Signal Definition

$$\begin{aligned}&x(n) = z \text{ where } z \in Z\\&\qquad or\\&x(n) = c \text{ where } c \in \mathcal{C}\end{aligned}$$

a signal may be finite or infinite length

Filter Definition

a real or complex mapping of $x(n)$ to $y(n)$ for every n.

$$\begin{aligned}&\text{where } \mathcal{S} \text{ is the signal space}\\&y(n) = \mathcal{T}\{x(.)\}\\&X \in \mathcal{S} \rightarrow Y \in \mathcal{S}\end{aligned}$$

If $\mathcal{S}$ only consists of N length signals then every linear filter $\mathcal{T}$ can be represented as an NxN matrix

LTI Filters

LTI filters are the only filters which preserve signal frequencies (prove)

- Linearity: that each output at time n is a linear combination of signals, the coefficients of which do not depend on the inputs or outputs (x or y). Scaling and superposition properties are required. See more at Linearity
- Time-invariance means that the coefficients applied to the input(s) do not vary with time.

Example: $y(n) = c_1x(n) + c_2x(n+1)$

- The coefficients can be complex or real, and the filter will remain LTI.
- This is an example of a non-causal filter, since it uses $x(n+1)$, therefore needs knowledge of the future.
- Use of past outputs $y(n)$ is called *feedback* or *recursive* filters.
  Example of a multi-input multi-output (MIMO) filter

$$\begin{bmatrix}y_1(n)\\y_2(n)\end{bmatrix} = \begin{bmatrix}a & b\\c & d\end{bmatrix}\begin{bmatrix}x_1(n)\\x_2(n)\end{bmatrix} + \begin{bmatrix}e & f\\g & h\end{bmatrix}\begin{bmatrix}x_1(n-1)\\x_2(n-1)\end{bmatrix}$$

Non-linear filters

Simplest example, which is also memoryless

$$y(n) = x^2(n)$$

Another example is a median smoother of order N which uses the median of N samples centered around time n to determine the output at time n.

Time variant filters

Example:

$$y(n) = x(n) + \cos(2\pi n)x(n-1)$$

is time-variant because the coefficient changes depending on n. It is linear however because no coefficients depend on x or y.

Linearity

2 properties have to hold for linearity
- Scaling: The amplitude of the output is proportional to the amplitude of the input (the *scaling property*).
- Superposition: When two signals are added together and fed to the filter, the filter output is the same as if one had put each signal through the filter separately and then added the outputs (the *superposition property*).

Mathmatical Definition

Let's say $\mathcal{L}$ is a linear, but not necessarily time-invariant filter. The output is dependent on 1 or more of x's variables, which is why we use $x(.)$.

$$y(n) = \mathcal{L}_n(x(.))$$

$\mathcal{L}_n$ is said to be linear if:

$$\begin{aligned} \text{Scaling}: \mathcal{L}_n\{gx(.)\} &= g\mathcal{L}_n\{x(.)\} \quad \forall g \in C, \forall x \in \mathcal{S} \\ \text{Superposition}: \mathcal{L}_n\{x_1(.) + x_2(.)\} &= \mathcal{L}_n\{x_1(.)\} + \mathcal{L}_n\{x_2(.)\} \quad \forall x_1, x_2 \in \mathcal{S} \end{aligned}$$

Where $\mathcal{S}$ denotes signal space, or complex values sequences in general.

Real Filtering of Complex Signals

$$\begin{aligned} w &= x + jy \\ \mathcal{L}_n\{w\} &\triangleq \mathcal{L}_n\{x\} + j\mathcal{L}_n\{y\} \end{aligned}$$

This means that applying a filter to a complex signal will treat the real and complex parts seperately

Time invariance:

If the input is shifted by N samples, the output is shifted by N samples.

$$\mathcal{L}_n\{SHIFT_N(x)\} = \mathcal{L}_n\{\text{SHIFT}_{N,n}(y)\} = \mathcal{L}_{n-N}\{x\}) = y(n-N)$$

$$\text{where}: \mathcal{L}_n\{SHIFT_N(x)\} \triangleq x(.-N)$$

Sliding linear combinations

$$y(n) = b_1 x(n) + b_2 x(n-1)$$

Any filter of the above form is linear and time invariant, and is a special case of sliding linear combination/weighted sum. Time invariance depends on b being a constant.
By induction, it can also be proven than adding a feedback term to these filters remains linear and time-invariant (see [problems](###Problems and proofs))

Problems and proofs

See https://ccrma.stanford.edu/~jos/filters/Showing_Linearity_Time_Invariance.html
Question: is $y(n) = c$ linear or time-invariant?

further reading

volterra series, can represent every non-linear system:
https://ccrma.stanford.edu/~jos/filters/Analysis_Nonlinear_Filters.html
dynamic convolution, used for mapping memory-less non-linear functions followed by and LTI:
https://www.uaudio.com/webzine/2004/july/text/content2.html