

Mathematical Tools
for Neural and Cognitive Science

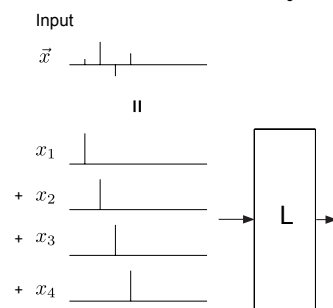
Fall semester, 2025

Section 3:
Linear Shift-Invariant Systems

Linear shift-invariant (LSI) systems

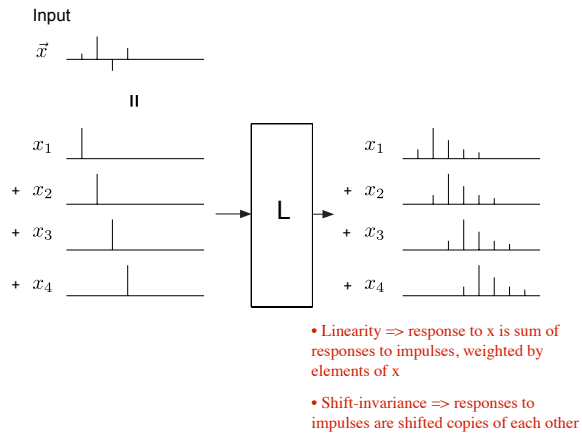
- Linearity (previously discussed):
“linear combination in, linear combination out”
- Shift-invariance (new property):
“shifted vector in, shifted vector out”
- These two properties are independent (think of
some examples that have both, one, or neither)

LSI system

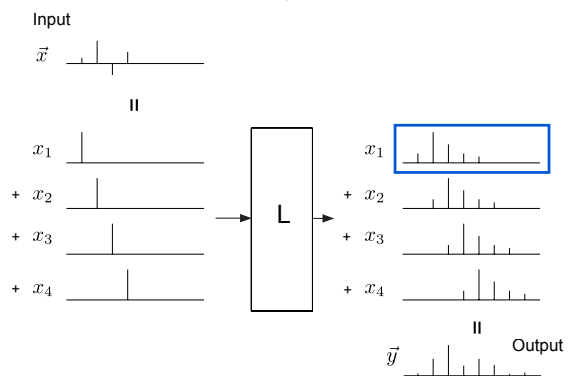


As before, express input as a sum of
“impulses”, weighted by elements of $\mathbf{x}$

LSI system

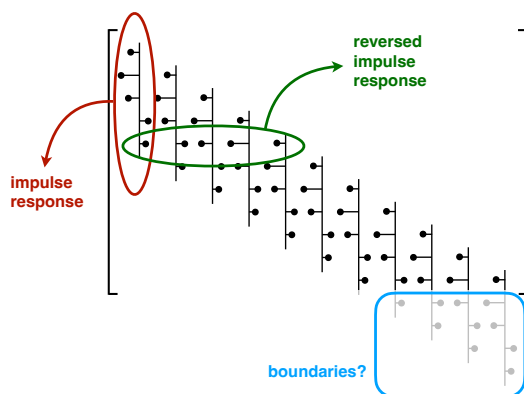


LSI system

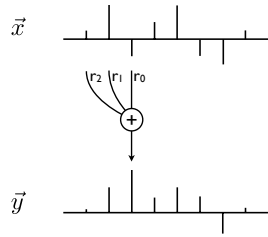


LSI systems are characterized by their “impulse response”

Convolution matrix



Convolution

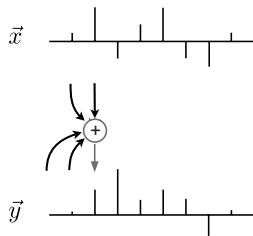


$$y(n) = \sum_k r(n-k)x(k)$$

$$= \sum_k r(k)x(n-k)$$

- Sliding dot product
- Structured matrix
- Boundaries? zero-padding, reflection, circular
- Examples: impulse, delay, average, difference

Feedback LSI system



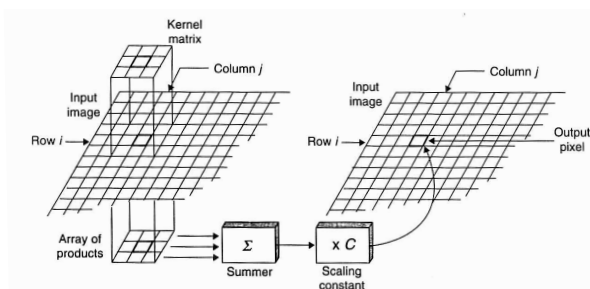
- Response depends on input, *and* previous outputs
- *Infinite* impulse response (IIR)
- Recursive => possibly *unstable*

$$y(n) = \sum_k f(n-k)x(k) + \sum_k g(n-k)y(k)$$

(For this class, we'll stick to feedforward (FIR) systems)

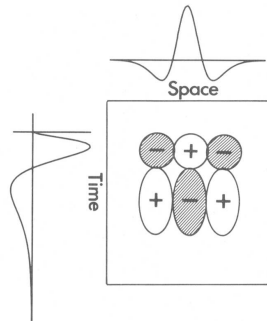
2D convolution

“sliding window”



[figure c/o Castleman]

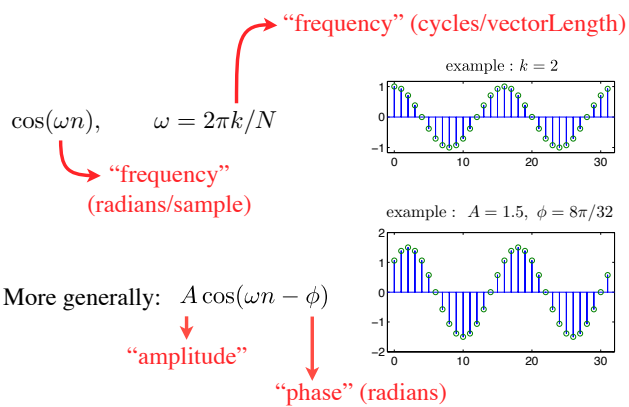
“separable” filter



- Outer product
- Simple design/implementation
- Efficient computation

[figure: Adelson & Bergen 85]

Discrete Sinusoids



Shifting Sinusoids

$$A \cos(\omega n - \phi) = A \cos(\phi) \cos(\omega n) + A \sin(\phi) \sin(\omega n)$$

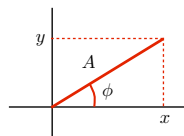
... using a well-known trigonometric identity:

$$\cos(a - b) = \cos(a) \cos(b) + \sin(a) \sin(b)$$

We’ll also need conversions between polar and rectangular coordinates:

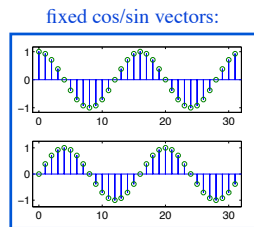
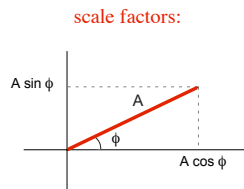
$$x = A \cos(\phi), \quad y = A \sin(\phi)$$

$$A = \sqrt{x^2 + y^2}, \quad \phi = \tan^{-1}(y/x)$$



Shifting Sinusoids

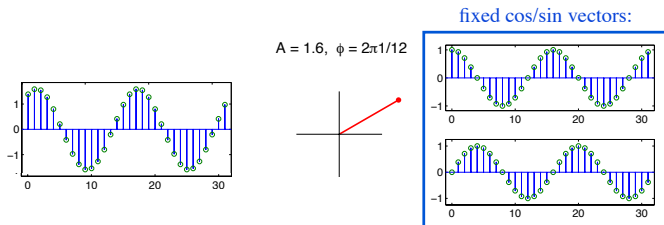
$$A \cos(\omega n - \phi) = \underbrace{A \cos(\phi)}_{\text{scale factors:}} \underbrace{\cos(\omega n)}_{\text{fixed cos/sin vectors:}} + \underbrace{A \sin(\phi)}_{\text{scale factors:}} \underbrace{\sin(\omega n)}_{\text{fixed cos/sin vectors:}}$$



Any *scaled* and *shifted* sinusoidal vector can be written as a weighted sum of two *fixed* {sin, cos} vectors!

Shifting Sinusoids

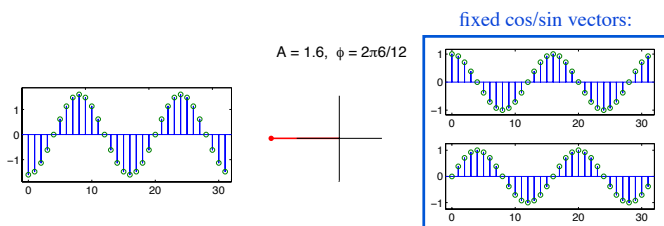
$$A \cos(\omega n - \phi) = \underbrace{A \cos(\phi)}_{\text{scale factors:}} \underbrace{\cos(\omega n)}_{\text{fixed cos/sin vectors:}} + \underbrace{A \sin(\phi)}_{\text{scale factors:}} \underbrace{\sin(\omega n)}_{\text{fixed cos/sin vectors:}}$$



Any *scaled* and *shifted* sinusoidal vector can be written as a weighted sum of two *fixed* {sin, cos} vectors!

Shifting Sinusoids

$$A \cos(\omega n - \phi) = \underbrace{A \cos(\phi)}_{\text{scale factors:}} \underbrace{\cos(\omega n)}_{\text{fixed cos/sin vectors:}} + \underbrace{A \sin(\phi)}_{\text{scale factors:}} \underbrace{\sin(\omega n)}_{\text{fixed cos/sin vectors:}}$$

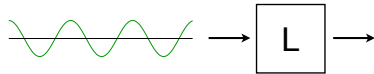


Any *scaled* and *shifted* sinusoidal vector can be written as a weighted sum of two *fixed* {sin, cos} vectors!

LSI response to sinusoids

$$x(n) = \cos(\omega n) \quad (\text{input})$$

$$y(n) = \sum_m r(m) \cos(\omega(n-m)) \quad (\text{convolution formula})$$



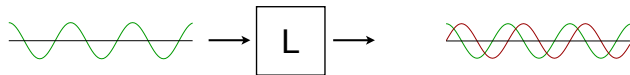
LSI response to sinusoids

$$x(n) = \cos(\omega n)$$

$$y(n) = \sum_m r(m) \cos(\omega(n-m))$$

$$= \sum_m r(m) \cos(\omega m) \cos(\omega n) + \sum_m r(m) \sin(\omega m) \sin(\omega n) \quad (\text{trig identity})$$

inner product of impulse response with cos/sin, respectively



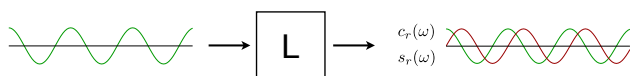
LSI response to sinusoids

$$x(n) = \cos(\omega n)$$

$$y(n) = \sum_m r(m) \cos(\omega(n-m))$$

$$= \sum_m r(m) \cos(\omega m) \cos(\omega n) + \sum_m r(m) \sin(\omega m) \sin(\omega n)$$

$$= c_r(\omega) \cos(\omega n) + s_r(\omega) \sin(\omega n)$$

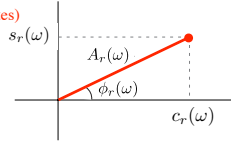


LSI response to sinusoids

$$x(n) = \cos(\omega n)$$

$$\begin{aligned} y(n) &= \sum_m r(m) \cos(\omega(n-m)) \\ &= \sum_m r(m) \cos(\omega m) \cos(\omega n) + \sum_m r(m) \sin(\omega m) \sin(\omega n) \\ &= c_r(\omega) \cos(\omega n) + s_r(\omega) \sin(\omega n) \\ &= A_r(\omega) \cos(\phi_r(\omega)) \cos(\omega n) + A_r(\omega) \sin(\phi_r(\omega)) \sin(\omega n) \end{aligned}$$

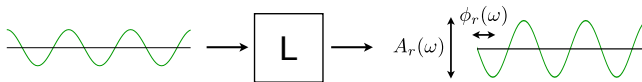
(cartesian -> polar coordinates)



LSI response to sinusoids

$$x(n) = \cos(\omega n)$$

$$\begin{aligned} y(n) &= \sum_m r(m) \cos(\omega(n-m)) \\ &= \sum_m r(m) \cos(\omega m) \cos(\omega n) + \sum_m r(m) \sin(\omega m) \sin(\omega n) \\ &= c_r(\omega) \cos(\omega n) + s_r(\omega) \sin(\omega n) \\ &= A_r(\omega) \cos(\phi_r(\omega)) \cos(\omega n) + A_r(\omega) \sin(\phi_r(\omega)) \sin(\omega n) \\ &= A_r(\omega) \cos(\omega n - \phi_r(\omega)) \quad (\text{trig identity, in the opposite direction}) \end{aligned}$$



“Sinusoid in, sinusoid out” (with modified amplitude & phase)

LSI response to sinusoids

More generally, if input has amplitude A_x and phase ϕ_x ,

$$x(n) = A_x \cos(\omega n - \phi_x)$$

then linearity and shift-invariance tell us that

$$y(n) = A_r(\omega) A_x \cos(\omega n - \phi_x - \phi_r(\omega))$$

amplitudes multiply

phases add

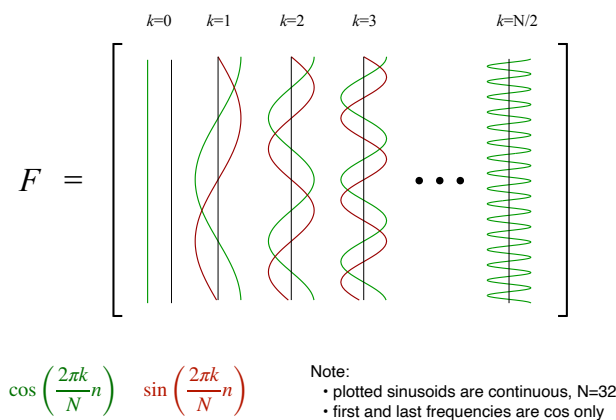


“Sinusoid in, sinusoid out” (with modified amplitude & phase)

The Discrete Fourier transform (DFT)

- Construct an orthogonal matrix of sin/cos pairs, covering different numbers of cycles
- Frequency multiples of $2\pi/N$ radians/sample, (specifically, $2\pi k/N$, for $k = 0, 1, 2, \dots, N/2$)

Discrete Fourier Transform matrix



The Discrete Fourier transform (DFT)

- Construct an orthogonal matrix of sin/cos pairs, covering different numbers of cycles
- Frequency multiples of $2\pi/N$ radians/sample, (specifically, $2\pi k/N$, for $k = 0, 1, 2, \dots, N/2$)
- For $k = 0$ and $k = N/2$, only need the cosine part (thus, $N/2 + 1$ cosines, and $N/2 - 1$ sines)
- When we apply this matrix to an input vector, think of output as *paired* coordinates
- Common to plot these pairs as amplitude/phase

[details on board...]

The Fourier family

frequency domain	signal domain	
	continuous	discrete
	continuous Fourier transform	discrete-time Fourier transform
discrete	Fourier series	● discrete Fourier transform

(we are here)

The “fast Fourier transform” (FFT) is a computationally efficient implementation of the DFT, requiring $N \log(N)$ operations, compared to the N^2 operations that would be needed for matrix multiplication.

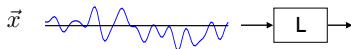
Reminder: LSI response to sinusoids

$$x(n) = \cos(\omega n)$$

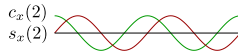
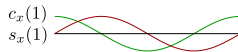
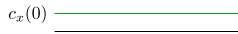
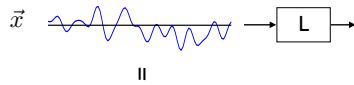
$$\begin{aligned}
 y(n) &= \sum_m r(m) \cos(\omega(n-m)) \\
 &= \sum_m r(m) \cos(\omega m) \cos(\omega n) + \sum_m r(m) \sin(\omega m) \sin(\omega n) \\
 &= c_r(\omega) \cos(\omega n) + s_r(\omega) \sin(\omega n) \\
 &= A_r(\omega) \cos(\phi_r(\omega)) \cos(\omega n) + A_r(\omega) \sin(\phi_r(\omega)) \sin(\omega n) \\
 &= A_r(\omega) \cos(\omega n - \phi_r(\omega))
 \end{aligned}$$

These dot products are the Discrete Fourier Transform of the impulse response, $r(m)$!

Fourier & LSI

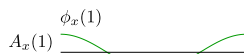
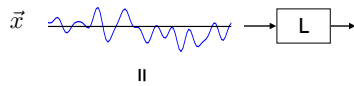


Fourier & LSI



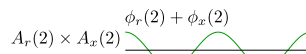
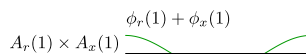
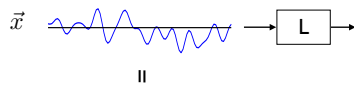
note: only 3 (of many) frequency components shown

Fourier & LSI



note: only 3 (of many) frequency components shown

Fourier & LSI



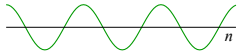
LSI systems are characterized by their *frequency response*, specified by the Fourier Transform of their impulse response

Complex exponentials: “bundling” sine and cosine

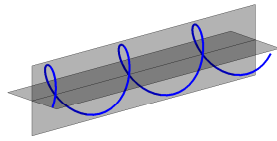
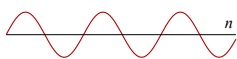
$$e^{i\theta} = \cos(\theta) + i \sin(\theta) \quad \text{“Euler’s formula”}$$

$$Ae^{i\omega n} = A \cos(\omega n) + iA \sin(\omega n)$$

real part:



imaginary part:



[on board: reminders of addition/multiplication of complex numbers]

Complex exponentials: “bundling” sine and cosine

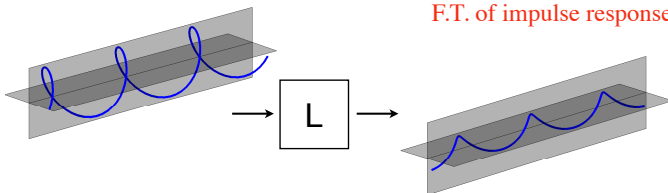
$$e^{i\omega n} \rightarrow \boxed{\text{L}} \rightarrow A_r(\omega) e^{i(\omega n - \phi_r(\omega))} = A_r(\omega) e^{-i\phi_r(\omega)} e^{i\omega n} = \tilde{r}(\omega) e^{i\omega n}$$

F.T. of impulse response!

Complex exponentials: “bundling” sine and cosine

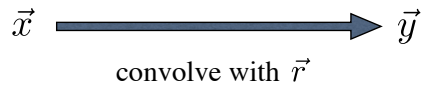
$$e^{i\omega n} \rightarrow \boxed{\text{L}} \rightarrow A_r(\omega) e^{i(\omega n - \phi_r(\omega))} = A_r(\omega) e^{-i\phi_r(\omega)} e^{i\omega n} = \tilde{r}(\omega) e^{i\omega n}$$

F.T. of impulse response!

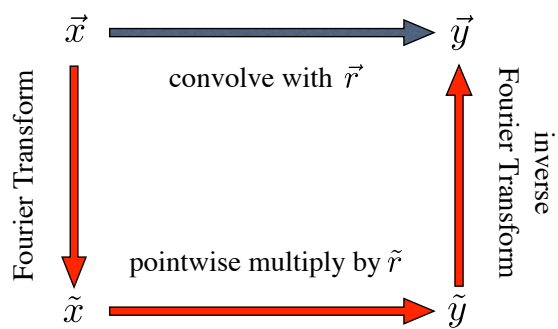


Note: the complex exponentials are *eigenvectors*!

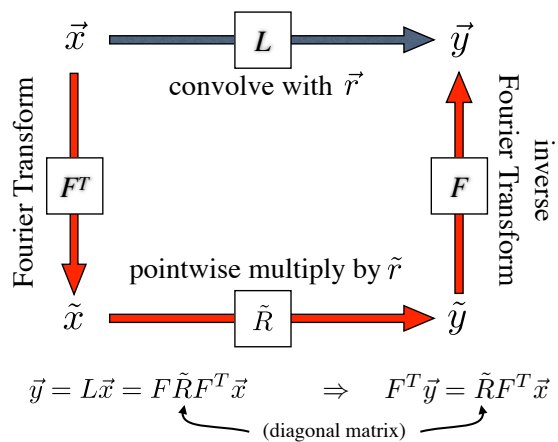
The “convolution theorem”



The “convolution theorem”



The “convolution theorem”



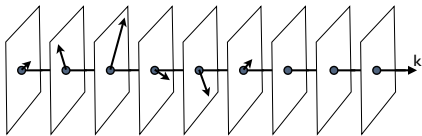
Recap...

- Linear system
 - defined by superposition
 - characterized by a matrix
- Linear Shift-Invariant (LSI) system
 - defined by superposition and shift-invariance
 - characterized by a vector, which can be either:
 - » the impulse response
 - » the frequency response (amplitude and phase). Specifically, the Fourier Transform of the impulse response specifies an amplitude multiplier and a phase shift for each frequency.

Discrete Fourier transform (with complex numbers)

$$\tilde{r}_k = \sum_{n=0}^{N-1} r_n e^{-i\omega_k n} \quad \text{where } \omega_k = \frac{2\pi k}{N}$$

$$r_n = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{r}_k e^{i\omega_k n} \quad (\text{inverse})$$



[on board: why minus sign? why 1/N?]

Visualizing the (Discrete) Fourier Transform

- Two conventional choices for frequency axis:
 - Plot frequencies from $k = 0$ to $k = N/2$
(in matlab: 1 to $N/2+1$)
 - Plot frequencies from $k = -N/2$ to $k = N/2 - 1$
(in matlab: recenter using `fftshift`)
- Typically, we plot amplitude (and optionally, phase), instead of the real/imaginary (cosine/sine) components

Some examples

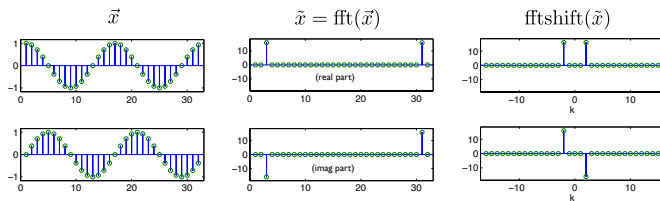
- constant
- sinusoid (see next slide)
- impulse
- Gaussian - “lowpass”
- Derivative of Gaussian - “bandpass”
- DoG (difference of 2 Gaussians) - “bandpass”
- Gabor (Gaussian windowed sinusoid) - “bandpass”

[on board]

$$e^{i\omega n} = \cos(\omega n) + i \sin(\omega n) \quad e^{-i\omega n} = \cos(\omega n) - i \sin(\omega n)$$

$$\Rightarrow \begin{aligned} \cos(\omega n) &= \frac{1}{2}(e^{i\omega n} + e^{-i\omega n}) \\ \sin(\omega n) &= \frac{-i}{2}(e^{i\omega n} - e^{-i\omega n}) \end{aligned}$$

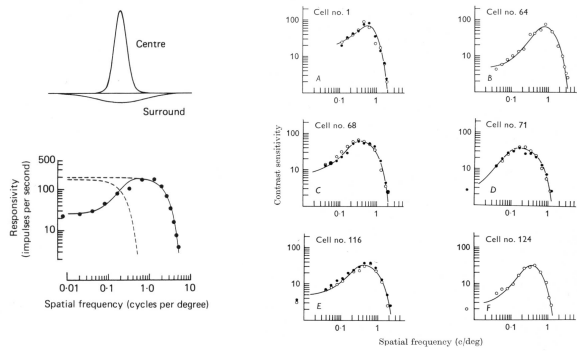
Example for $k=2$, $N=32$ (note indexing and amplitudes):



What do we *do* with Fourier Transforms?

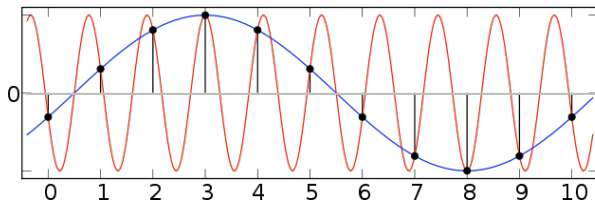
- Represent/analyze periodic *signals*
- Analyze/design LSI *systems*. In particular, how do you identify the nullspace?

Retinal ganglion cells (1D)



Enroth-Cugell and Robson (1984)

Sampling causes “aliasing”

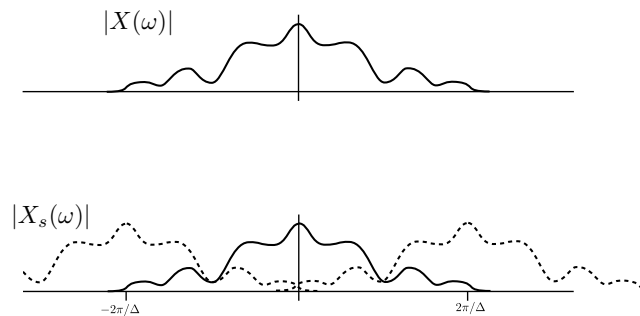


Sampling process is linear, but many-to-one (non-invertible)

“Aliasing” - one frequency masquerades as another *[on board]*

Given the samples, it is common/natural to assume, or enforce, that they arose from the *lowest* compatible frequency...

Effect of sampling on the Fourier Transform:
Sum of shifted copies



Real-world aliasing

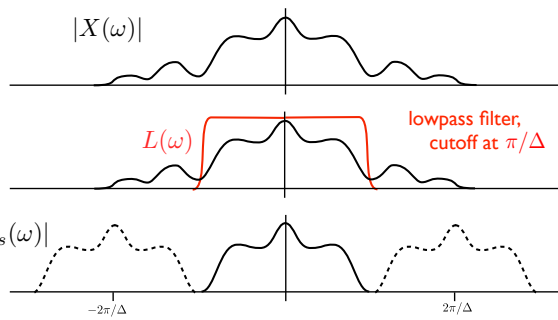
downsample by 2

"Moiré pattern"



Pre-filtering to avoid spectral overlap ("aliasing")

$$X(\omega) \rightarrow L(\omega) \rightarrow \Delta \rightarrow X_s(\omega)$$



Real-world aliasing

downsample by 2,
with pre-filtering

