_k x_1[n-k]\]  √

**Time invariance:** delaying an input delays its output
\[\sum_m c_m y_1[(n-n_0)-m] = \sum_k \alpha d_k x_1[(n-n_0)-k]\]  √

Notice that \(y[n] = x[n] + 1\) does **not** represent a linear system. Why?
Multiple Representations of a System

Representing a system by its unit-sample response.

**Difference Equation:** algebraic constraint on samples

**Convolution:** represent system by its unit-sample response

**Filter:** as amplification/attenuation of frequency components
If a system is linear and time-invariant, its input-output relation is completely specified by the system’s **unit-sample response** $h[n]$. The unit-sample response $h[n]$ is the output of the system when the input is the unit-sample signal $\delta[n]$. The output for more complicated inputs can be computed by summing scaled and shifted versions of the unit-sample response.
Superposition

Break input into additive parts and sum the responses to the parts.

\[
x[n] = y[n]
\]
Superposition

Break input into additive parts and sum the responses to the parts.

Note that this depends on both linearity and time-invariance.
Structure of Superposition

If a system is linear and time-invariant (LTI) then its output is the sum of weighted and shifted unit-sample responses.

\[
\begin{align*}
\delta[n] &\rightarrow \text{system} \rightarrow h[n] \\
\delta[n-k] &\rightarrow \text{system} \rightarrow h[n-k] \\
x[k]\delta[n-k] &\rightarrow \text{system} \rightarrow x[k]h[n-k] \\
x[n] &= \sum_{k=-\infty}^{\infty} x[k]\delta[n-k] \rightarrow \text{system} \rightarrow y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]
\end{align*}
\]
Convolution

Response of an LTI system to an arbitrary input.

\[ y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \equiv (x * h)[n] \]

This operation is called \textit{convolution}.
Notation

Convolution is represented with an asterisk.

\[
\sum_{k=-\infty}^{\infty} x[k]h[n-k] \equiv (x \ast h)[n]
\]

It is customary (but confusing) to abbreviate this notation:

\[(x \ast h)[n] = x[n] \ast h[n]\]
Notation

Do not be fooled by the confusing notation.

Confusing (but conventional) notation:
\[
\sum_{k=-\infty}^{\infty} x[k]h[n-k] = x[n] * h[n]
\]

\(x[n] * h[n]\) looks like an operation of samples; but it is not!
\[x[1] * h[1] \neq (x * h)[1]\]

Convolution operates on signals not samples.

Unambiguous notation:
\[
\sum_{k=-\infty}^{\infty} x[k]h[n-k] \equiv (x * h)[n]
\]

The symbols \(x\) and \(h\) represent DT signals. Convolving \(x\) with \(h\) generates a new DT signal \(x * h\).
Structure of Convolution

\[ y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n - k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k]h[0 - k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k]h[0-k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k] h[0 - k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k]h[0 - k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k] h[0 - k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k] h[0 - k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k]h[0-k] \]
Structure of Convolution

\[ y[0] = \sum_{k=-\infty}^{\infty} x[k] h[0 - k] \]
Structure of Convolution

\[ y[1] = \sum_{k=-\infty}^{\infty} x[k] h[1-k] \]
Structure of Convolution

\[ y[2] = \sum_{k=-\infty}^{\infty} x[k] h[2 - k] \]
Structure of Convolution

\[ y[3] = \sum_{k=-\infty}^{\infty} x[k] h[3 - k] \]

Diagram:
- \( x[k] \)
- \( h[3 - k] \)
- \( x[k] h[3 - k] \)
- \( \sum_{k=-\infty}^{\infty} \)
- \( = 2 \)
Structure of Convolution

\[ y[4] = \sum_{k=-\infty}^{\infty} x[k] h[4 - k] \]

\[ \sum_{k=-\infty}^{\infty} x[k] h[4 - k] = 1 \]
Structure of Convolution

\[ y[5] = \sum_{k=-\infty}^{\infty} x[k] h[5 - k] \]

While this operation looks complicated, it is \textit{exactly} equivalent to superposition.
DT Convolution: Summary

Unit-sample response $h[n]$ is a complete description of an LTI system.

$$x[n] \rightarrow h[n] \rightarrow y[n]$$

Given $h[n]$ one can compute the response $y[n]$ to any input $x[n]$:

$$y[n] = (x * h)[n] \equiv \sum_{k=-\infty}^{\infty} x[k]h[n - k]$$
The same sort of reasoning applies to CT signals.

\[ x(t) = \lim_{\Delta \to 0} \sum_{k} x(k\Delta)p(t - k\Delta)\Delta \]

where

\[ x(t) \to \int_{-\infty}^{\infty} x(\tau)\delta(t - \tau)d\tau \]

As \( \Delta \to 0 \), \( k\Delta \to \tau \), \( \Delta \to d\tau \), and \( p(t) \to \delta(t) \):
Structure of Superposition

If a system is linear and time-invariant (LTI) then its output is the integral of weighted and shifted unit-impulse responses.

\[
\delta(t) \rightarrow \text{system} \rightarrow h(t)
\]

\[
\delta(t - \tau) \rightarrow \text{system} \rightarrow h(t - \tau)
\]

\[
x(\tau)\delta(t - \tau) \rightarrow \text{system} \rightarrow x(\tau)h(t - \tau)
\]

\[
x(t) = \int_{-\infty}^{\infty} x(\tau)\delta(t - \tau)d\tau \rightarrow \text{system} \rightarrow y(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau)d\tau
\]
CT Convolution

Convolution of CT signals is analogous to convolution of DT signals.

**DT:** \[ y[n] = (x * h)[n] = \sum_{k=-\infty}^{\infty} x[k]h[n - k] \]

**CT:** \[ y(t) = (x * h)(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau)d\tau \]
Multiple Representations of a System

Representing a system by its unit-sample response.

**Difference Equation:** algebraic constraint on samples  

**Convolution:** represent system by its unit-sample response  

**Filter:** as amplification/attenuation of frequency components
Frequency Representation of Convolution

Let \( y[n] = (h \ast x)[n] \). Find \( Y(\Omega) \).

\[
Y(\Omega) = \sum_{n=-\infty}^{\infty} (h \ast x)[n] e^{-j\Omega n} \quad \text{(Fourier analysis equation)}
\]

\[
= \sum_{n=-\infty}^{\infty} \left( \sum_{m=-\infty}^{\infty} h[m]x[n - m] \right) e^{-j\Omega n} \quad \text{(definition of convolution)}
\]

\[
= \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h[m]x[n - m] e^{-j\Omega n} \quad \text{(swap order of summation)}
\]

\[
= \sum_{m=-\infty}^{\infty} h[m] \sum_{l=-\infty}^{\infty} x[l] e^{-j\Omega (l+m)} \quad \text{(pull out } h[m]; \ l = n - m)\]

\[
= \sum_{m=-\infty}^{\infty} h[m] e^{-j\Omega m} \sum_{l=-\infty}^{\infty} x[l] e^{-j\Omega l} \quad \text{(separate } l \text{ and } m \text{ terms)}
\]

\[
= H(\Omega)X(\Omega)
\]

Convolving two signals is equivalent to multiplying their transforms.
Filtering

Multiplication of Fourier transforms can be regarded as **filtering**.

\((h \ast x)[n] \overset{\text{DTFT}}{\longleftrightarrow} H(\Omega)X(\Omega)\)

**Time Domain**

\[
x[n] \rightarrow h[n] \rightarrow y[n] = (h \ast x)[n]
\]

**Frequency Domain**

\[
X(\Omega) \rightarrow H(\Omega) \rightarrow Y(\Omega) = H(\Omega)X(\Omega)
\]

Each frequency component of input \(X(\Omega)\) is scaled by a factor \(H(\Omega)\). The system is completely described by the set of scale factors \(H(\Omega)\).
Multiple Representations of a System

Representing a system by its unit-sample response.

Difference Equation: algebraic constraint on samples

Convolution: represent system by its unit-sample response

Filter: as amplification/attenuation of frequency components