Module 1: Basics of Digital Signal Processing

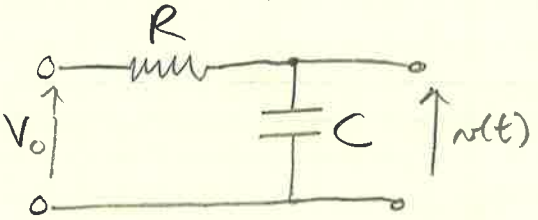
1.1 Introduction to digital signal processing

Signal: Description of evolution of a physical phenomenon

- Weather → temperature
- Sound → pressure
- Sound → magnetic deviation
- Light intensity → gray level on paper

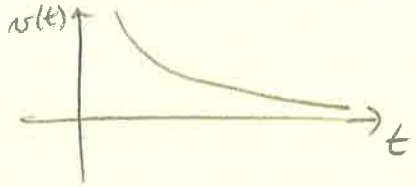
Analysis: understanding the information carried by the signal

Synthesis: creating a signal to contain the given information



2 dof R, C

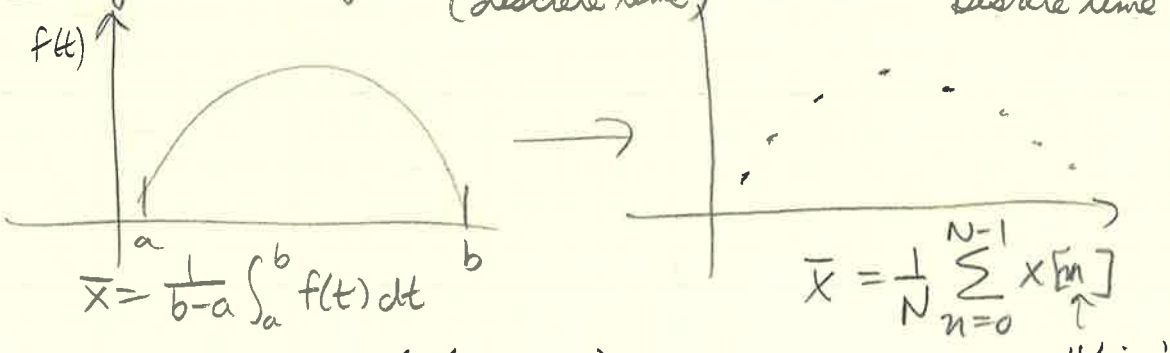
$$v(t) = V_0(1 - e^{-t/RC})$$



Analog signals $f: \mathbb{R} \to \mathbb{V}$

From analog to digital: $f(t) \to$ sample

The digital paradigm



$$\bar{x} = \frac{1}{b-a}\int_a^b f(t)\,dt$$

$$\bar{x} = \frac{1}{N}\sum_{n=0}^{N-1} x[n]$$

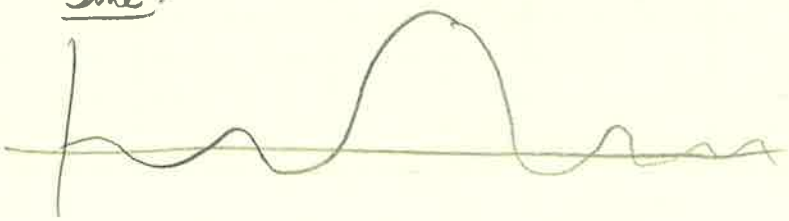
"time"

$x: \mathbb{Z} \to \mathbb{V}$ ($\mathbb{Z}$ or $\mathbb{R}$)

The Sampling Theorem (1920)

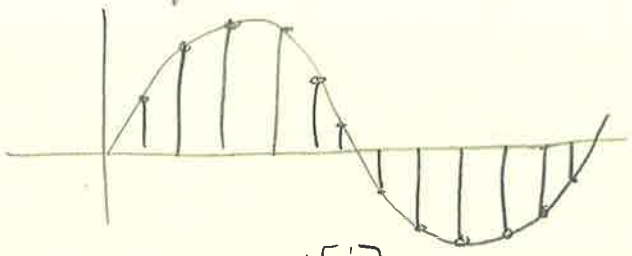
$$x(t) = \sum_{n=-\infty}^{\infty} x[n]\,\mathrm{sinc}\left(\frac{t - nT_s}{T_s}\right)$$

Sinc:

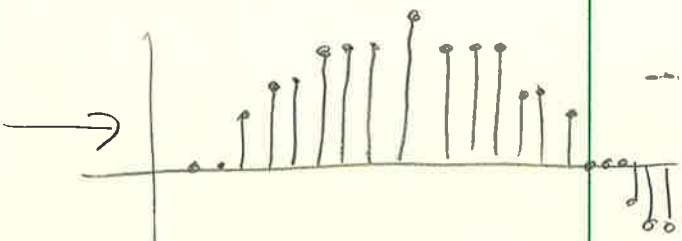


Infinite support

(discrete amplitude)



$x[n]$



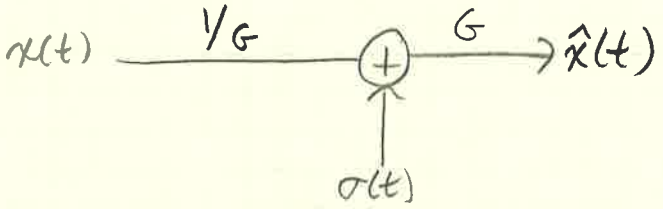
$\hat{x}[n]$
Countable set of levels

Why is it important?
- storage
- processing
- transmission

Digital Storage : $\{0, 1\}$

Data Transmission

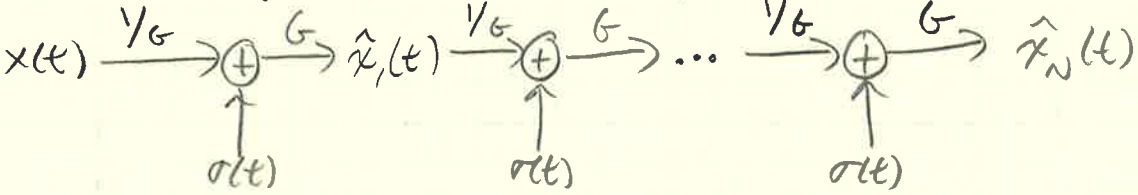
TX → channel → RX



$\hat{x}(t) = x(t)/G + G\sigma(t)$

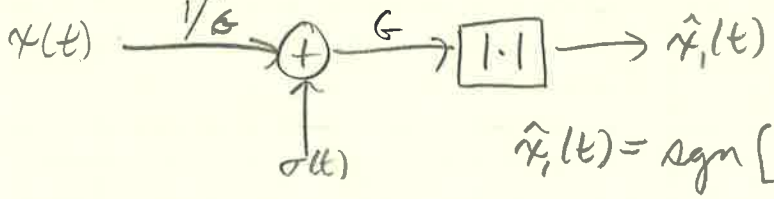
now, it is noisy

For a long channel, we need repeaters

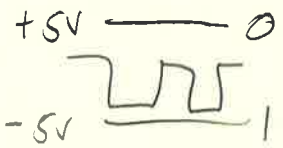


$$\hat{x}_N(t) = x(t) + NG\sigma(t)$$

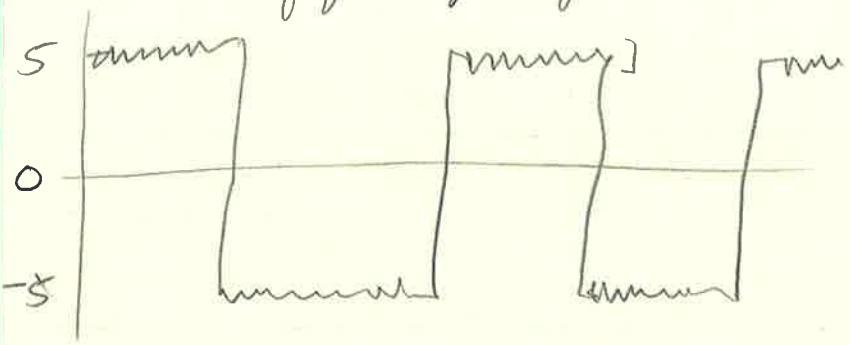
In digital signals, we can threshold



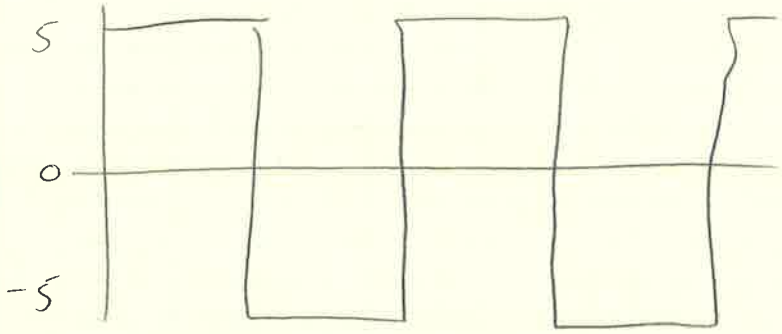
$\hat{x}_1(t) = \text{sgn}[x(t) + G\sigma(t)]$



Transmission of quantized signals



$$G\left(x(t)/G + \sigma(t)\right) = x(t) + G(\sigma(t))$$



$$\hat{x}(t) = G\,\mathrm{sgn}\left[x(t) + G\sigma(t)\right]$$

(after thresholding operation)

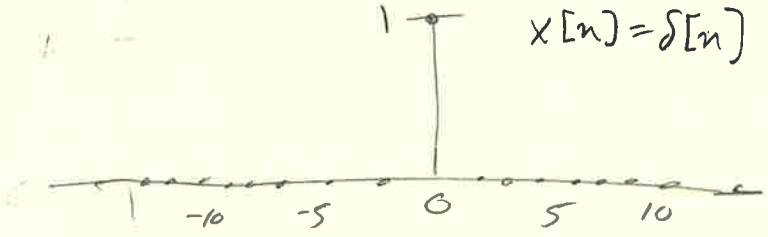
Digital Signal Processing: Key Ideas

- Discretization of time:
  - samples replace idealized models
  - simple maths replaces calculus
- Discretization of values:
  - general-purpose of storage
  - general-purpose processing (CPU)
  - noise can be controlled

1.2 Discrete-time signals

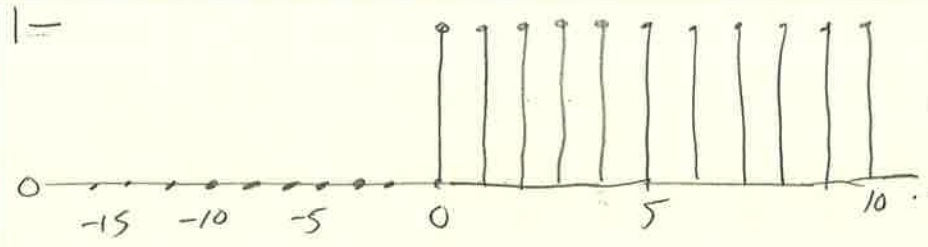
discrete-time signal: a sequence of complex numbers

- One dimension (for now)
- notation: $x[n]$
- two-sided sequences: $x: \mathbb{Z} \to \mathbb{C}$
- n is a-dimensional "time"
- analysis: periodic measurement
- synthesis: stream of generated samples



$$x[n] = \delta[n]$$

Ex: Used to synchronize audio and video in a movie



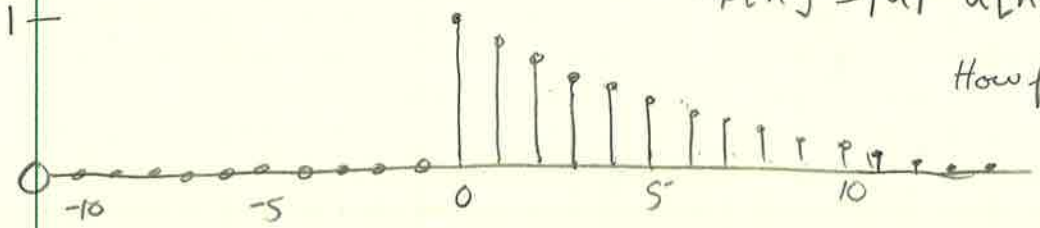
$x[n] = u[n]$

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

(switch)

Exponential decay

$x[n] = |a|^n u[n], \quad |a| < 1$



How fast does coffee get cold?

Newton's Law of cooling $\frac{dT}{dt} = -c(T - T_{env}) \Rightarrow T(t) = T_{env} + (T_0 - T_{env})e^{-ct}$

Sinusoid $x[n] = \sin(\omega_0 n + \theta)$, ω_0, θ in rad

Four signal classes

- finite-length
- infinite-length
- periodic
- finite-support

Finite-length signals

- sequence notation: $x[n]$, $n = 0, 1, \ldots, N-1$
- vector notation: $\overline{x} = [x_0 \; x_1 \; \ldots \; x_{N-1}]^T$
- practical entities, good for numerical packages (e.g. numpy)

Infinite-length signals

- sequence notation: $x[n]$, $n \in \mathbb{Z}$
- abstraction, good for theorems

Periodic signals

- N-periodic sequence: $\tilde{x}[n] = \tilde{x}[n + kN]$, $n, k, N \in \mathbb{Z}$
- same information as finite-length of length N
- "natural" bridge between finite and infinite lengths

Finite-support signals

- Finite-support sequence:

$$\bar{x}[n] = \begin{cases} x[n], & 0 \le n < N \\ 0, & \text{otherwise} \end{cases} \qquad n \in \mathbb{Z}$$

- same information as finite-length of length N
- another bridge between finite and infinite lengths

Elementary operators

- scaling: $y[n] = \alpha x[n]$, $\alpha \in \mathbb{C}$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$

$\Big\}$ $0 \le n \le N-1$

- shift by k (delay): $y[n] = x[n-k]$, $k \in \mathbb{Z}$

Shift of a finite-length: finite-support

$\ldots 0\,0\,0 \underbrace{(x_0\ x_1\ \ldots\ x_7)}_{\bar{x}[n]} 0\,0\,0 \ldots$

$\underbrace{}_{\bar{x}[n-1]}$

$\cdot 0\,0\,0\ (0\ x_0\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6)\ x_7\ 0\ 0 \ldots$

$\underbrace{}_{\bar{x}[n-4]}$

$\cdot 0\,0\,0\ (0\ 0\ 0\ 0\ x_0\ x_1\ x_2\ x_3)\ x_4\ x_5\ x_6\ x_7\ 0\ 0\ 0 \ldots$

Shift of a finite length: periodic extension

$\ldots \underbrace{(x_0\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7)}_{x[n]} \ldots$

$\ldots x_5\ x_6\ x_7\ \underbrace{(x_0\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7)}_{\tilde{x}[n]}\ x_0\ x_1\ x_2 \ldots$

$\ldots x_4\ x_5\ x_6\ \underbrace{(x_7\ x_0\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6)}_{\tilde{x}[n-1]}\ x_7\ x_0\ x_1 \ldots$

$\tilde{x}[n-4]$

$\cdots x_1\ x_2\ x_3\ (x_4\ x_5\ x_6\ x_7\ x_0\ x_1\ x_2\ x_3)\ x_4\ x_5\ x_6 \cdots$

Energy and power

$$E_x = \sum_{n=-\infty}^{\infty} |x[n]|^2$$

$$P_x = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2$$

Energy and power: periodic signals

$$E_{\tilde{x}} = \infty$$

$$P_{\tilde{x}} \equiv \frac{1}{N} \sum_{n=0}^{N-1} |\tilde{x}[n]|^2$$

1.3 Basic signal processing

1.3.a How your PC plays discrete-time sounds

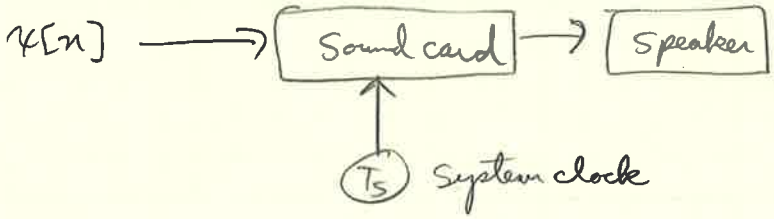
The discrete-time sinusoid

$$x[n] = \sin(\omega_0 n + \theta)$$

Digital vs. physical frequency

- Discrete time:
  - no physical dimension (just a counter)
  - periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

How your PC plays sounds



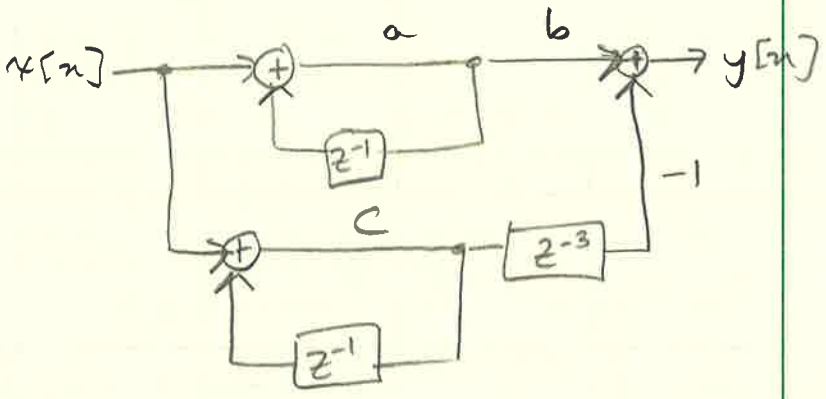
- set T_s, time in seconds between samples
- periodicity of M samples → periodicity of MT_s seconds
- real world frequency: $f = \frac{1}{MT_s}$ Hz

- usually we choose F_S, the number of samples per second
- $T_S = 1/F_S$

Eg. for a typical value, $F_S = 48000$ Hz, $T_S \approx 20.8\,\mu s$.
If $M = 110$, $f \approx 440$ Hz

1.3b The Karplus-Strong algorithm

DSP as Meccano



Building blocks:

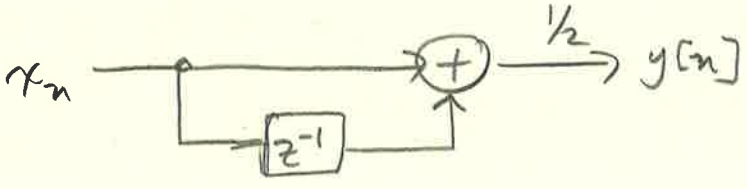
- Adder: $x[n]$, $y[n]$ → $x[n]+y[n]$
- Multiplier: $x[n] \xrightarrow{\alpha} \alpha x[n]$
- Unit Delay: $x[n] \longrightarrow [z^{-1}] \longrightarrow x[n-1]$
- Arbitrary Delay: $x[n] \longrightarrow [z^{-N}] \longrightarrow x[n-N]$

The 2-point Moving Average

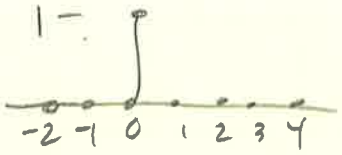
- simple average: $m = \frac{a+b}{2}$
- moving average: take a "local" average

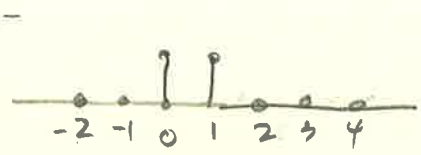
$$y[n] = \frac{x[n]+x[n-1]}{2}$$

- DSP Blocks:



Ex: $x[n] = \delta[n]$





$$y[0] = \frac{x[0]+x[-1]}{2} = \frac{1}{2}$$

$$y[1] = \frac{x[1]+x[0]}{2} = \frac{1}{2}$$

- $x[n] = u[n]$

$$y[0] = \frac{x[0]+x[-1]}{2} = \frac{1+0}{2} = \frac{1}{2}$$

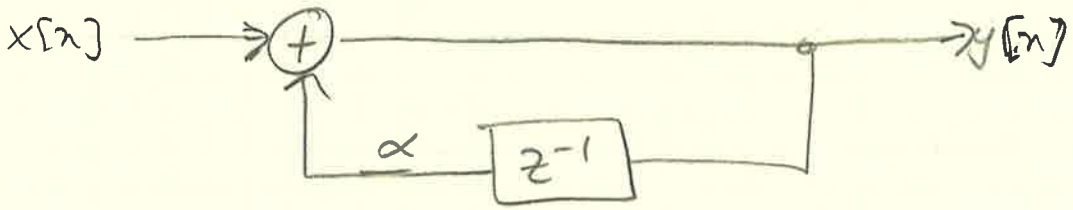
$$y[1] = \frac{x[1]+x[0]}{2} = \frac{1+1}{2} = 1$$

- $x[n] = \cos(\omega n)$, $\omega = \pi/10$

$$y[n] = \frac{\cos \omega n - \cos \omega(n-1)}{2} = \cos(\omega n + \theta)$$

- $x[n] = (-1)^n \Rightarrow y[n] = 0, \forall n$

What if we reverse the loop?



$$y[n] = x[n] + \alpha y[n-1], \quad \alpha \in \mathbb{R}$$

(recursion)

How we solve the chicken-and-egg problem

Zero Initial conditions

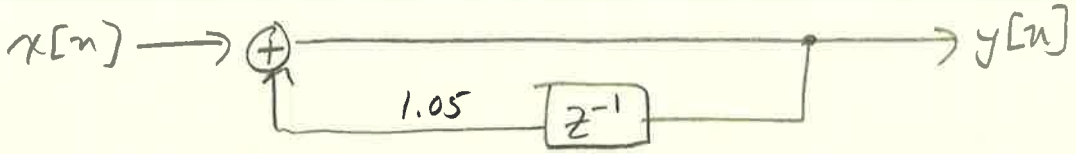
- set a start time (usually $n_0 = 0$)
- assume input and output are zero for all times before n_0

Ex: A simple model for banking

A simple equation to describe compound interest:

- constant interest/borrowing rate of 5% per year
- interest accrues on Dec 31
- deposits/withdrawals during year n: $x[n]$
- balance at year n:

$$y[n] = 1.05\, y[n-1] + x[n]$$

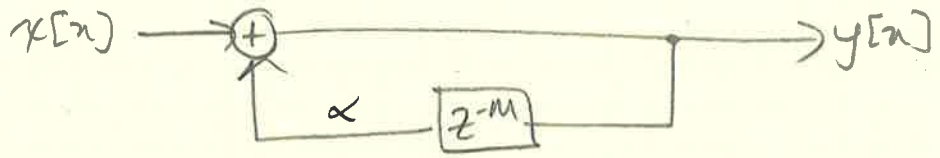


$$y[n] = 1.05\,y[n-1] + x[n]$$

Ex: One-time investment $x[n] = 100\,\delta[n]$

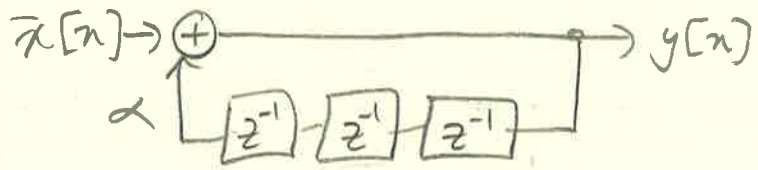
- $y[0] = 100$
- $y[1] = 105$
- $y[2] = 110.25$, $y[3] = 115.7625$, etc.
- In general: $y[n] = (1.05)^n \cdot 100\,u[n]$

An interesting generalization



$$y[n] = \alpha\,y[n-M] + x[n]$$

- Creating loops



$$y[n] = \alpha\,y[n-3] + x[n]$$

Ex: $M = 3$, $\alpha = 0.7$, $x[n] = \delta[n]$

- $y[0] = 1$, $y[1] = 0$, $y[2] = 0$
- $y[3] = 0.7$, $y[4] = 0$, $y[5] = 0$
- $y[6] = 0.7^2$, $y[7] = 0$, $y[8] = 0$, etc.

Ex: $M = 3$, $\alpha = 1$, $x[n] = \delta[n] + 2\delta[n-1] + 3\delta[n-2]$

- $y[0] = 1$, $y[1] = 2$, $y[2] = 3$
- $y[3] = 1$, $y[4] = 2$, $y[5] = 3$
- $y[6] = 1$, $y[7] = 2$, $y[8] = 3$, etc.

We can make music with that!

- build a recursion loop with a delay of M
- choose a signal $\bar{x}[n]$ that is nonzero only for $0 \le n < M$
- choose a decay factor
- input $\bar{x}[n]$ to the system
- play the output

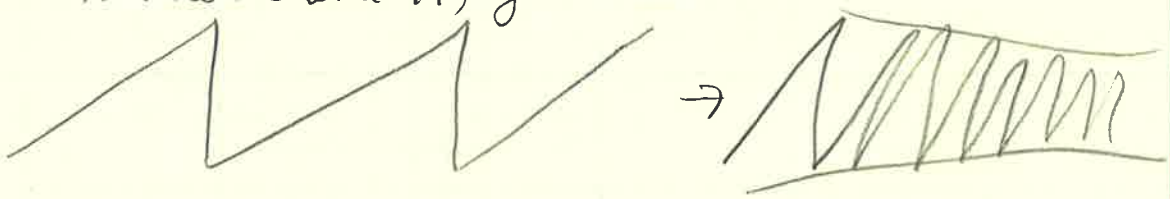
Ex: $M=100$, $\alpha=1$, $\bar{x}[n] = \sin(2\pi n/100)$ for $0 \le n < 100$ and zero elsewhere

$F_S = 48\,kHz \rightarrow 480\,Hz$

Introducing some realism

- M controls frequency (pitch)
- α controls envelope (decay)
- $\bar{x}[n]$ controls color (timbre)

Photo-reden: $M=100$, $\alpha=0.95$, $\bar{x}[n]$: zero-mean sawtooth wave between 0 and 99, zero elsewhere



The Karplus-Strong Algorithm

$M=100$, $\alpha=0.9$, $\bar{x}[n]$: 100 random values between 0 and 99 (in $[-1,1]$), zero elsewhere.

Similar to a harpsichord.

Signal of the Day: Goethe's Temperature Measurement

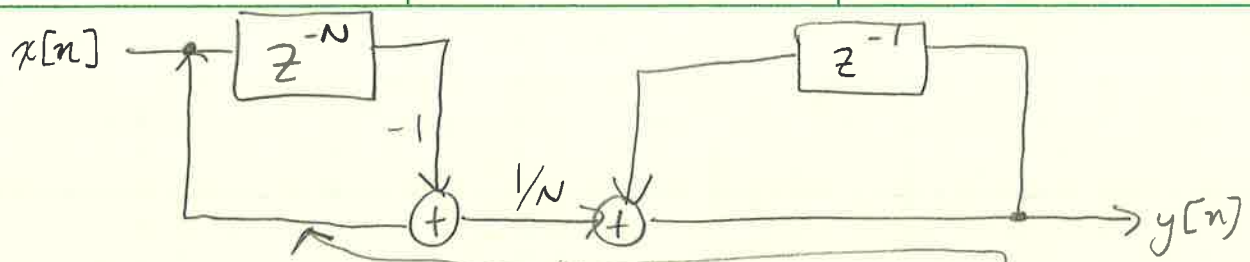
Smoothing:
- Moving average: $y[n] = \frac{1}{N}\sum_{m=0}^{N-1} x[n-m]$
- N: window of last observations over which the average is computed

A recursive method

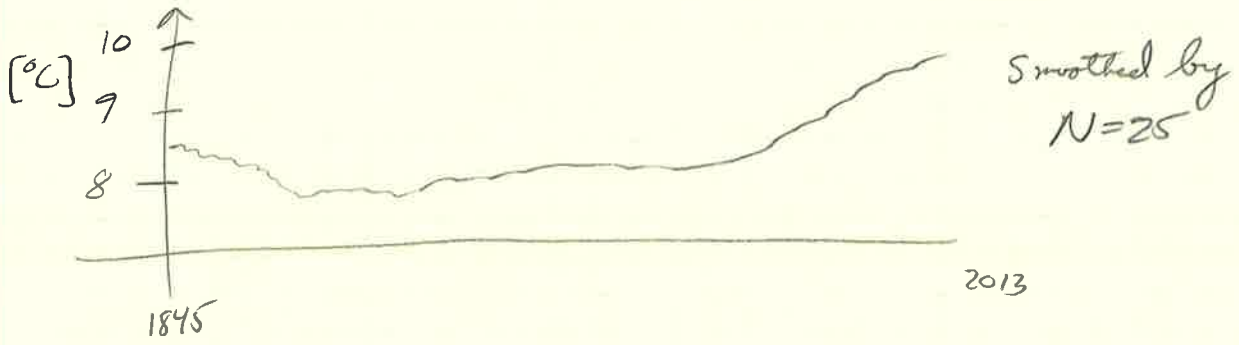
$$y[n] = \frac{1}{N}\sum_{m=0}^{N-1} x[n-m]$$

$$= \frac{1}{N}x[n] + \underbrace{\frac{1}{N}\sum_{m=1}^{N-1} x[n-m]}_{y[n-1]} + \frac{1}{N}x[n-N] - \frac{1}{N}x[n-N]$$

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



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

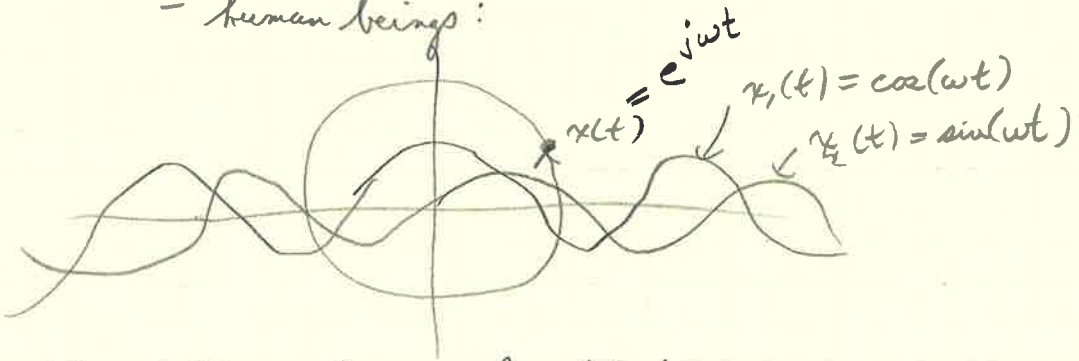


1.4 Complex exponentials

$j = \sqrt{-1}$

Oscillations are everywhere!

- Sustainable dynamic systems exhibit oscillatory behavior
- Intuitively: things that don't move in circles can't last:
  - bombs
  - rockets
  - human beings:



- The discrete-time oscillatory heartbeat

Ingredients:

- a frequency ω (units: radians)
- an initial phase ϕ (units: radians)
- an amplitude A

$$x[n] = Ae^{j(\omega n + \phi)}$$
$$= A\left[\cos(\omega n + \phi) + j\sin(\omega n + \phi)\right]$$

Why complex exponentials?

- we can use complex numbers in digital systems, so why not?
- it makes sense: every sinusoid can always be written as a sum of sine and cosine
- math is simpler: trigonometry becomes algebra

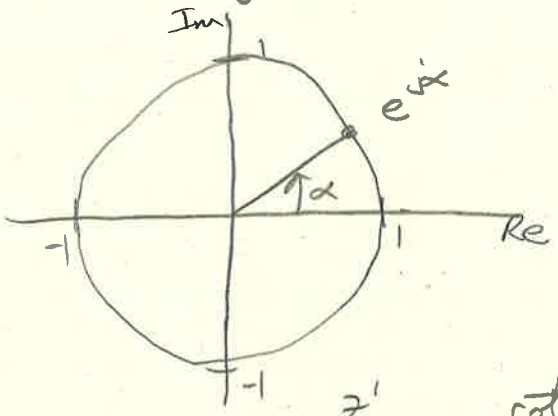
Ex: change the phase of a cosine the "old-school" way

$$\cos(\omega n+\phi) = a\cos(\omega n) - b\sin(\omega n), \quad a=\cos\phi,\ b=\sin\phi$$

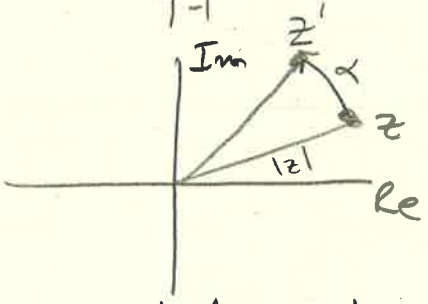
use complex exponentials

$$\cos(\omega n+\phi) = \mathrm{Re}\left[e^{j(\omega n+\phi)}\right] = \mathrm{Re}\left[e^{j\omega n}e^{j\phi}\right]$$

$$e^{j\alpha} = \cos\alpha + j\sin\alpha$$



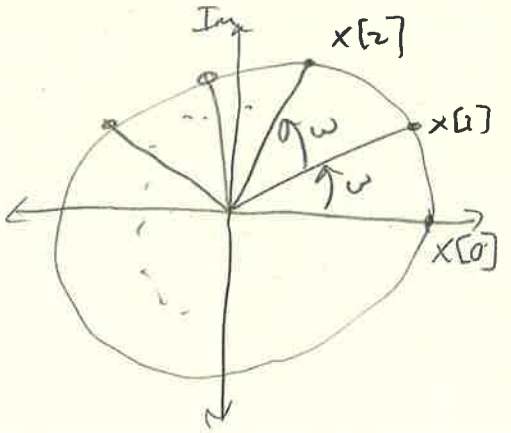
$|e^{j\alpha}| = 1$



rotation $z' = ze^{j\alpha}$

The complex exponential generating machine

$$x[n] = e^{j\omega n}; \quad x[n+1] = e^{j\omega}x[n]$$

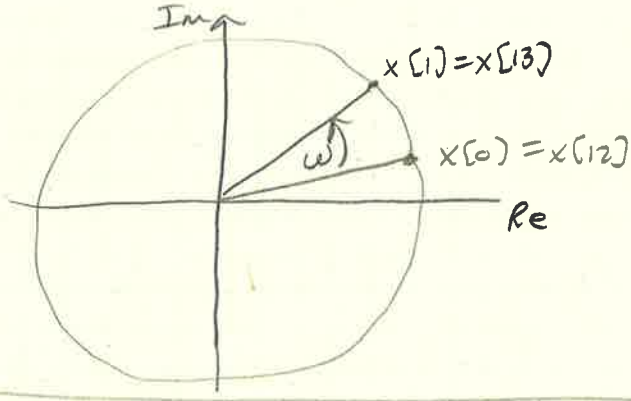


$x[0] = 1$

$\omega = \frac{2\pi}{12}$

Initial phase

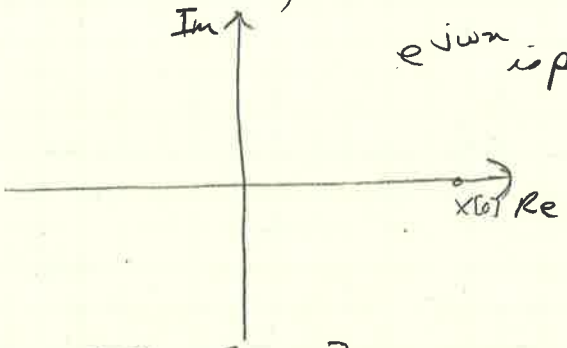
$x[n] = e^{j(\omega n + \phi)}$; $x[n+1] = e^{j\omega} x[n]$, $x[0] = e^{j\phi}$



$\omega = \frac{2\pi}{12}$

Careful: not every sinusoid is periodic in discrete time

$x[n] = e^{j\omega n}$; $x[n+1] = e^{j\omega} x[n]$



$e^{j\omega n}$ is periodic in $n \Leftrightarrow \omega = \frac{M}{N} 2\pi$, $M, N \in \mathbb{Z}$

$x[n] = x[n+N]$

$e^{j(\omega n + \phi)} = e^{j(\omega(n+N)+\phi)}$

$e^{j\omega n} e^{j\phi} = e^{j\omega n} e^{j\omega N} e^{j\phi}$

$e^{j\omega N} = 1 \Leftrightarrow \omega N = 2M\pi, \; M \in \mathbb{Z}$

$\omega = \frac{M}{N} 2\pi$

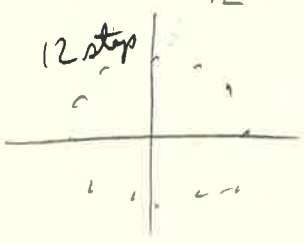
2π-periodicity : one point, many names

$e^{j\alpha} = e^{j(\alpha + 2\pi k)}$, $\forall k \in \mathbb{Z}$

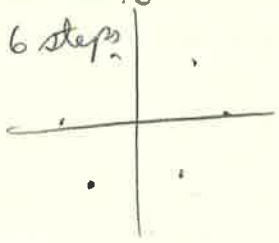
One point, many names : Aliasing

How "fast" can we go?

$\omega = \frac{2\pi}{12}$ — 12 steps



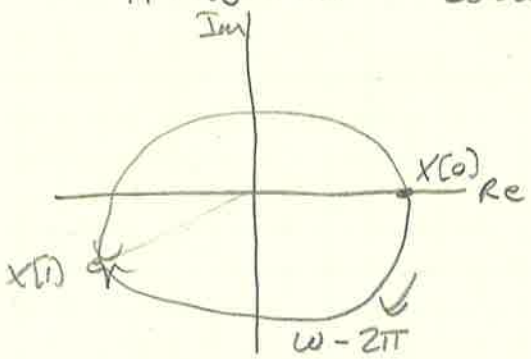
$\omega = 2\pi/6$ — 6 steps



$\omega = 2\pi/2$ — 2 steps

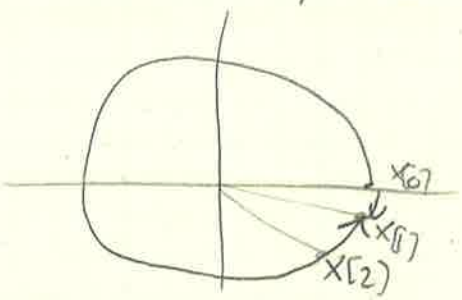
What if we go faster?

$\pi < \omega < 2\pi$ corresponds to going slower in opposite direction



$\omega = 2\pi - \alpha$, α small

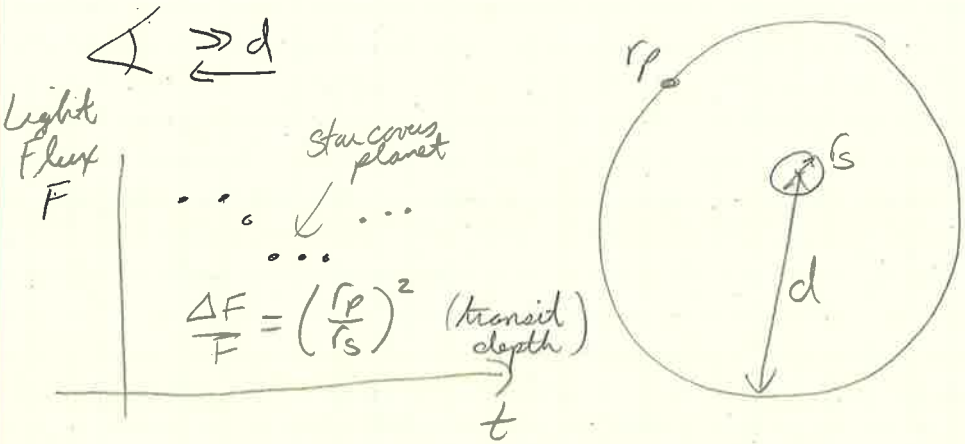
very slow in opposite direction



2.1 Signal processing and vector spaces

Common framework: vector space
- vector spaces are very general objects
- vector spaces are defined by their properties
- once you know the properties are satisfied, you can use all the tools for the space

Signal of the day: exoplanet hunting



- Earth: $\frac{\Delta F}{F} = \left(\frac{r_p}{r_s}\right)^2 = \left(\frac{6,371}{696,000}\right)^2 \approx 0.01\%$
- Jupiter: $\frac{\Delta F}{F} = \left(\frac{69,911}{696,000}\right)^2 \approx 1\%$
- Best telescope today can detect a transit depth of 0.1%

2.2 Vector Spaces

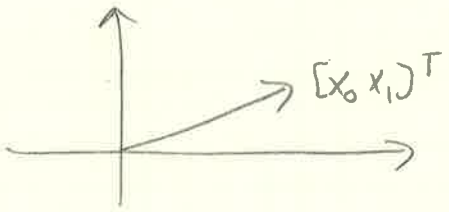
2.2a Vector space

Some familiar examples
- $\mathbb{R}^2, \mathbb{R}^3$: Euclidean space
- $\mathbb{R}^N, \mathbb{C}^N$: linear algebra

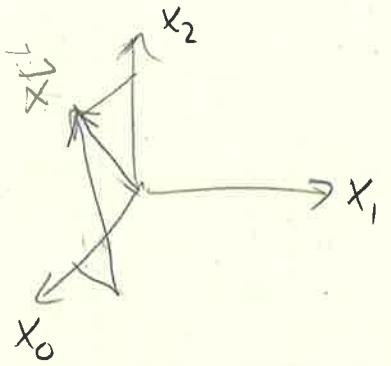
Other examples
- $l_2(\mathbb{R})$: space of square-summable infinite sequences
- $L_2([a,b])$: space of square-integrable functions over an interval

Some can be represented geometrically

$\mathbb{R}^2$: $\vec{x} = [x_0\ x_1]^T$

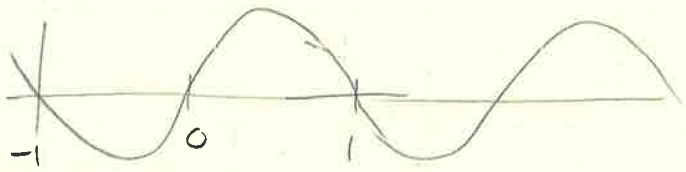


$\mathbb{R}^3$: $\vec{x} = [x_0\ x_1\ x_2]^T$



$L_2([-1,1])$: $\vec{x} = x(t)$, $t \in [-1,1]$

$\vec{x} = \sin(\pi t)$



Can't plot $\mathbb{R}^N$, $N > 3$ or $\mathbb{C}^N$, $N > 1$

Ingredients

- the set of vectors V
- a set of scalars (say $\mathbb{C}$)

We need at least to be able to:

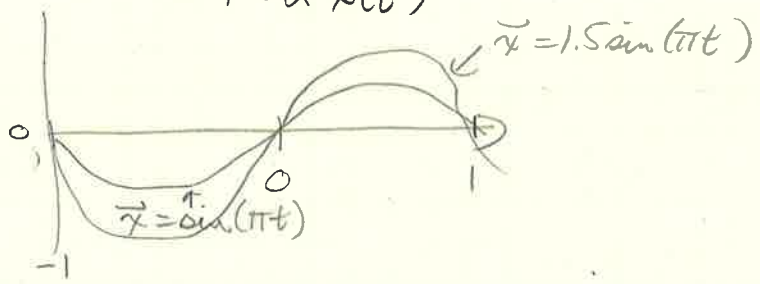
- resize vectors i.e., multiply a vector by a scalar
- combine vectors together, i.e., sum them

Formal properties: For $\vec{x}, \vec{y}, \vec{z} \in V$ and $\alpha, \beta \in \mathbb{C}$:

- $\vec{x} + \vec{y} = \vec{y} + \vec{x}$
- $(\vec{x} + \vec{y}) + \vec{z} = \vec{x} + (\vec{y} + \vec{z})$
- $\alpha(\vec{x} + \vec{y}) = \alpha\vec{y} + \alpha\vec{x}$
- $(\alpha + \beta)\vec{x} = \alpha\vec{x} + \beta\vec{x}$
- $\alpha(\beta\vec{x}) = (\alpha\beta)\vec{x}$
- $\exists 0 \in V : \vec{x} + 0 = 0 + \vec{x} = \vec{x}$
- $\forall \vec{x} \in V, \exists (-\vec{x}) : \vec{x} + (-\vec{x}) = 0$

Scalar multiplication in $L_2[-1,1]$

$$\alpha \vec{x} = \alpha x(t)$$



We need something more: inner product (aka dot product)

$$\langle \cdot, \cdot \rangle : V \times V \to \mathbb{C}$$

- measure of similarity between vectors
- inner product is zero? vectors are orthogonal (maximally different)

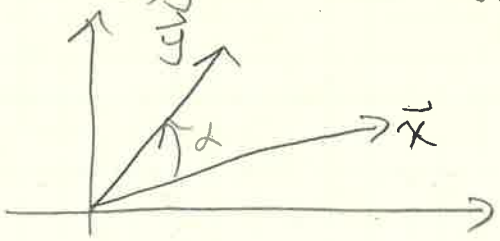
Formal properties of the inner product

For $\vec{x}, \vec{y}, \vec{z} \in V$, $\alpha \in \mathbb{C}$:

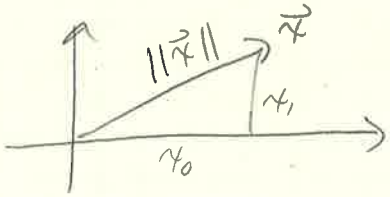
- $\langle \vec{x}+\vec{y}, \vec{z} \rangle = \langle \vec{x}, \vec{z} \rangle + \langle \vec{y}, \vec{z} \rangle$
- $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle^*$
- $\langle \alpha \vec{x}, \vec{y} \rangle = \alpha^* \langle \vec{x}, \vec{y} \rangle$
- $\langle \vec{x}, \alpha \vec{y} \rangle = \alpha \langle \vec{x}, \vec{y} \rangle$
- $\langle \vec{x}, \vec{x} \rangle \geq 0$

- $\langle \vec{x}, \vec{x} \rangle = 0 \iff \vec{x} = 0$
- If $\langle \vec{x}, \vec{y} \rangle = 0$ and $\vec{x}, \vec{y} \neq 0$, then $\vec{x}$ and $\vec{y}$ are called orthogonal

$$\langle \vec{x}, \vec{y} \rangle = x_0 y_0 + x_1 y_1 = \|\vec{x}\| \, \|\vec{y}\| \cos\alpha$$



$$\langle x, x \rangle = x_0^2 + x_1^2 = \|\vec{x}\|^2$$

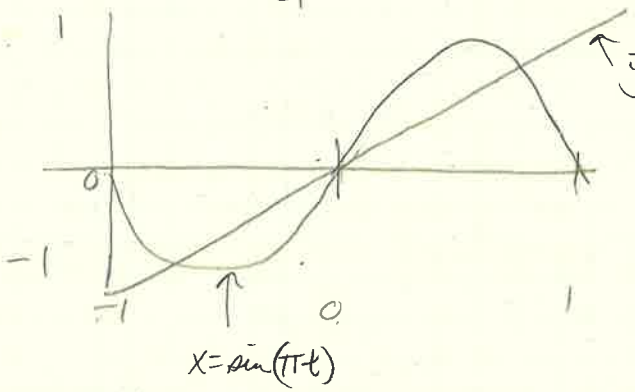


Inner product in $L_2[-1,1]$

$$\langle \vec{x}, \vec{y} \rangle = \int_{-1}^{1} x(t)\, y(t)\, dt$$

$$\| \sin(\pi t) \|^2 = \int_{-1}^{1} \sin^2 \pi t \, dt = 1$$

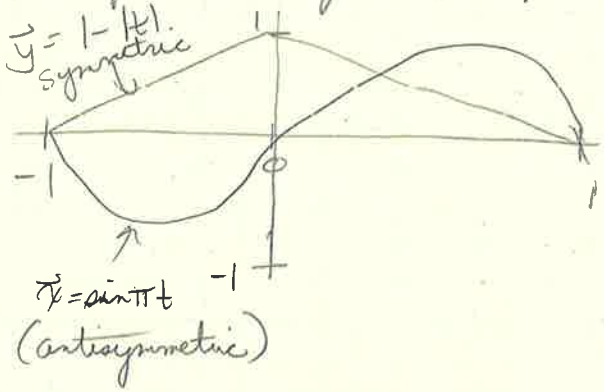
$\vec{y} = t$: $\quad \|\vec{y}\|^2 = \int_{-1}^{1} t^2 dt = \frac{2}{3}$



$$\langle \vec{x}, \vec{y} \rangle = \int_{-1}^{1} \sqrt{\tfrac{3}{2}}\, t \sin \pi t \, dt$$

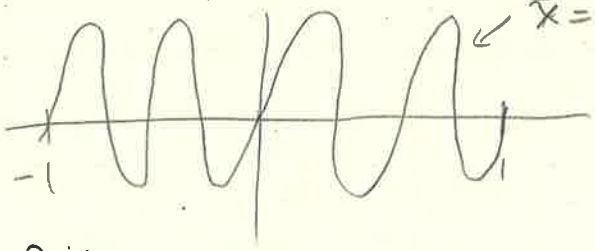
$$= \frac{2}{\pi}\sqrt{\frac{3}{2}} \approx 0.78$$

$\vec{x}, \vec{y}$ from orthogonal subspaces:



$$\langle \vec{x}, \vec{y} \rangle = 0$$

Sinusoids with frequencies that are integer multiples of a fundamental



$\vec{y} = \sin 5\pi t$

$$\langle \vec{x}, \vec{y} \rangle = 0$$

Norm vs Distance

- inner product defines a norm: $\|\vec{x}\| = \sqrt{\langle \vec{x}, \vec{x} \rangle}$
- norm defines a distance: $d(\vec{x}, \vec{y}) = \|\vec{x} - \vec{y}\|$

Distance in $L_2[-1,1]$: the Mean Square Error

$$\|\vec{x} - \vec{y}\|^2 = \int_{-1}^{1} |x(t) - y(t)|^2 dt$$

$\vec{x} = \sin 4\pi t$, $\vec{y} = \sin 5\pi t$,, $\|\vec{x}-\vec{y}\|^2 = \int_{-1}^{1} |\sin 4\pi t - \sin 5\pi t|^2 dt = 2$

2.2.b Signal Spaces

Finite-Length Signals

finite-length and periodic signals live in $\mathbb{C}^N$

- vector notation: $\vec{x} = [x_0 \; x_1 \; \dots \; x_{N-1}]^T$
- all operations well-defined and intuitive
- space of N-periodic signals sometimes indicated by $\tilde{\mathbb{C}}^N$

Inner product for signals

$$\langle \vec{x}, \vec{y} \rangle = \sum_{n=0}^{N-1} x^*[n]\, y[n]$$

well-defined for all finite-length vectors

Infinite Signals?

$$\langle \vec{x}, \vec{y} \rangle = \sum_{n=-\infty}^{\infty} x^*[n]\, y[n]$$

We require sequences to be square-summable: $\sum |x[n]|^2 < \infty$

i.e. in $\ell_2(\mathbb{Z})$ (finite-energy)

Many interesting signals are not in $\ell_2(\mathbb{Z})$, such as,

$x[n] = 1$, $x[n] = \cos(\omega n)$, etc.

Completeness

Limiting operations must yield vector space elements

An incomplete space: $\mathbb{Q}$ $\quad x_n = \sum_{k=0}^{n} \frac{1}{k!} \in \mathbb{Q}$,

but $\lim_{n\to\infty} x_n = e \notin \mathbb{Q}$

Hilbert Space

1. a vector space: $H(V, \mathbb{C})$
2. an inner product: $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{C}$
3. Complete

Bases

Linear combination is the basic operation in vector spaces:

$$\vec{g} = \alpha \vec{x} + \beta \vec{y}$$

Can we find a set of vectors $\{\vec{w}^{(k)}\}$ so that we can write any vector as a linear combination of the $\{\vec{w}^{(k)}\}$?

Canonical $\mathbb{R}^2$ basis

$$\vec{e}^{(0)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \vec{e}^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = x_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + x_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Another $\mathbb{R}^2$ basis

$$\vec{v}^{(0)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \vec{v}^{(1)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \alpha_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \alpha_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Rightarrow \alpha_0 = x_0 - x_1, \; \alpha_1 = x_1$$

Not a basis for $\mathbb{R}^2$

$$\vec{g}^{(0)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \vec{g}^{(1)} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$ not linearly independent

What about infinite-dimensional spaces?

$$\vec{x} = \sum_{k=0}^{\infty} \alpha_k \vec{w}^{(k)}$$

a basis for $l_2(\mathbb{Z})$

$$\vec{e}^{(k)} = \begin{bmatrix} \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix}, \quad 1 \text{ in } k^{\text{th}} \text{ position}, \; k \in \mathbb{Z}$$

What about function vector spaces?

$$f(t) = \sum_{k} \alpha_k h^{(k)}(t)$$

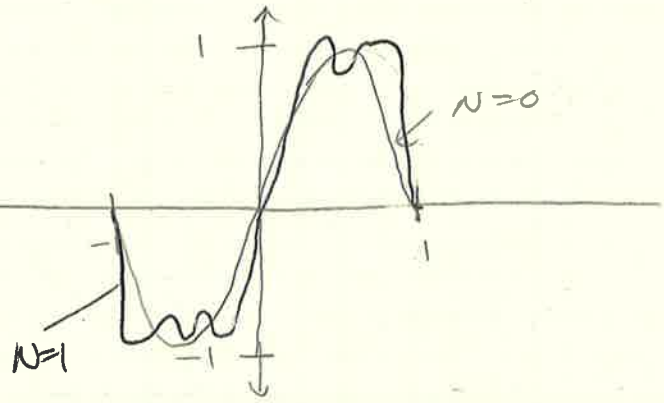
A basis for the functions over an interval?

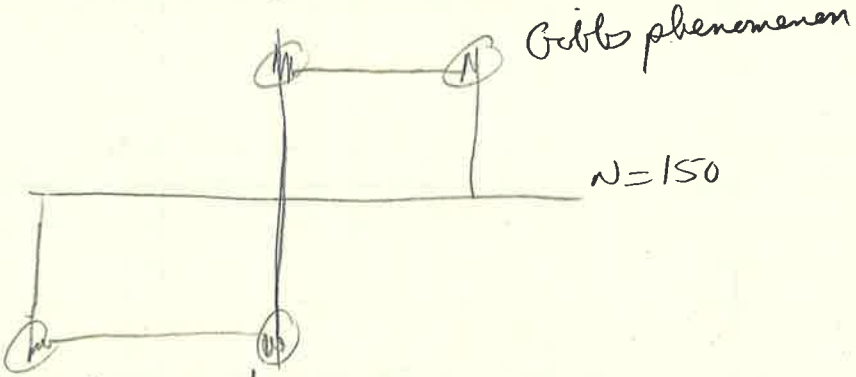
the Fourier basis for $[-1, 1]$

$$\left\{ \frac{1}{\sqrt{2}}, \cos \pi t, \sin \pi t, \cos 2\pi t, \sin 2\pi t, \cos 3\pi t, \sin 3\pi t, \ldots \right\}$$

Using the Fourier Basis (approximating a square wave)

$$\sum_{k=0}^{N} \frac{\sin(2k+1)\pi t}{2k+1} = \sum_{k=0}^{N} \frac{w^{(4k+2)}}{2k+1}$$





Bases: formal definition

Given:

- a vector space H
- a set of K vectors from H: $W = \{\vec{w}^{(k)}\}_{k=0,1,\ldots,K-1}$

W is a basis for H if:

1. We can write for all $x \in H$:

$$\vec{x} = \sum_{k=0}^{K-1} \alpha_k \vec{w}^{(k)}, \quad \alpha_k \in \mathbb{C}$$

2. the coefficients α_k are unique

Uniqueness implies linear independence

$$\sum_{k=0}^{K-1} \alpha_k \vec{w}^{(k)} = 0 \iff \alpha_k = 0, \quad k = 0, 1, \ldots, K-1$$

Special bases

Orthogonal basis:

$$\langle \vec{w}^{(k)}, \vec{w}^{(n)} \rangle = 0, \quad k \neq n$$

Orthonormal basis : $\langle \vec{w}^{(k)}, \vec{w}^{(n)} \rangle = \delta[n-k]$

We can use Gram-Schmidt to normalize any orthogonal basis

Basis expansion

$$\vec{x} = \sum_{k=0}^{K-1} \alpha_k \vec{w}^{(k)}$$, how do we find the α's ?

Orthonormal bases are the best : $\alpha_k = \langle \vec{w}^{(k)}, \vec{x} \rangle$

Change of basis

$$\vec{x} = \sum_{k=0}^{K-1} \alpha_k \vec{w}^{(k)} = \sum_{k=0}^{K-1} \beta_k \vec{v}^{(k)}$$

If $\{\vec{v}^{(k)}\}$ is orthonormal :

$$\beta_h = \langle \vec{v}^{(h)}, \vec{x} \rangle$$

$$= \langle \vec{v}^{(h)}, \sum_{k=0}^{K-1} \alpha_k \vec{w}^{(k)} \rangle = \sum_{k=0}^{K-1} \alpha_k \langle \vec{v}^{(h)}, \vec{w}^{(k)} \rangle$$

$$= \sum_{k=0}^{K-1} \alpha_k C_{hk}$$

$$= \begin{bmatrix} C_{00} & C_{01} & \cdots & C_{0(K-1)} \\ & & \vdots & \\ C_{(K-1)0} & C_{(K-1)1} & \cdots & C_{(K-1)(K-1)} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{K-1} \end{bmatrix}$$

Change of basis : example

- canonical basis $E = \{\vec{e}^{(0)}, \vec{e}^{(1)}\}$
- $\vec{x} = \alpha_0 \vec{e}^{(0)} + \alpha_1 \vec{e}^{(1)}$
- new basis $V = \{\vec{v}^{(0)}, \vec{v}^{(1)}\}$ with $\vec{v}^{(0)} = [\cos\theta \ \sin\theta]^T$, $\vec{v}^{(1)} = [-\sin\theta \ \cos\theta]^T$

$$\vec{x} = \beta_0 \vec{v}^{(0)} + \beta_1 \vec{v}^{(1)}$$

- new basis is orthonormal : $C_{hk} = \langle \vec{v}^{(h)}, \vec{e}^{(k)} \rangle$
- in compact form :

$$\begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} = R\vec{x}$$

- R : Rotation matrix
- $R^T R = I$

Vector Subspace

- A subset of vectors closed under addition and scalar multiplication
- Example: $\mathbb{R}^2 \subset \mathbb{R}^3$

- Subspace of symmetric functions over $L_2[-1,1]$

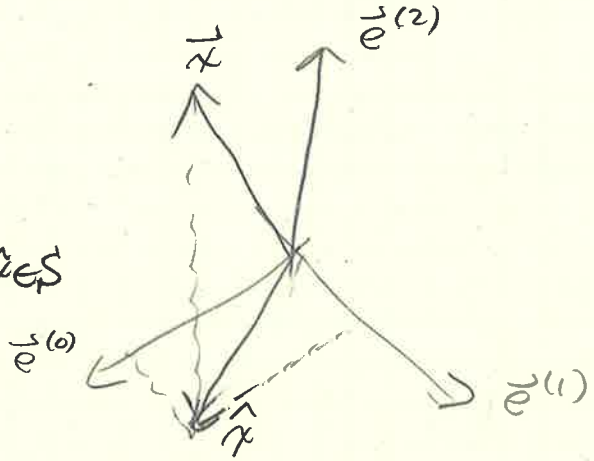
$\vec{x} = \cos \pi t$
$\vec{y} = \cos 5\pi t$ to name a couple

- Subspaces have their own bases

$\left\{ \vec{e}^{(0)} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \vec{e}^{(1)} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \right\}$ basis for a plane

Approximation

Problem:
- vector $x \in V$
- subspace $S \subseteq V$
- approximate $\vec{x}$ with $\hat{x} \in S$



Least-Squares Approximation

- $\{\vec{s}^{(k)}\}_{k=0,1,\ldots,K-1}$ orthonormal basis for S
- orthogonal projection:

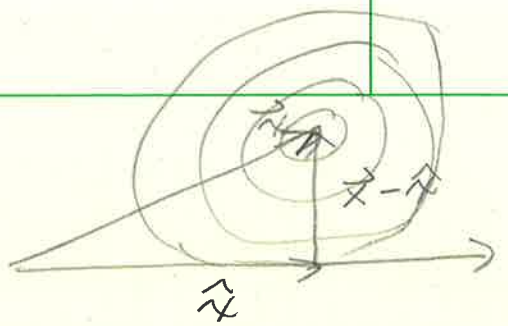
$$\hat{x} = \sum_{k=0}^{K-1} \langle \vec{s}^{(k)}, \hat{x} \rangle \vec{s}^{(k)}$$

- orthogonal projection is the "best" approximation over S
- orthogonal projection has minimum-norm error:

$$\operatorname{argmin}_{\vec{y} \in S} \|\vec{x} - \vec{y}\| = \hat{x}$$

- error is orthogonal to approximation:

$$\langle \vec{x} - \hat{x}, \hat{x} \rangle = 0$$



draw concentric circles until hitting $\vec{s}$. This radius vector is $\vec{x}-\hat{x}$.

Example: polynomial approximation

- vector space $P_N[-1,1] \subset L_2[-1,1]$
- $\vec{p} = a_0 + a_1 t + \ldots + a_{N-1} t^{N-1}$
- a self-evident, naive basis: $\vec{s}^{(k)} = t^k$, $k = 0, 1, \ldots, N-1$
- naive basis is not orthonormal

goal: approximate $\vec{x} = \sin t \in L_2[-1,1]$ over $P_3[-1,1]$

- build orthonormal basis from naive basis
- project $\vec{x}$ over the orthonormal basis
- compute approximation error
- compare error to Taylor approximation (well known but not optimal over the interval)

Building an orthonormal basis

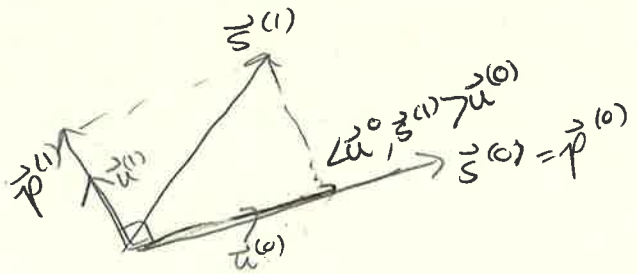
Gram-Schmidt orthonormalization procedure:

$\{\vec{s}^{(k)}\} \longrightarrow \{\vec{u}^{(k)}\}$

original set → orthonormal set

Algorithmic procedure: at each step k

1. $\vec{p}^{(k)} = \vec{s}^{(k)} - \sum_{n=0}^{k-1} \langle \vec{u}^{(n)}, \vec{s}^{(k)} \rangle \vec{u}^{(n)}$
2. $\vec{u}^{(k)} = \vec{p}^{(k)} / \| \vec{p}^{(k)} \|$



Apply Gram-Schmidt to $S = \{1, t, t^2, t^3, \dots\}$

$$\langle \vec{x}, \vec{y} \rangle = \int_{-1}^{1} x(t)\, y(t)\, dt$$

- $\vec{s}^{(0)} = 1$
  - $\vec{p}^{(0)} = \vec{s}^{(0)} = 1$
  - $\|\vec{p}^{(0)}\|^2 = 2$
  - $\vec{u}^{(0)} = \vec{p}^{(0)} / \|\vec{p}^{(0)}\| = \frac{1}{\sqrt{2}}$
- $\vec{s}^{(1)} = t$
  - $\langle \vec{u}^{(0)}, \vec{s}^{(1)} \rangle = \int_{-1}^{1} \frac{t}{\sqrt{2}} = 0$
  - $\vec{p}^{(1)} = \vec{s}^{(1)} = t$
  - $\|\vec{p}^{(1)}\|^2 = \frac{2}{3}$
  - $\vec{u}^{(1)} = \sqrt{3/2}\, t$
- $\vec{s}^{(2)} = t^2$
  - $\langle \vec{u}^{(0)}, \vec{s}^{(2)} \rangle = \int_{-1}^{1} \frac{t^2}{\sqrt{2}} dt = \frac{2}{3\sqrt{2}}$
  - $\langle \vec{u}^{(1)}, \vec{s}^{(2)} \rangle = \int_{-1}^{1} t^3/\sqrt{2} = 0$
  - $\vec{p}^{(2)} = \vec{s}^{(2)} - \frac{2}{3\sqrt{2}} \vec{u}^{(0)} = t^2 - \frac{1}{3}$
  - $\|\vec{p}^{(2)}\|^2 = 8/45$
  - $\vec{u}^{(2)} = \sqrt{\frac{5}{8}}(3t^2 - 1)$

Legendre Polynomials

The Gram-Schmidt algorithm leads to an orthonormal basis for $P_n([-1,1])$

$$\vec{u}^{(0)} = \sqrt{\frac{1}{2}}, \quad \vec{u}^{(1)} = \sqrt{\frac{3}{2}}\, t, \quad \vec{u}^{(2)} = \sqrt{\frac{5}{8}}(3t^2 - 1), \quad \vec{u}^{(3)} = \dots$$

Orthogonal projection over $P_3[-1,1]$

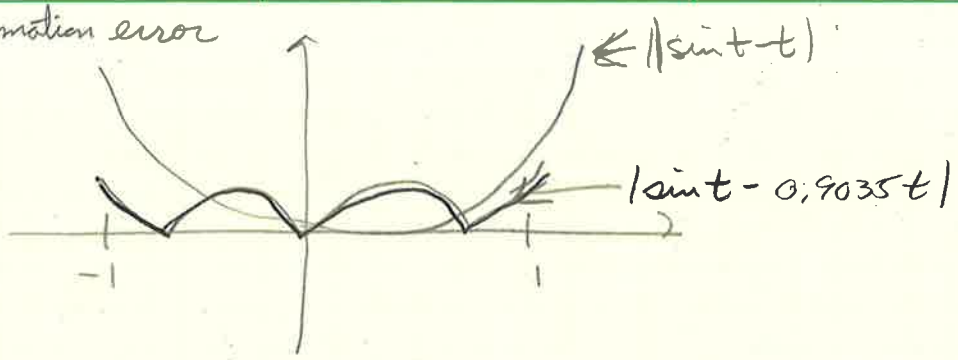
$$\alpha_k = \langle \vec{u}^{(k)}, \vec{x} \rangle = \int_{-1}^{1} u_k(t) \sin t\, dt$$

- $\alpha_0 = \langle \frac{1}{\sqrt{2}}, \sin t \rangle = 0$
- $\alpha_1 = \langle \sqrt{\frac{3}{2}}\, t, \sin t \rangle \approx 0.7377$
- $\alpha_2 = \langle \sqrt{\frac{5}{8}}(3t^2 - 1), \sin t \rangle = 0$

$\sin t \to \alpha_1 \vec{u}^{(1)} \approx 0.9035\, t$

Taylor Series: $\sin t \approx t$

Approximation error



Error norm:

Orthogonal projection over P_3 $[-1,1]$:

$\|\sin t - \alpha_1 \vec{u}^{(1)}\| \approx 0{,}0337$

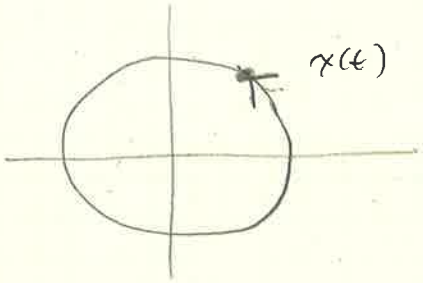
Taylor series: $\|\sin t - t\| \approx 0{,}0857$

3.1.a The frequency domain

- Oscillations are everywhere
  - Sustainable dynamic systems exhibit oscillatory behavior
  - Intuitively : things that don't move in circles don't last :
    - bombs
    - rockets
    - human beings...

Period P

Frequency $f = \frac{1}{P}$



- The intuition
  - humans analyze complex signals (audio, images) in terms of their sinusoidal components
  - We can build instruments that "resonate" at one or multiple frequencies (tuning fork vs. piano)
  - the "frequency domain" seems to be as important as the time domain
- Fundamental question : can we decompose any signal into sinusoidal elements? Yes, using Fourier analysis

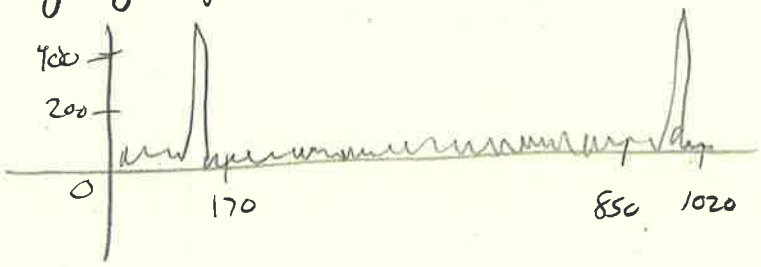
Analysis	Synthesis
- from time domain to frequency domain	- from frequency domain to time domain
- find the contribution of different frequencies	- create signals with known frequency content
- discover "hidden" signal properties	- fit signals to specific frequency regions

3.1.b The DFT as a change of basis

- The mathematical setup
  - let's start with finite-length signals (i.e. vectors in $\mathbb{C}^N$)
  - Fourier analysis is a simple change of basis
  - a change of basis is a change of perspective

- Mystery signal in time domain

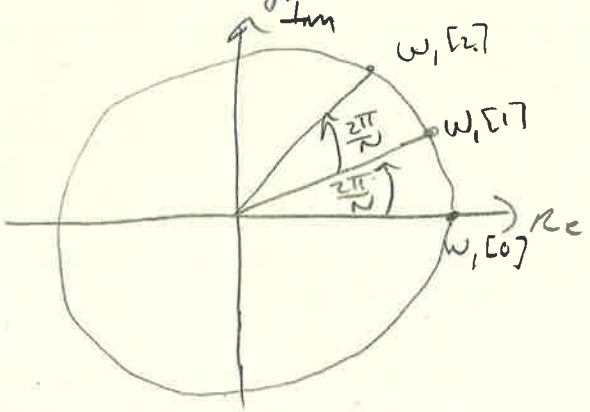
- Mystery signal in the Fourier basis



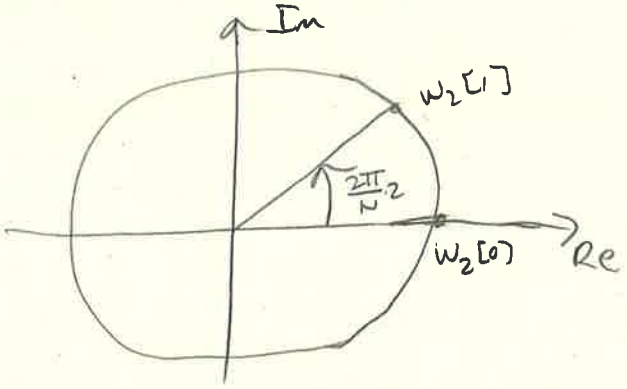
- The Fourier Basis for $\mathbb{C}^N$

Claim: the set of N signals in $\mathbb{C}^N$
$w_k[n] = e^{j\frac{2\pi}{N}nk}$, $n,k = 0, 1, \ldots, N-1$ is an orthogonal basis in $\mathbb{C}^N$. $\omega = \frac{2\pi}{N}k$

In vector notation: $\{\vec{w}^{(k)}\}_{k=0,1,\ldots,N-1}$ with $w_n^{(k)} = e^{j\frac{2\pi}{N}nk}$ is an orthogonal basis in $\mathbb{C}^N$.

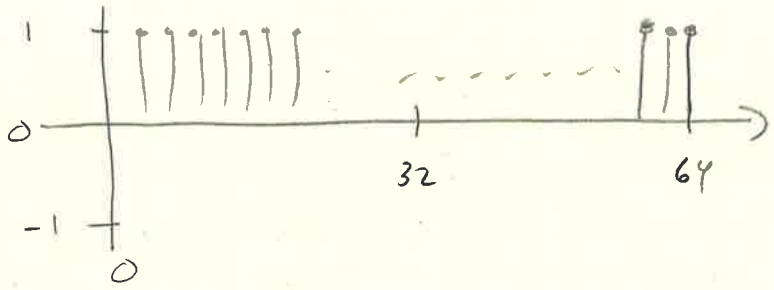


$w_n^{(1)} = e^{j\frac{2\pi}{N}k}$

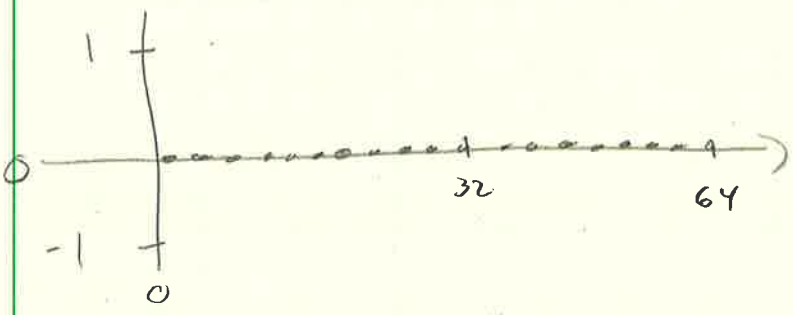


$w_n^{(2)} = e^{j\frac{2\pi}{N}2n}$

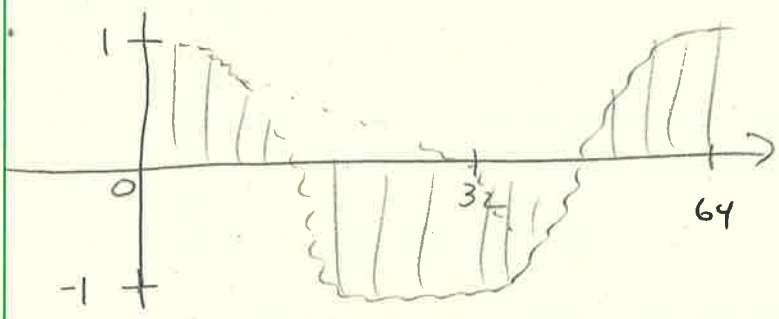
Basis vector $\vec{w}^{(0)} \in \mathbb{C}^{64}$



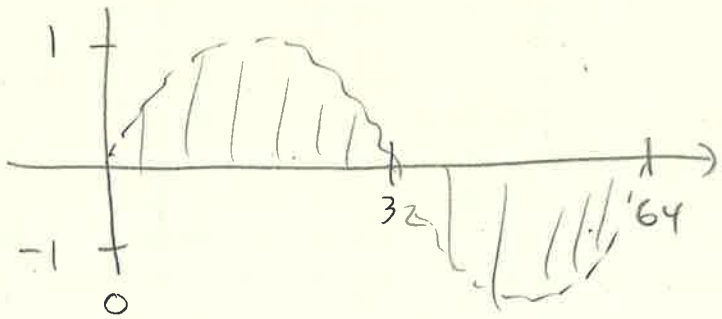
$\vec{w}_n^{(0)} = e^{j\frac{2\pi}{N}0n} = 1$



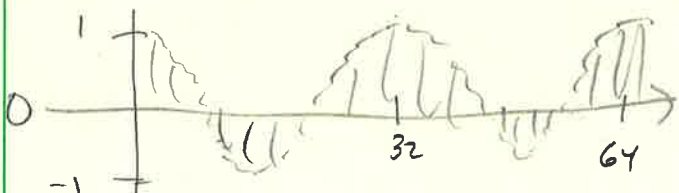
Basis vector $\vec{w}^{(1)} \in \mathbb{C}^{64}$, $\vec{w}_n^{(1)} = e^{j\frac{2\pi}{N}1n}$



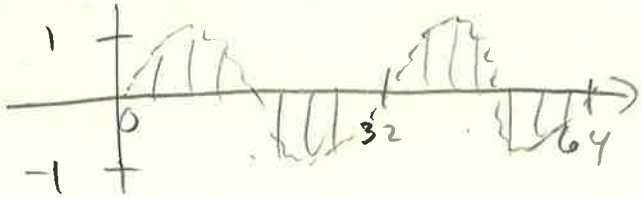
$\omega = \frac{2\pi}{N}$



Basis vector $\vec{w}^{(2)} \in \mathbb{C}^{64}$



$\omega = \frac{2\pi}{N} \cdot 2$



$\vec{w}^{(3)} \in \mathbb{C}^{64}$; $\omega = \frac{2\pi}{N} 3 = \frac{2\pi}{64} \cdot 3$

$\vdots$

$\vec{w}^{(16)} \in \mathbb{C}^{64}$; $\omega = \frac{2\pi}{64} 16 = \frac{\pi}{2}$

$\vdots$

$\vec{w}^{(32)} \in \mathbb{C}^{64}$; $\omega = \frac{2\pi}{64} 32 = \pi$

$\vdots$

$\vec{w}^{(62)} \in \mathbb{C}^{64}$ has same real part as $\vec{w}^{(2)}$ but the imaginary part is inverted

$\vdots$

$Re(\vec{w}^{(63)}) = Re(\vec{w}^{(1)})$ but imaginary part are inverted

- Proof of orthogonality

$$\langle \vec{w}^{(k)}, \vec{w}^{(n)} \rangle = \sum_{n=0}^{N-1} \left(e^{j\frac{2\pi}{N}nk}\right)^* e^{j\frac{2\pi}{N}nh}$$

$$= \sum_{n=0}^{N-1} e^{j\frac{2\pi}{N}(h-k)n}$$

$$\left(\sum_{n=0}^{N-1} a^n = \frac{1-a^N}{1-a}\right) = \begin{cases} N, & h=k \\ \frac{1-e^{j2\pi(h-k)}}{1-e^{j\frac{2\pi}{N}(h-k)}} = 0, & \text{otherwise} \end{cases}$$

$h-k \in \mathbb{N} \Rightarrow e^{j2\pi(n-k)} = 1$

- Remarks
  - N orthogonal vectors → basis for $\mathbb{C}^N$
  - vectors are not orthonormal. Normalization factor would be $1/\sqrt{N}$

3.2 The Discrete Fourier Transform (DFT)

3.2a DFT definition

- Basis expansion
  - Analysis formula: $X_k = \langle \vec{w}^{(k)}, \vec{x} \rangle$
  - Synthesis formula: $\vec{x} = \frac{1}{N} \sum_{k=0}^{N-1} X_k \vec{w}^{(k)}$

- Change of bases in matrix form

  - Define $W_N = e^{-j\frac{2\pi}{N}}$ (or simply W when N is evident)

  Change of basis matrix $\underline{W}$ with $\underline{W}[n,m] = W_N^{nm}$

$$\underline{W} = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & W^1 & W^2 & W^3 & \cdots & W^{N-1} \\ 1 & W^2 & W^4 & W^6 & \cdots & W^{2(N-1)} \\ \vdots & & & & & \vdots \\ 1 & W^{N-1} & W^{2(N-1)} & W^{3(N-1)} & \cdots & W^{(N-1)^2} \end{bmatrix}$$

Analysis formula: $\underline{X} = \underline{W}\,\vec{x}$

Synthesis formula: $\vec{x} = \frac{1}{N}\underline{W}^H \underline{X}$

- Basis expansion (signal notation)

Analysis formula:

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}nk}, \quad k = 0, 1, \ldots, N-1$$

N-point signal in the frequency domain

Synthesis formula:

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]\, e^{j\frac{2\pi}{N}nk}, \quad n = 0, 1, \ldots, N-1$$

N-point signal in the "time" domain

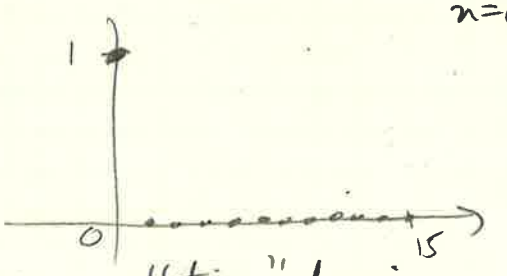
3.2b Examples of DFT calculation

- DFT is obviously linear

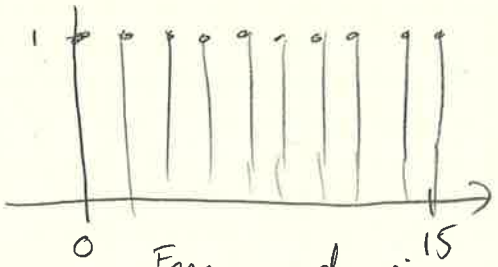
$$\text{DFT}\{\alpha x[n] + \beta y[n]\} = \alpha\,\text{DFT}\{x[n]\} + \beta\,\text{DFT}\{y[n]\}$$

- DFT of $x[n] = \delta[n]$, $x[n] \in \mathbb{C}^N$

$$X[k] = \sum_{n=0}^{N-1} \delta[n]\, e^{-j\frac{2\pi}{N}nk} = 1$$



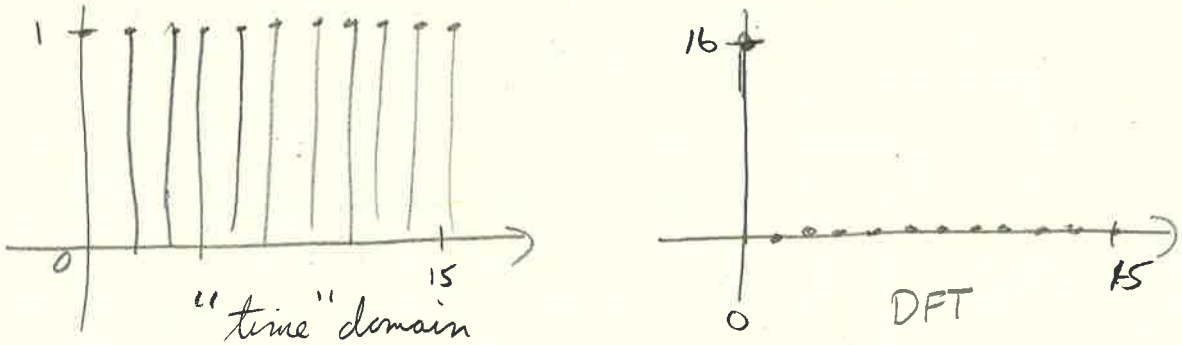
"time" domain



Frequency domain

DFT of $x[n] = 1$, $x[n] \in \mathbb{C}^N$

$$X[k] = \sum_{n=0}^{N-1} e^{-j\frac{2\pi}{N}nk} = N\delta[k]$$



DFT of $x[n] = 3\cos(2\pi/16 n)$, $x[n] \in \mathbb{C}^{64}$

$$x[n] = 3\cos\left(\frac{2\pi}{16}n\right) = 3\cos\left(\frac{2\pi}{64}4n\right) \qquad \omega = \frac{2\pi}{64}$$

$$= \frac{3}{2}\left[e^{j\frac{2\pi}{64}4n} + e^{-j\frac{2\pi}{64}4n}\right]$$

$$= \frac{3}{2}\left[e^{j\frac{2\pi}{64}4n} + e^{j\frac{2\pi}{64}60n}\right] \qquad -j\tfrac{2\pi}{64}4n = j\tfrac{2\pi}{64}60n$$

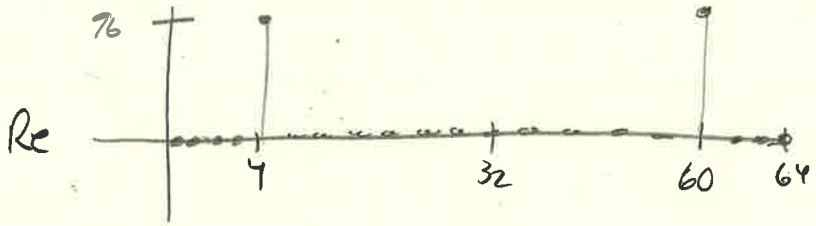
$$= \frac{3}{2}\left[W_4[n] + W_{60}[n]\right]$$

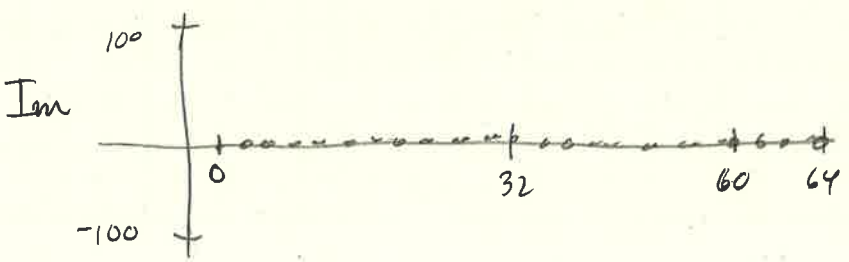
$$X[k] = \langle W_k[n], x[n]\rangle$$

$$= \langle W_k[n], \tfrac{3}{2}(W_4[n] + W_{60}[n])\rangle$$

$$= \frac{3}{2}\langle W_k[n], W_4[n]\rangle + \frac{3}{2}\langle W_k[n], W_{60}[n]\rangle$$

$$= \begin{cases} \frac{3}{2}\cdot 64 = 96, & k = 4, 60 \\ 0, & \text{otherwise} \end{cases}$$





- DFT of $x[n] = 3\cos\left(\frac{2\pi}{16}n + \pi/3\right)$, $x[n] \in \mathbb{C}^{64}$

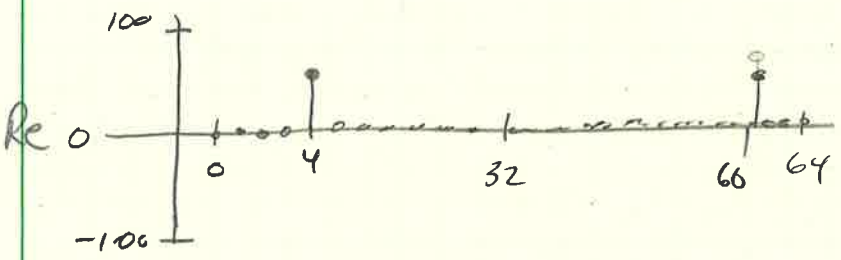
$$x[n] = 3\cos\left(\frac{2\pi}{16}n + \frac{\pi}{3}\right)$$

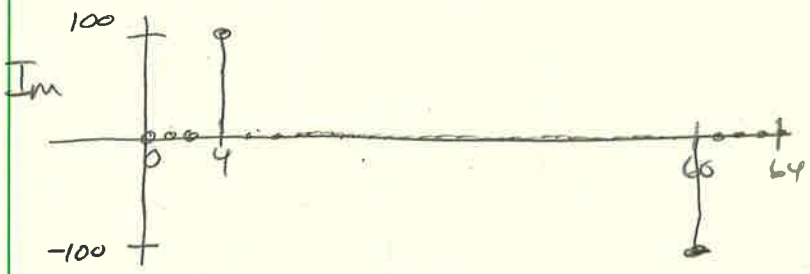
$$= 3\cos\left(\frac{2\pi}{64}4n + \frac{\pi}{3}\right)$$

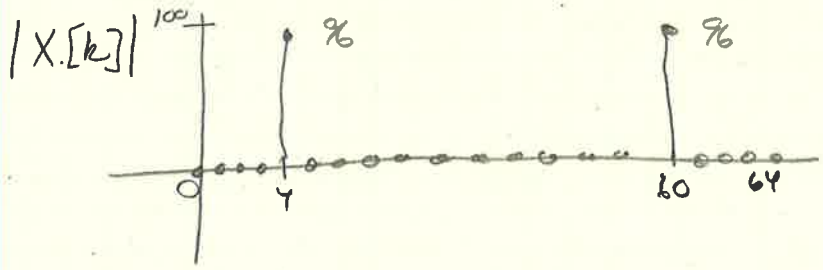
$$= \frac{3}{2}\left[e^{j\frac{2\pi}{64}4n}\, e^{j\frac{\pi}{3}} + e^{-j\frac{2\pi}{64}4n}\, e^{-j\frac{\pi}{3}}\right]$$

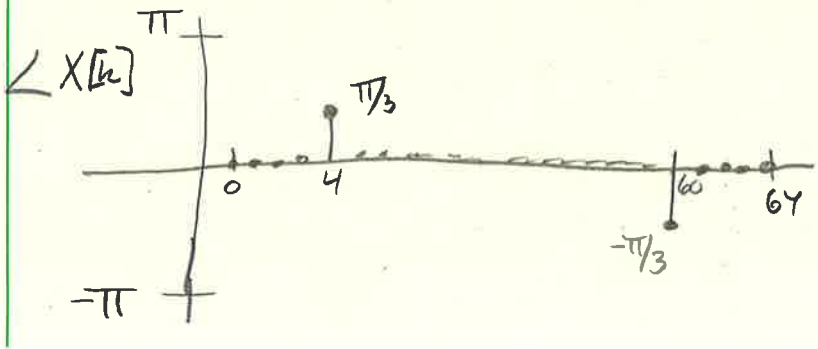
$$= \frac{3}{2}\left[e^{j\frac{\pi}{3}} W_4[n] + e^{-j\frac{\pi}{3}} W_{60}[n]\right]$$

$$X[k] = \langle w_k[n], x[n] \rangle = \begin{cases} 96\, e^{j\frac{\pi}{3}}, & k=4 \\ 96\, e^{-j\frac{\pi}{3}}, & k=6 \\ 0, & \text{otherwise} \end{cases}$$



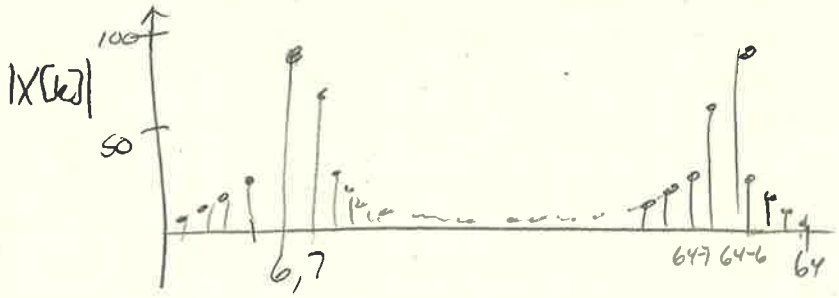


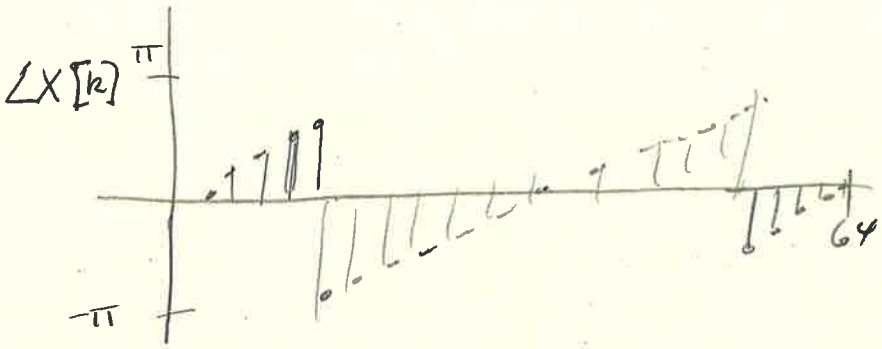




- DFT of $x[n] = 3\cos\left(\frac{2\pi}{10}n\right)$, $x[n] \in \mathbb{C}^{64}$

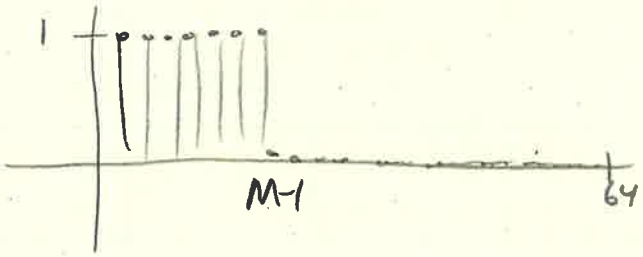
$$\frac{2\pi}{64}6 < \frac{2\pi}{10} < \frac{2\pi}{64}7$$





- DFT of length-M step in $\mathbb{C}^N$

$$x[n] = \sum_{h=0}^{M-1} \delta[n-h], \quad n = 0, 1, \ldots, N-1$$



$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}nk} = \sum_{n=0}^{M-1} e^{-j\frac{2\pi}{N}nk}$$

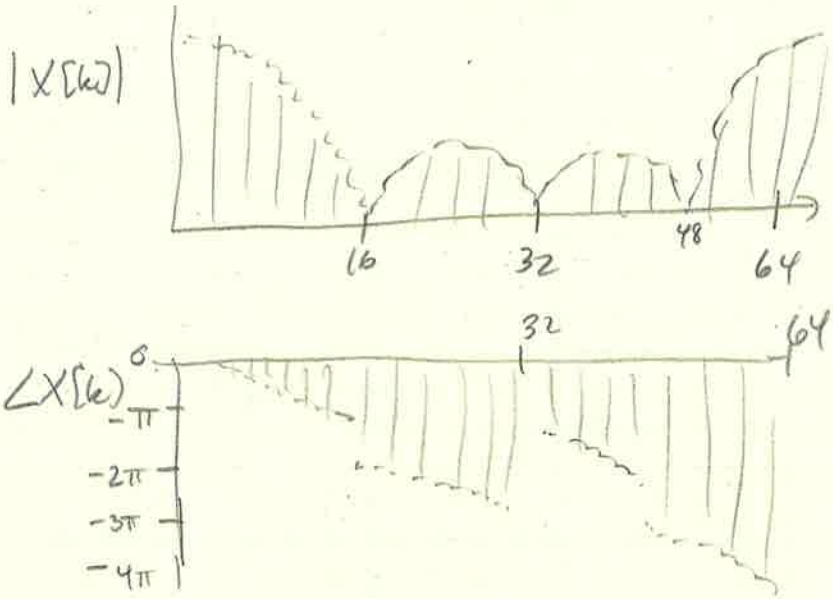
$$= \frac{1 - e^{-j\frac{2\pi}{N}kM}}{1 - e^{-j\frac{2\pi}{N}k}} \qquad \left(1 - e^{-j\alpha} = e^{-j\frac{\alpha}{2}}\left(\underbrace{e^{j\frac{\alpha}{2}} - e^{-j\frac{\alpha}{2}}}_{2j\sin\frac{\alpha}{2}}\right)\right)$$

$$= \frac{e^{-j\frac{\pi}{N}kM}\left[e^{j\frac{\pi}{N}kM} - e^{-j\frac{\pi}{N}kM}\right]}{e^{-j\frac{\pi}{N}k}\left[e^{j\frac{\pi}{N}k} - e^{-j\frac{\pi}{N}k}\right]}$$

$$= \frac{\sin\left(\frac{\pi}{N}Mk\right)}{\sin\left(\frac{\pi}{N}k\right)} e^{-j\frac{\pi}{N}(M-1)k}$$

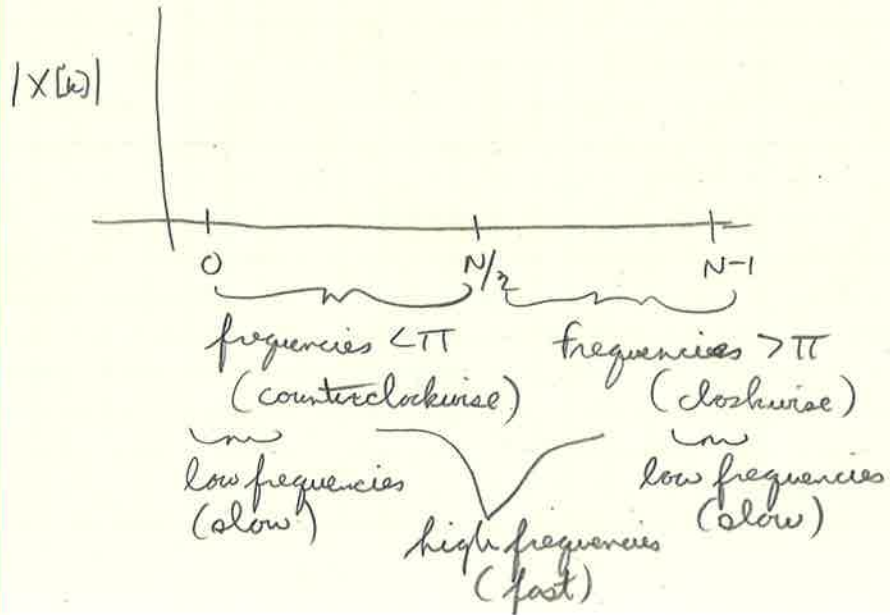
- $X[0] = M$, from the definition of the sum
- $X[k] = 0$, if Mk/N is an integer $(0 \le k < N)$
- $\angle X[k]$ is linear in k (except at sign changes for the real part)

— DFT of length-4 step in $\mathbb{C}^{64}$

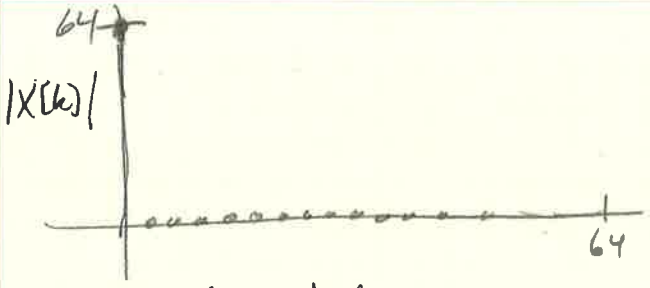


- Wrapping the phase
  - Often the phase is displayed "wrapped" over the $[-\pi, \pi)$ interval
    - most numerical packages return wrapped phase
    - phase can be unwrapped by adding multiples of 2π.

3.2c Interpreting a DFT plot

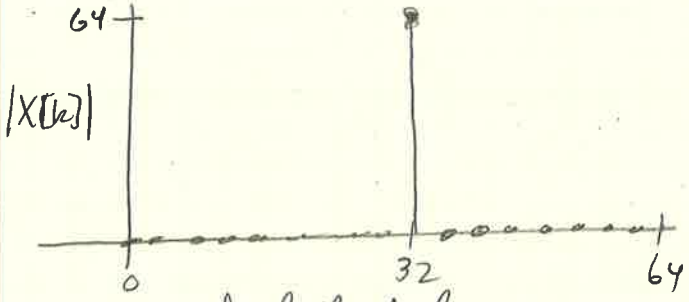


$x[n] = 1$ (slowest signal)



only lowest frequency

$x[n] = \cos \pi n = (-1)^n$ (fastest signal)



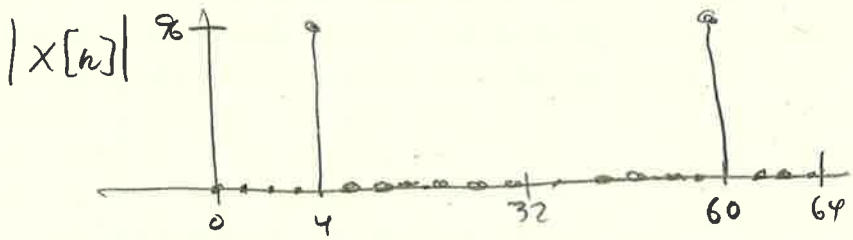
only highest frequency

- Energy distribution
  - Parseval : $\|\vec{x}\|^2 = \sum |\alpha_k|^2$

$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2$$

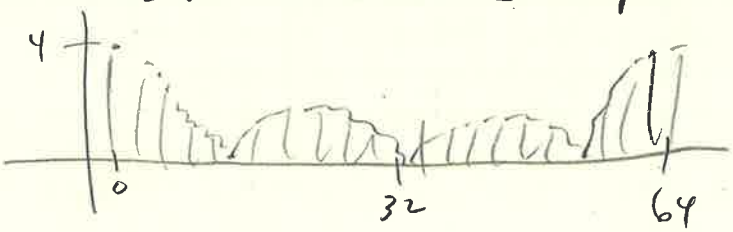
  - square magnitude of k-th DFT coefficient proportional to signal's energy at frequency $\omega = \frac{2\pi}{N} k$

$x[n] = 3\cos\left(\frac{2\pi}{16} n\right)$ (sinusoid)



energy concentrated on single frequency (counterclockwise and clockwise combine to give real signal)

$x[n] = u[n] - u[n-4]$ (step)

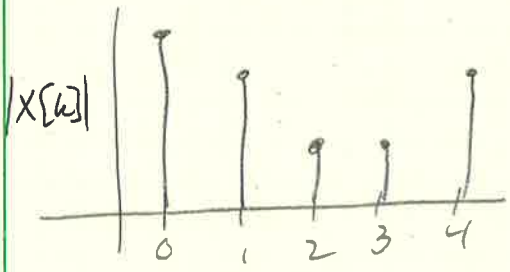


energy mostly in low frequencies

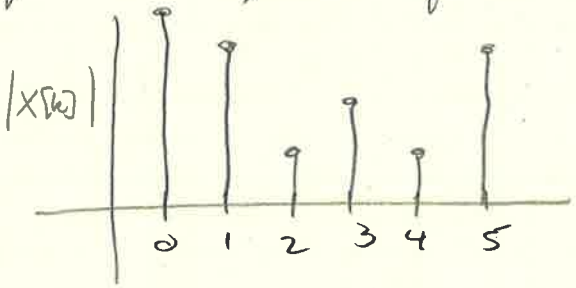
- DFT of real signals

• For real signals the DFT is "symmetric" in magnitude:

$$|X[k]| = |X[N-k]| \quad \text{for } k = 1, 2, \ldots, \lfloor N/2 \rfloor \text{ (floor)}$$

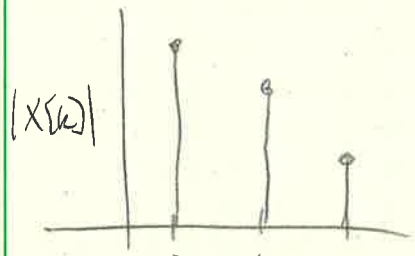


N=5, odd length

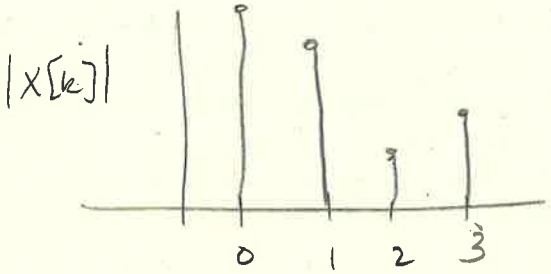


N=6, even length

• For real signals, magnitude plots need only $\lfloor N/2 \rfloor + 1$ points



N=5, odd length



N=6, even length

3.3 : The DFT in practice

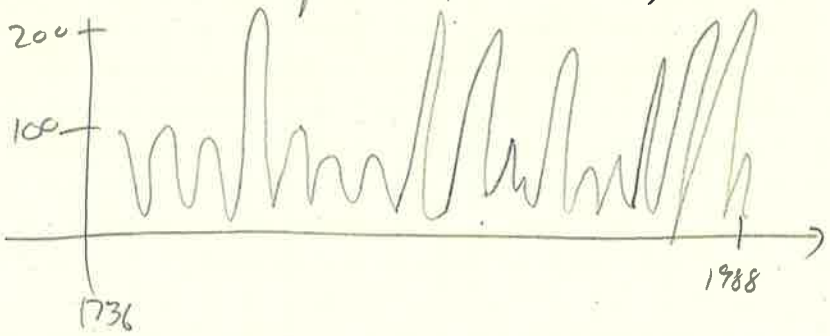
3.3a DFT analysis

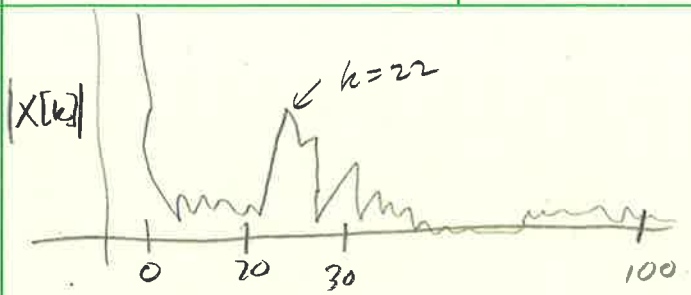
- Mystery signal revisited

$$x[n] = \cos(\omega n + \phi) + \eta[n] \text{ with}$$

$$\phi = 0, \quad \omega = \frac{2\pi}{1024} 64 \quad \text{↑ peak at } k = 64$$

- Solar spots
  - sunspot number : $S = 10 \times$ # of clusters + # of spots
  - data set from 1749 to 2003, 2904 months

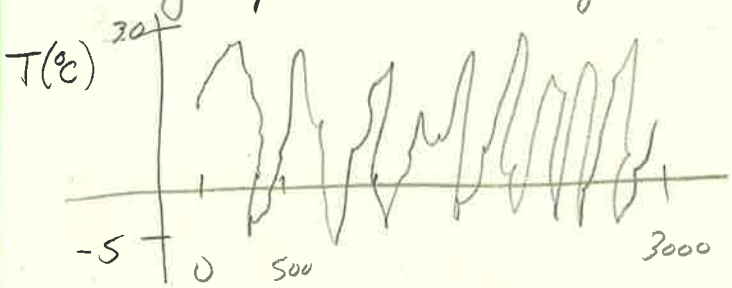


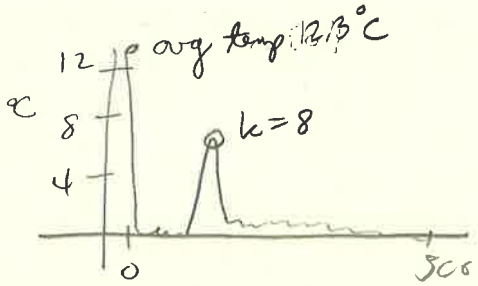


DFT of solar spots signal

- DFT main peak for $k=22$
- 22 cycles over 2904 months
- period: $\frac{2904}{22} \approx 11$ years

- Daily temperature (2920 days)



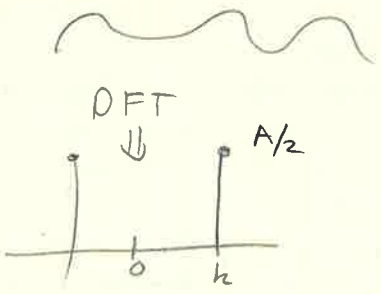


first few hundred DFT coefficients (in magnitude and normalized by the length of the temperature vector)

- average value (0-th DFT coefficient normalized): 12.3°C
- DFT main peak for $k=8$, value 6.4°C
- 8 cycles over 2920 days
- period: $\frac{2920}{8} = 365$ days
- temperature excursion: $12.3°C \pm 12.8°C$

$$X[0] = \sum_{n=0}^{N-1} x(n)$$

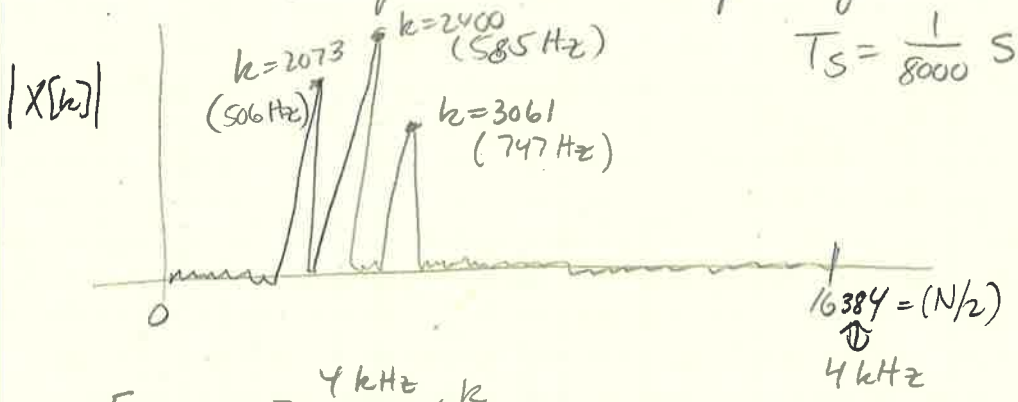
$A = \cos(\omega n)$



- Labeling the frequency axis
  - If we know the "clock" of the system T_s
    - fastest (positive) frequency is $\omega = \pi$
    - sinusoid at $\omega = \pi$ needs two samples to do a full revolution
    - time between samples: $T_s = \frac{1}{F_s}$ seconds
  - real-world period for fastest sinusoid: $2T_s$ seconds
  - real-world frequency for fastest sinusoid: $F_s/2$ Hz

- Example: train whistle

32768 samples (the "clock" of the system $F_S = 8000$ Hz)



$T_S = \frac{1}{8000}$ s

$F_S/2 \Leftrightarrow k = N/2$

$\text{Frequency} = \frac{4\text{ kHz}}{16384} \cdot k$

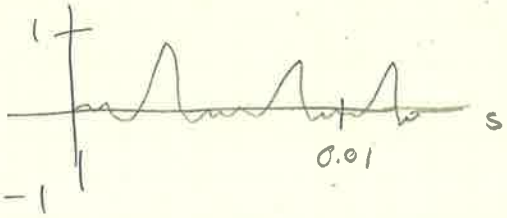
If we look up the frequencies:

B minor chord

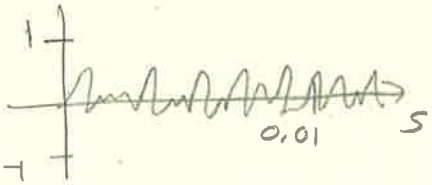
3.36 DFT example - analysis of musical instruments

- Analysis of musical instruments
  - "We all know about pitch...
    - It is really about frequency, or cycles per seconds
    - How about harmonics?
    - That is what gives the timbre of an instrument!
  - A difficult temporal analysis of musical instruments

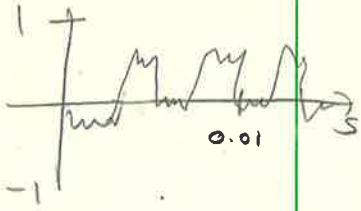
Sax



violin

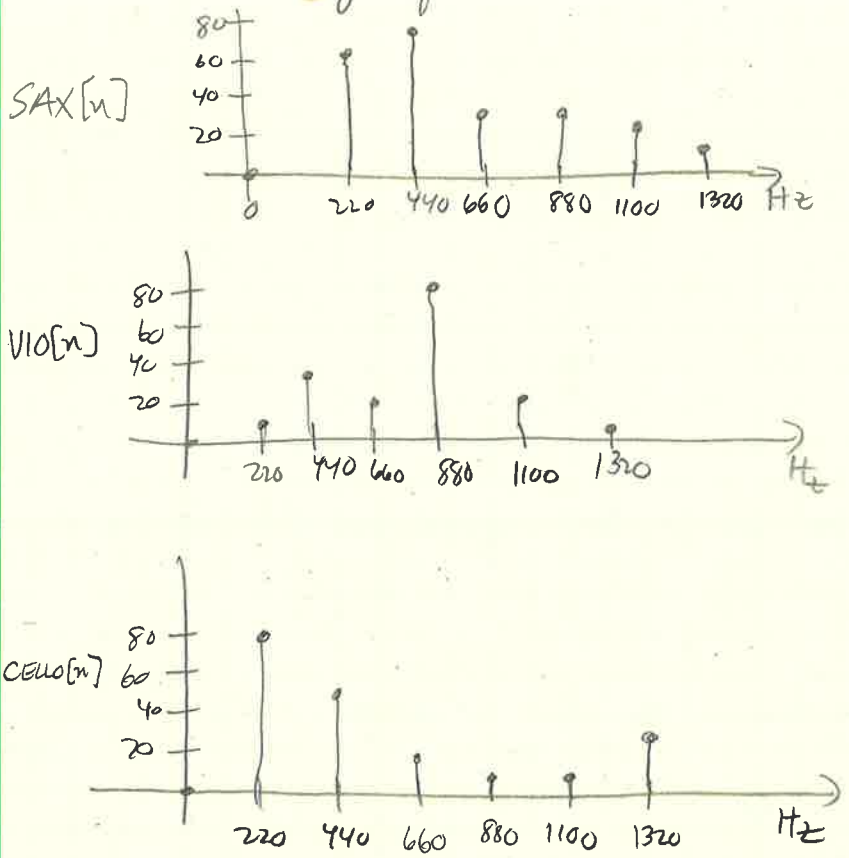


cello



- What is the note?
- Could you guess the instrument from the temporal plot?
- In the time domain it is hard to process information of the sound

- A Fourier analysis of musical instruments

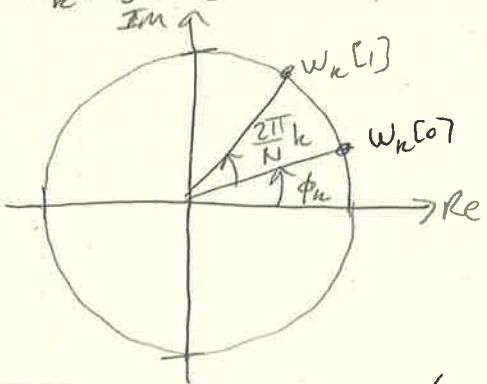


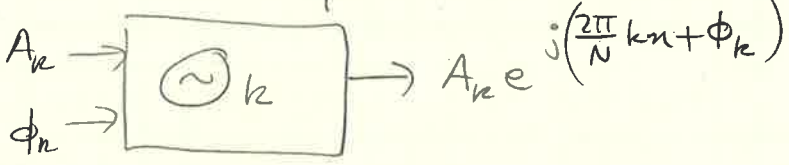
- The played note is the frequency of the first peak : 220 Hz in this case
- Other peaks are called harmonics : they defines the typical sound of the instrument
- Without Fourier, we would have been lost

3.3c DFT synthesis

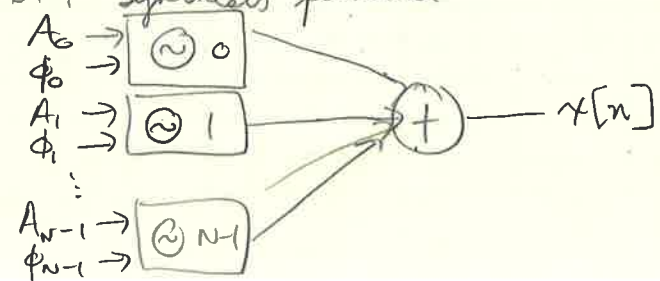
- Synthesis : the sinusoidal generator

$$W_k[n] = e^{j\left(\frac{2\pi}{N}kn + \phi_k\right)}$$





- DFT Synthesis formula

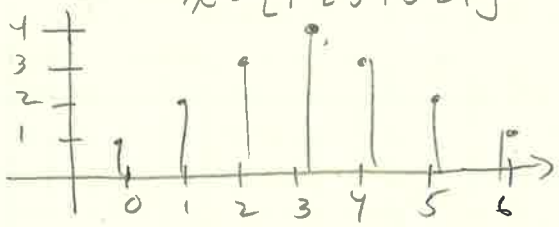


- Initializing the machine

$$A_k = |X[k]|/N$$

$$\phi_k = \angle X[k]$$

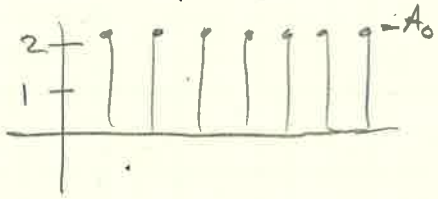
$$\vec{x} = [1\ 2\ 3\ 4\ 3\ 2\ 1]^T$$

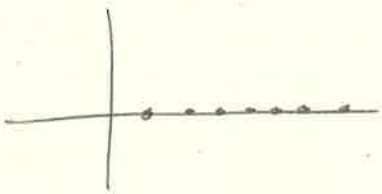


k	A_k	ϕ_k
0	2.2857	0
1	0.7213	-2.6928
2	0.0440	0.8976
3	0.0919	-1.7952
4	0.0919	-1.7952
5	0.0440	-0.8976
6	0.7213	2.6928

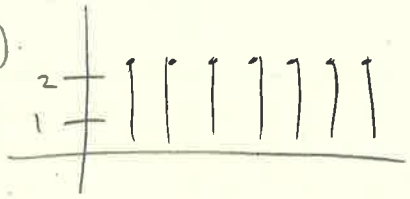
$k=0$

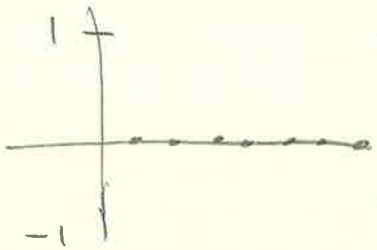
$$A_k e^{j\left(\frac{2\pi}{N}kn+\phi_k\right)}$$





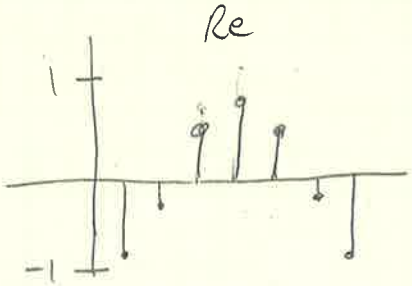
$$\sum A_k e^{j\left(\frac{2\pi}{N}kn+\phi_k\right)}$$

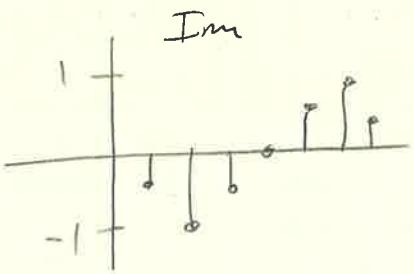




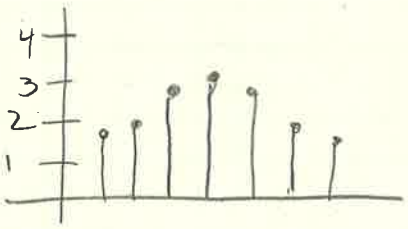
$k=1$

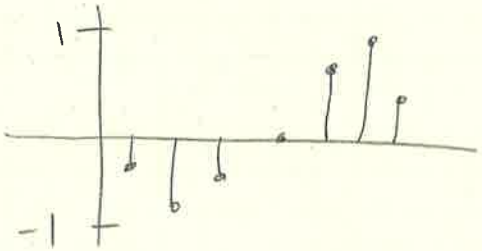
$$A_k e^{j\left(\frac{2\pi}{N}kn+\phi_k\right)}$$





$$\sum A_k e^{j\left(\frac{2\pi}{N}kn+\phi_k\right)}$$





at $k=6$, we have reconstructed the signal

- Running the machine too long...

$x[n+N] = x[n]$ – Output signal is N-periodic!

- Inherent periodicities in the DFT

the synthesis formula:

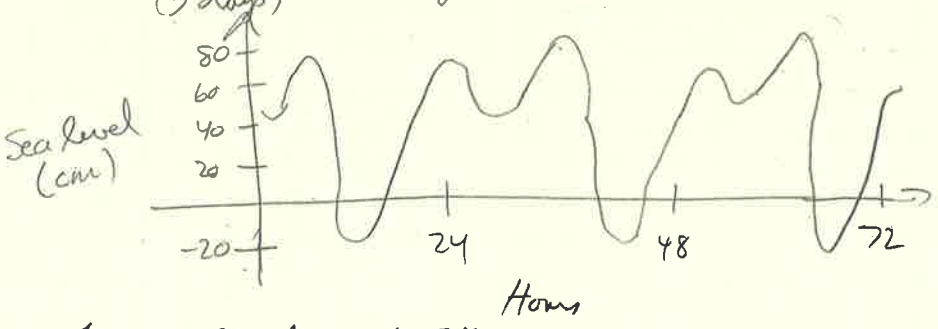
$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}nk}$, $n \in \mathbb{Z}$, produces an N-periodic signal in the time domain

the analysis formula:

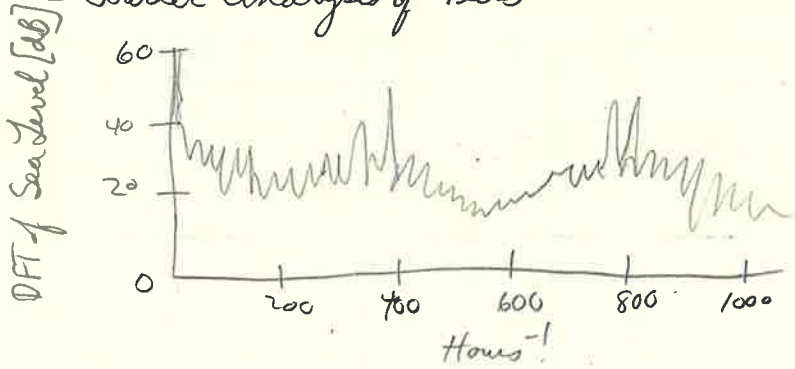
$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}nk}$, $k \in \mathbb{Z}$, produces N-periodic signal in the frequency domain.

3.3d DFT example – tide prediction in Venice

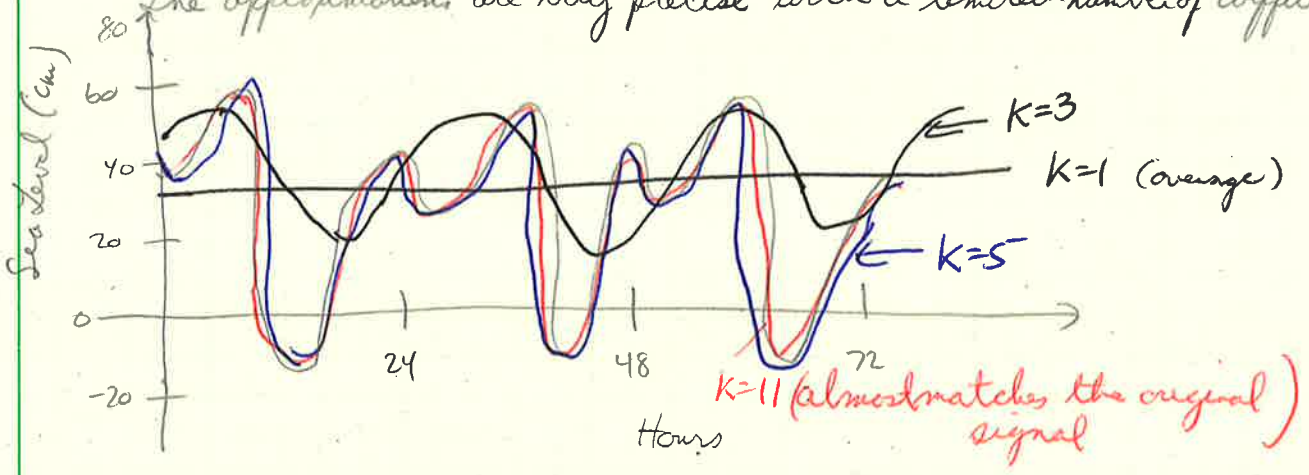
- Tides are due mostly to periodic astronomical phenomena
- Can we predict tides using Fourier? The first step is to approximate them
- We consider hourly measurements taken in Canal Grande during 2011 (3 days)



Fourier Analysis of Tides



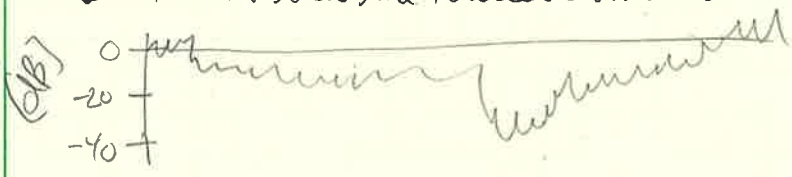
- Let us approximate tides using few Fourier coefficients
  - We consider on $K = 1, 3, 5, 11$ Fourier coefficients
  - Darker gray represents approximation with more coefficients
  - The approximations are very precise with a limited number of coefficients



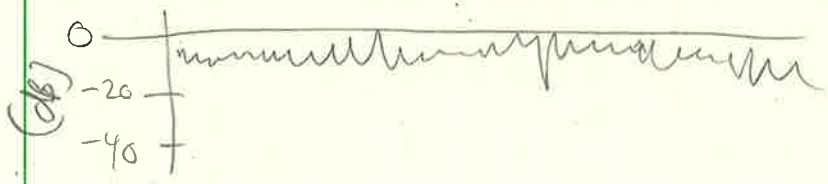
3.3e DFT example - MP3 compression

- MP3 Compression trick
  - How can an MP3 song sound so good while being so compressed?
  - Compression introduces noise
  - The trick is to shape the noise!

The secret is in the Fourier domain!



Original Music

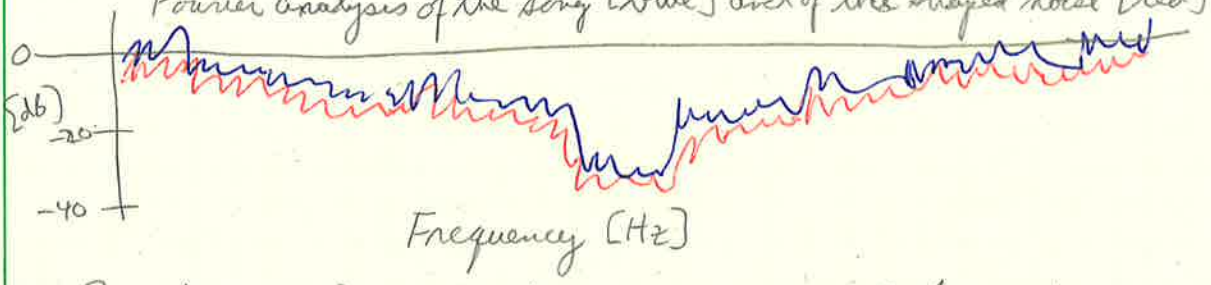


Badly noisy music (3 dB SNR)

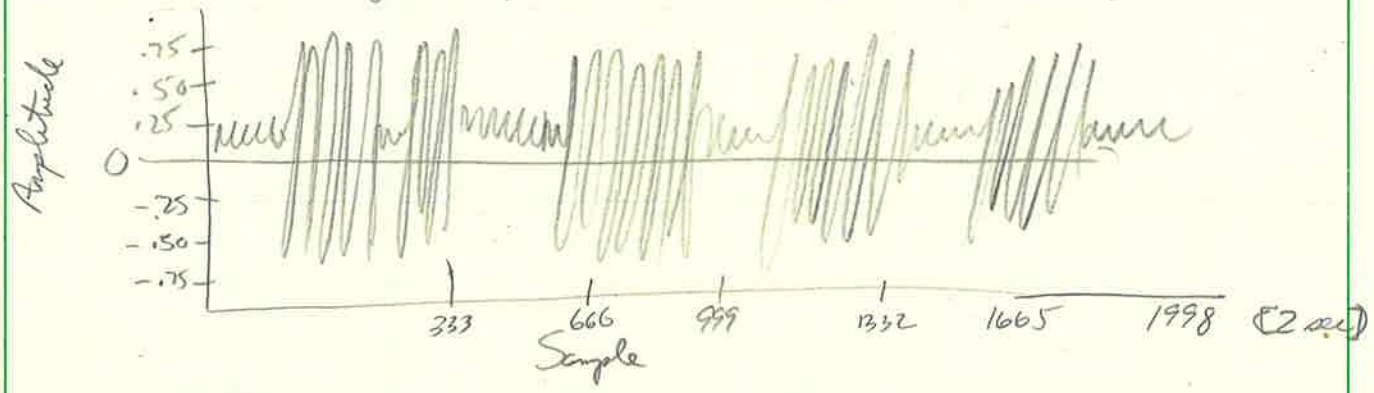
MP3 ((13 dB SNR)
(resembles the original

- Conclusions
  - MP3: complex compression algorithm that introduces errors
  - Errors shaped as the song in the Fourier domain → higher perceived quality
  - MP3 minimizes the perceived quality decay by shaping the compression errors

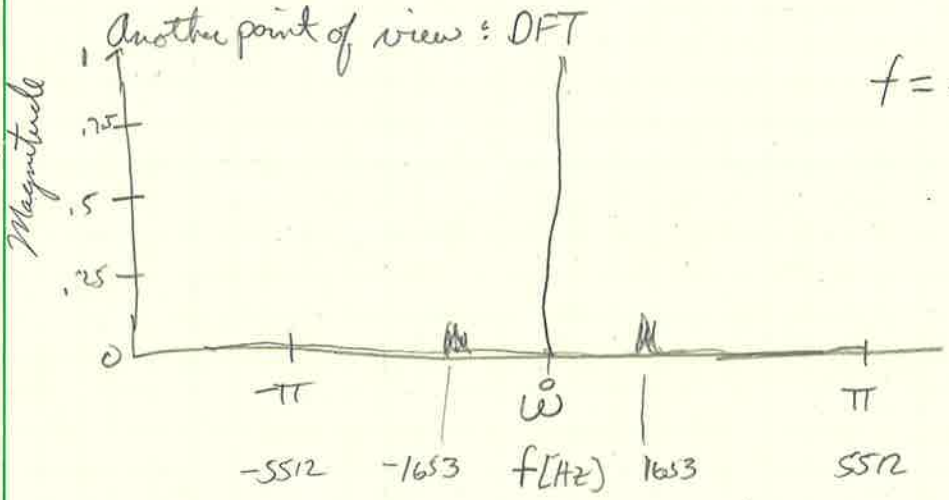
Fourier analysis of the song [blue] and of the shaped noise [red]



Signal of the Day: The first man-made signal from outer space (Sputnik)

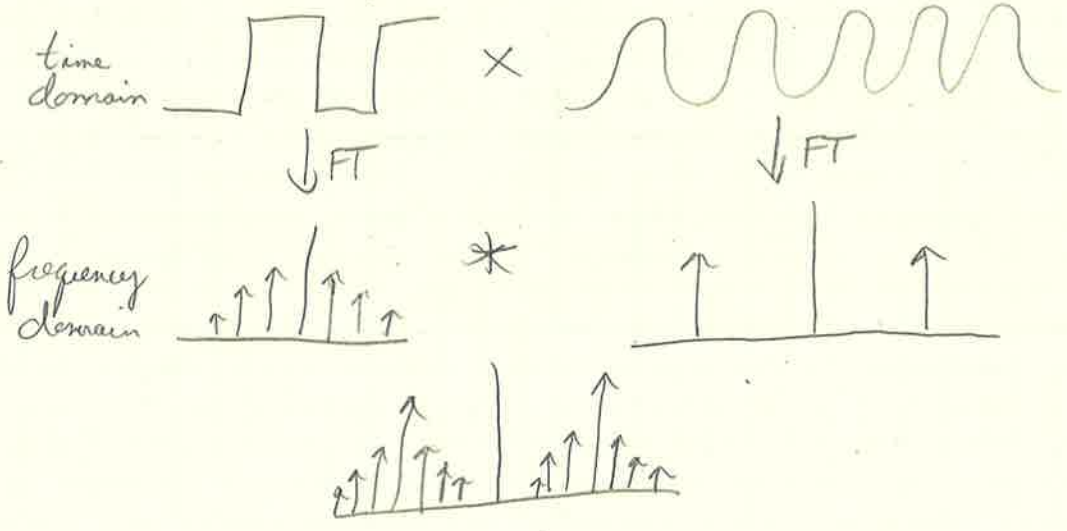


Another point of view: DFT



$f = \frac{\omega f_s}{2\pi}$ ← frequency at which we measure the signal

- Let's understand this...



Summary of Lesson 3.3

The DFT can be used as an analysis tool to understand the frequency components that a signal contains. If a signal has an associated system clock T_S (or a frequency $F_S = 1/T_S$), we can map the index k of the DFT coefficients to real frequencies. The largest digital frequency $N/2$ is associated with the largest continuous-time frequency $F_S/2$. Thus, the continuous frequency corresponding to index k is given by $\frac{kF_S}{N}$ and is measured in Hz.

The DFT synthesis can be seen as a series of up to N coupled sinusoidal generators:

- sinusoidal generator k has frequency $\frac{2\pi k}{N}$
- the amplitude of sinusoidal generator k is given by the magnitude of the DFT coefficient $|X[k]|$
- the phase of sinusoidal generator k is given by the phase of the DFT coefficient $\angle X[k]$

If we let the DFT synthesis run beyond $N-1$, we obtain an N-periodic signal, $x[n+N] = x[n]$. Likewise, the analysis formula produces also an N-periodic series of Fourier coefficients. This side comment will be very important when we study another form of Fourier transform for periodic sequences, namely discrete Fourier Series (DFS).

3.4 The Short-Time Fourier Transform (STFT)

3.4a The STFT

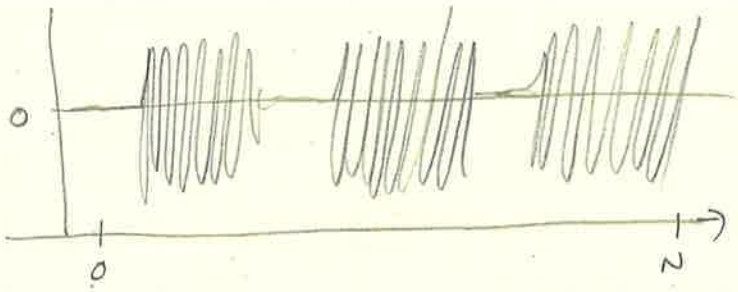
– Dual-Tone Multi Frequency dialing (DTMF)

	1209 Hz	1336 Hz	1477 Hz
697 Hz	1	2	3
770 Hz	4	5	6
852 Hz	7	8	9
941 Hz	*	0	#

Analog ~~telephone~~

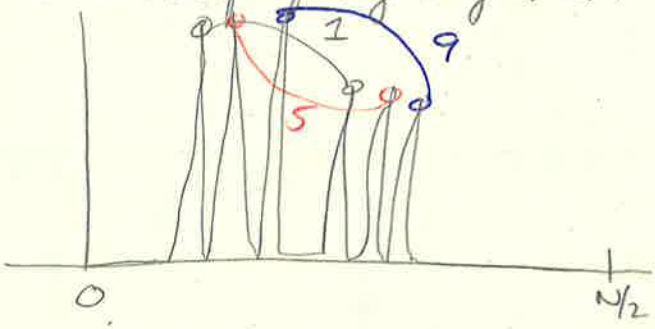
1) Frequencies are co-prime
2) No sum or difference of frequencies is in the set

1-5-9 in time



Can't tell the digit in time domain

1-5-9 in frequency (magnitude)



- The fundamental tradeoff
  - time representation obfuscates frequency
  - frequency representation obfuscates time

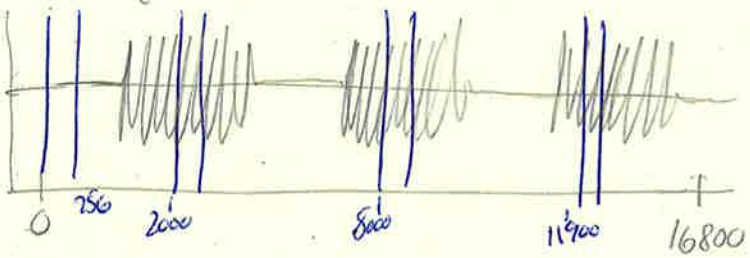
- Short-term Fourier Transform

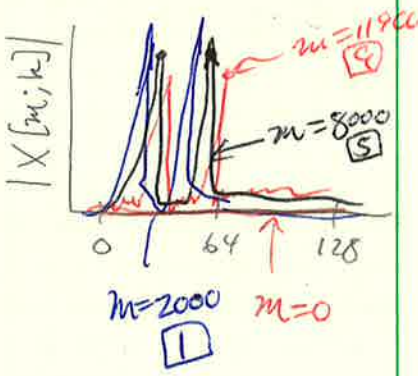
Idea:
- take small signal pieces of length L
- look at the DFT of each piece

$$X[m;k] = \sum_{n=0}^{L-1} x[m+n]\, e^{-j\frac{2\pi}{L}nk}$$

(m: starting point)

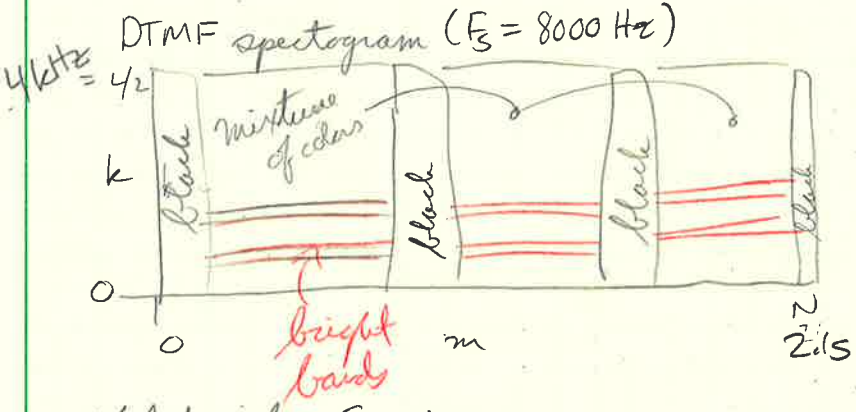
STFT (L=256)





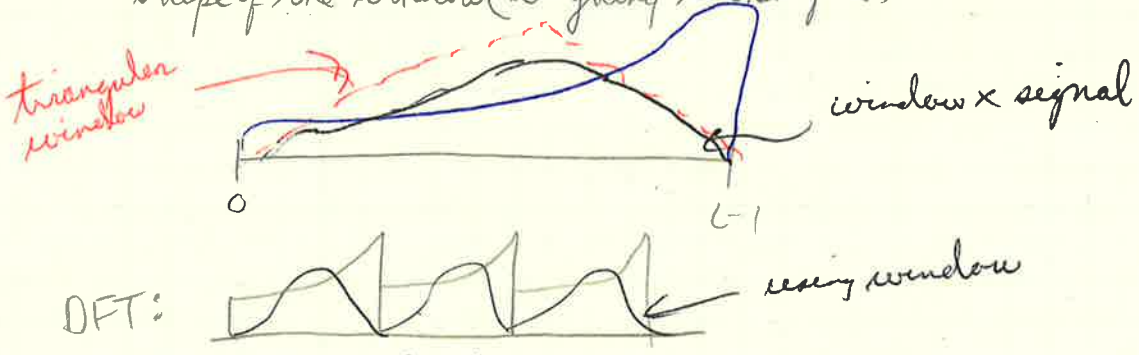
3.4b The spectogram

Idea:
- Color-code the magnitude: dark is small, white is large
- use $10 \log_{10}(|X[m;k]|)$ to see better (power in dB)
- plot spectral slices one after another



- Labeling the Spectogram
  - If we know the "system clock" $F_S = 1/T_S$, we can label the axis
    - highest positive frequency: $F_S/2$ Hz
    - frequency resolution: F_S/L Hz
    - width of time slices: LT_S seconds

- The Spectogram Questions:
  - width of the analysis window?
  - position of the windows (overlapping?)
  - shape of the window (weighing the samples)



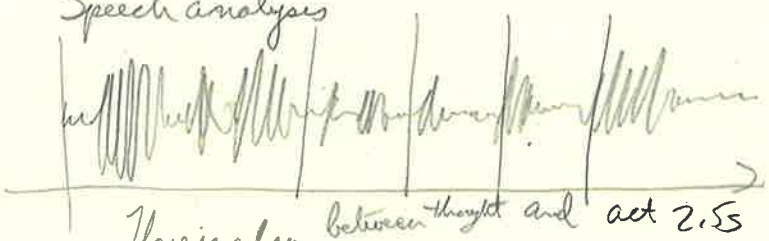
- Wideband vs. Narrowband

Long window: narrowband spectrogram
- long window ⇒ more DFT points ⇒ more frequency resolution $\frac{F_S}{L}$
- long window ⇒ more "things can happen" ⇒ less precision in time

Short window: wideband spectrogram
- short window ⇒ many time slices ⇒ precise location of transitions
- short window ⇒ fewer DFT points ⇒ poor frequency resolution

3.4c Time-frequency tiling

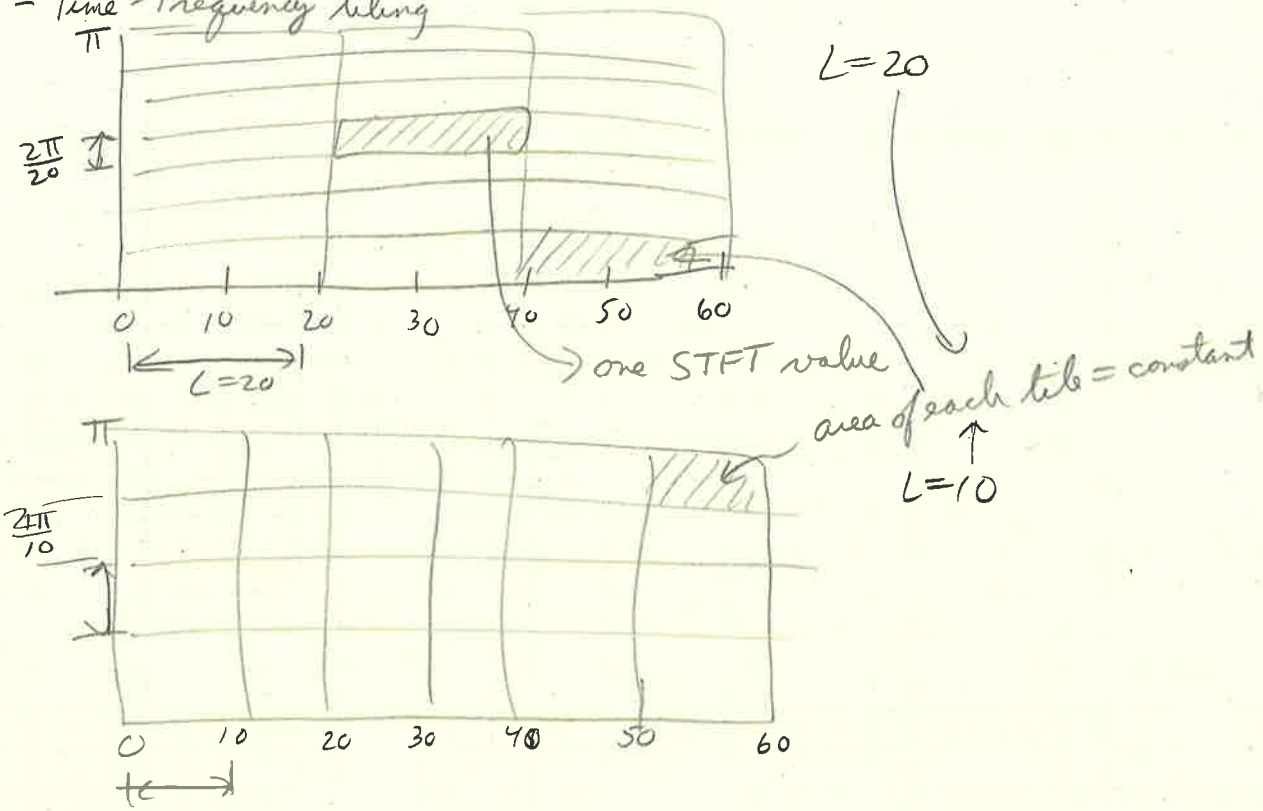
Speech analysis



8 1/2 Hz

There is a lag between thought and act 2.55

- Time-Frequency tiling



- Food for thought
  - time "resolution" $\Delta t = L$
  - frequency "resolution" $\Delta f = 2\pi/L$
  - $\Delta t \Delta f = 2\pi$

uncertainty principle!

- Even more food for thought
  - more sophisticated tilings of time-frequency planes can be obtained with the wavelet transform.

3.5 Discrete Fourier Series

3.5a Discrete Fourier Series

DFS = DFT with periodicity explicit

- The DFS maps an N-periodic signal onto an N-periodic sequence of Fourier coefficients
- The inverse DFS maps an N-periodic sequence of Fourier coefficients onto an N-periodic signal
- The DFS of an N-periodic signal is mathematically equivalent to the DFT of one period

- Finite-length time shifts revisited

The DFS helps us understand how to define time shifts for finite-length signals.

- For an N-periodic sequence $\tilde{x}[n]$
  - $\tilde{x}[n-M]$ is well-defined for all $M \in \mathbb{N}$
  - $\text{DFS}\{\tilde{x}[n-M]\} = e^{-j\frac{2\pi}{N}Mk}\tilde{X}[k]$, $\tilde{X}[k] = \text{DFS}\{\tilde{x}[n]\}$
  - $\text{IDFS}\{\underbrace{e^{-j\frac{2\pi}{N}Mk}}_{n}\tilde{X}[k]\} = \tilde{x}[n-M]$

delay factor

- For an N-point signal $x[n]$:
  - $x[n-M]$ is not well-defined
  - build $\tilde{x}[n] = x[n \bmod N] \Rightarrow \tilde{X}[k] = X[k]$
  - $\text{IDFT}\{e^{-j\frac{2\pi}{N}Mk}X[k]\} = \text{IDFS}\{e^{-j\frac{2\pi}{N}Mk}\tilde{X}[k]\}$
    $= \tilde{x}[n-M] = x[(n-M) \bmod N]$
  - Shifts for finite-length signals are "naturally" circular

3.5b Karplus-Strong revisited and DFS

- Periodic sequences: a bridge to infinite-length signals
  - N-periodic sequence: N degrees of freedom
  - DFS: only N Fourier coefficients capture all of the information

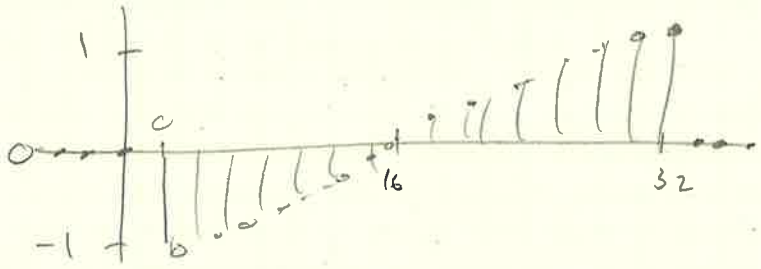
- Karplus-Strong revisited

$$y[n] = \alpha y[n-M] + x[n]$$

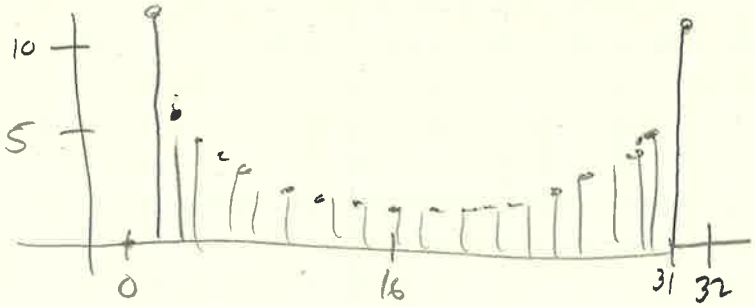
- choose a signal $\bar{x}[n]$ that is nonzero only for $0 \le n < M$
- $\alpha = 1$ (for now)

$$y[n] = \underbrace{\bar{x}[0], \bar{x}[1], \ldots, \bar{x}[M-1]}_{1^{st}\text{ period}}, \underbrace{\bar{x}[0], \bar{x}[1], \ldots, \bar{x}[M-1]}_{2^{nd}\text{ period}}, \underbrace{\bar{x}[0], \bar{x}[1], \ldots}_{\ldots}$$

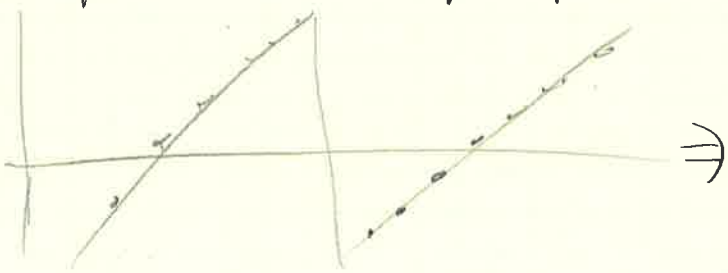
- Example : 32-tap sawtooth wave



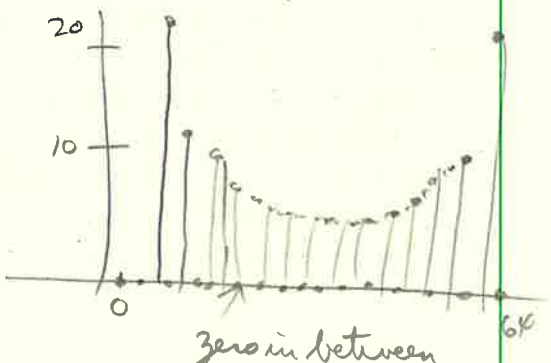
DFT of 32-tap sawtooth wave



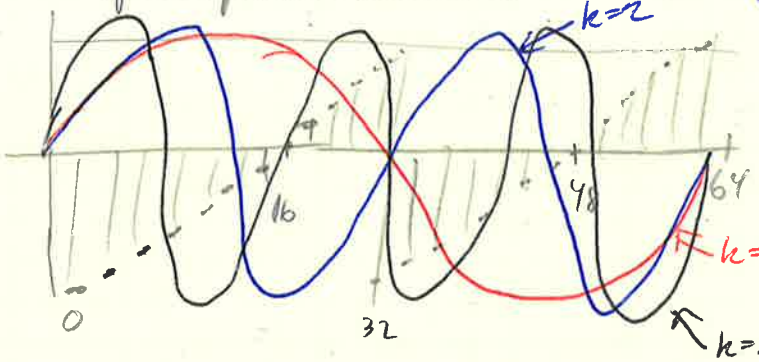
- What if we take the DFT of two periods?



⇒



- DFT of two periods: intuition



inner product gives ~~same~~ DFT coeff twice the original

$k=0$ inner product sums to average

$k=1$ inner product sums to 0

$k=3$ antisymmetry ⇒ inner product sums to 0

(same for all odd k)

- DFT of L periods

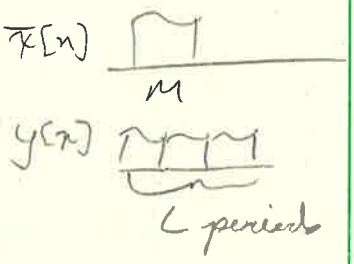
$$X_L[k] = \sum_{n=0}^{LM-1} y[n] e^{-j\frac{2\pi}{LM}nk}, \quad k = 0, 1, 2, \ldots, LM-1$$

$(k = n+pM)$

$$= \sum_{p=0}^{L-1} \sum_{n=0}^{M-1} y[n+pM] e^{-j\frac{2\pi}{LM}(n+pM)k}$$

$$= \sum_{p=0}^{L-1} \sum_{n=0}^{M-1} y[n] e^{-j\frac{2\pi}{LM}nk} e^{-j\frac{2\pi}{L}pk}$$

$$= \left(\sum_{p=0}^{L-1} e^{-j\frac{2\pi}{L}pk} \right) \sum_{n=0}^{M-1} \bar{x}[n] e^{-j\frac{2\pi}{LM}nk}$$



- We've seen this before

$$\sum_{p=0}^{L-1} e^{-j\frac{2\pi}{L}pk} = \begin{cases} L, & \text{if } k \text{ is a multiple of } L \\ 0, & \text{otherwise} \end{cases}$$

(remember the orthogonality proof for the DFT basis)

- DFT of L periods

$$X_L[k] = \begin{cases} L\bar{X}[k/L], & \text{if } k = 0, L, 2L, \ldots \\ 0, & \text{otherwise} \end{cases}$$

- DFT and DFS
  - again, all the spectral information for a periodic signal is contained in the DFT coefficients of a single period.
  - To stress the periodicity of the underlying signal, we use the term DFS

Summary of Lesson 3.5

The discrete Fourier series is just a different flavor of the DFT applied to periodic series. N-periodic sequences in the time domain are mapped onto N-periodic sequences in the frequency domain. Furthermore, the definition of the DFS retrospectively better justifies the use of circular shifts as the natural extension of shifts for finite-length sequences.

Later in the lesson, we have revisited the Karplus-Strong algorithm to illustrate a key point about the DFS. If we take the DFT of L repetitions of a finite length sequence of length N, we obtain a series which is non-zero only at multiple integers of L. Moreover, these non zero coefficients are just scaled versions of the DFT coefficients of the original finite-length sequence. Therefore, all the spectral information of a N-periodic sequence is entirely captured by the DFT coefficients of one period.

3.6 The Discrete-Time Fourier Transform

- The situation so far
  - Fourier representation for signal classes:
    - N-point finite-length: DFT
    - N-point periodic: DFS
    - infinite length: ?
- Karplus-Strong revisited, part 2
  - consider now $\alpha < 1$
  - generated signal is infinite-length but not periodic:

$$y[n] = \underbrace{\bar{x}[0], \bar{x}[1], \ldots, \bar{x}[M-1]}_{1^{st}\text{ period}}, \underbrace{\alpha\bar{x}[0], \alpha\bar{x}[1], \ldots, \alpha\bar{x}[M-1]}_{2^{nd}\text{ period}}, \underbrace{\alpha^2\bar{x}[0], \alpha^2\bar{x}[1] \ldots}_{\ldots}$$

  - What is a good spectral representation?
- DFT of increasingly long signals
  - Start with the DFT. What happens when $N \to \infty$?
  - $(2\pi/N)k$ becomes denser in $[0, 2\pi]$ ... C^N, $\omega = \frac{2\pi}{N}$
  - In the limit $(\frac{2\pi}{N})k \to \omega$: $\sum_n x[n] e^{-j\omega n}$, $\omega \in \mathbb{R}$
- Discrete-Time Fourier Transform (DTFT)
  - Formal definition:
    - $x[n] \in l_2(\mathbb{Z})$

- define a function of $\omega \in \mathbb{R}$

$$F(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$

- inversion (when $F(\omega)$ exists):

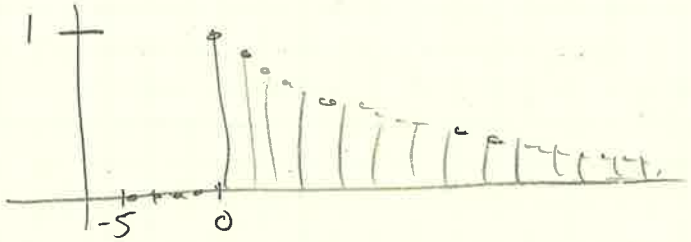
$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\omega) e^{j\omega n} d\omega, \quad n \in \mathbb{Z}$$

- DTFT periodicity and notation
  - $F(\omega)$ is 2π-periodic
  - to stress periodicity (and for other reasons) we will write

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$

  - by convention, $X(e^{j\omega})$ is represented over $[-\pi, \pi]$

- $x[n] = \alpha^n u[n]$, $|\alpha| < 1$



- DTFT of $x[n] = \alpha^n u[n]$, $|\alpha| < 1$

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$

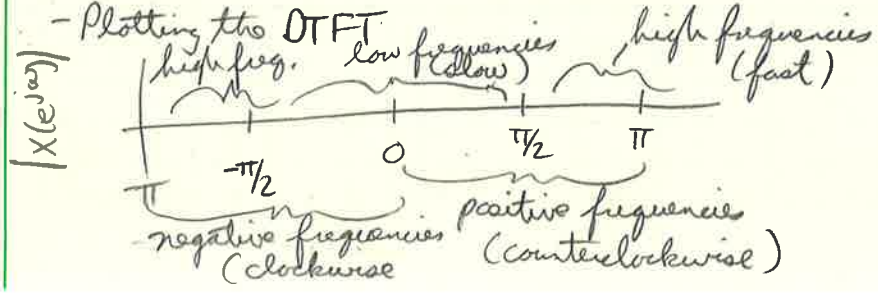
$$= \sum_{n=0}^{\infty} \alpha^n e^{-j\omega n}$$

$$= \sum_{n=0}^{\infty} (\alpha e^{-j\omega})^n$$

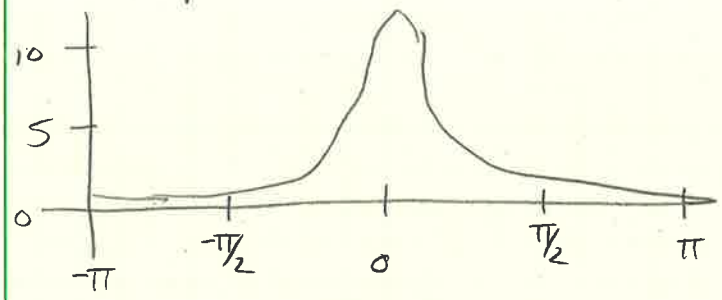
$$= \frac{1}{1 - \alpha e^{-j\omega}}$$

$$|X(e^{j\omega})|^2 = \frac{1}{1 + \alpha^2 - 2\alpha \cos\omega}$$

- Plotting the DTFT

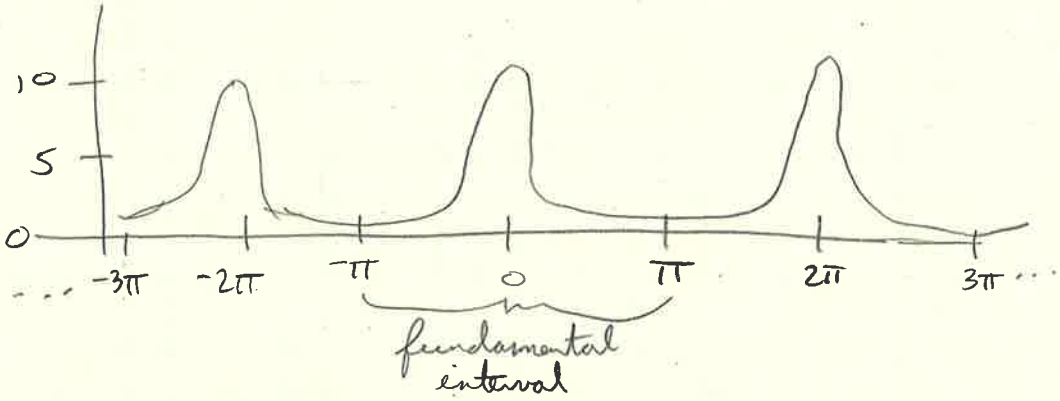


- DTFT of $x[n] = \alpha^n u[n]$, $|\alpha| < 1$



$\alpha = 0.9$

- Remember the periodicity!

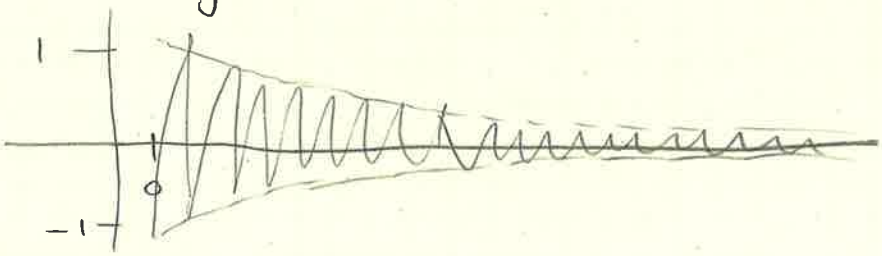


- KS revisited, part 2: 32-tap sawtooth wave

$$x[n] = \frac{2n}{M-1} - 1, \quad n = 0, 1, \ldots, M-1 \qquad (M = 32)$$

- KS revisited, part 2: decay $\alpha = 0.9$

$$y[n] = \alpha^{\lfloor n/M \rfloor} \bar{x}[n \bmod M]\, u[n]$$



- DTFT of KS signal

$$Y(e^{j\omega}) = \sum_{n=-\infty}^{\infty} y[n] e^{-j\omega n}$$

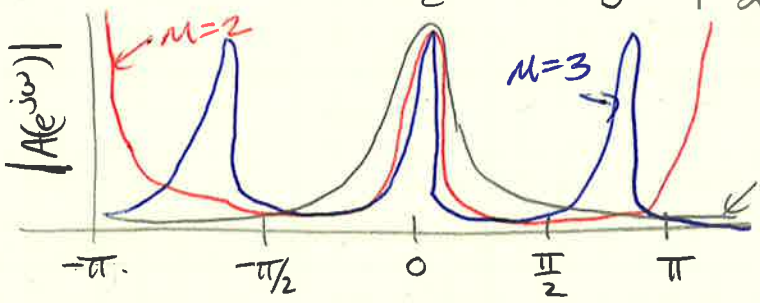
$$= \sum_{p=0}^{\infty} \sum_{n=0}^{M-1} \alpha^p \bar{x}[n] e^{-j\omega(pM+n)}$$

$$= \sum_{p=0}^{\infty} \alpha^p e^{-j\omega Mp} \sum_{n=0}^{M-1} \bar{x}[n] e^{-j\omega n}$$

$$= A(e^{j\omega M})\, \bar{X}(e^{j\omega})$$

- We know the first term

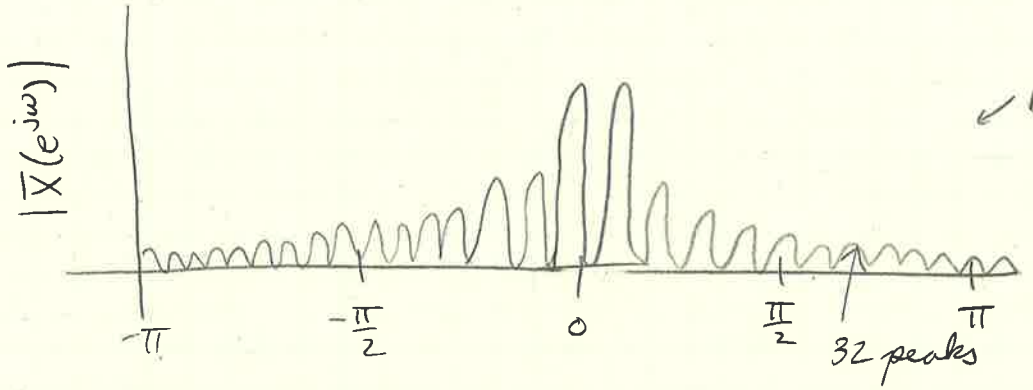
$$A(e^{j\omega}) = \text{DTFT}\{\alpha^n u[n]\} = \frac{1}{1-\alpha e^{-j\omega}} \quad (M=1)$$



$A(e^{j\omega M})$ rescales the frequency axis: periodicity!

- Second term

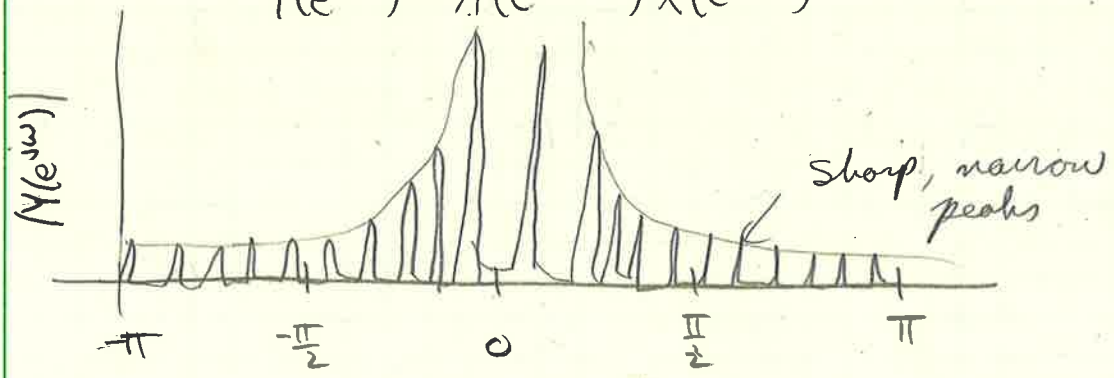
$$\bar{X}(e^{j\omega}) = e^{-j\omega}\left(\frac{M+1}{M-1}\right)\frac{1-e^{-j(M-1)\omega}}{(1-e^{-j\omega})^2} - \frac{1-e^{-j(M+1)\omega}}{(1-e^{-j\omega})^2}$$



↙ DTFT of 32-tap sawtooth

- DTFT of KS with decay

$$Y(e^{j\omega}) = A(e^{j\omega M})\,\bar{X}(e^{j\omega})$$



3.6b Existence and properties of the DTFT

- Existence easy for absolutely summable sequences

$$|X(e^{j\omega})| = \left|\sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}\right|$$

$$\leq \sum_{n=-\infty}^{\infty} |x[n]e^{-j\omega n}| = \sum_{n=-\infty}^{\infty} |x[n]| < \infty$$

- Inversion easy for absolutely summable sequences

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})\, e^{j\omega n}\, d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\sum_{k=-\infty}^{\infty} x[k]\, e^{-j\omega k}\right) e^{j\omega n}\, d\omega$$

$$= \sum_{k=-\infty}^{\infty} x[k] \int_{-\pi}^{\pi} \frac{e^{j\omega(n-k)}}{2\pi}\, d\omega$$

$$= x[n] \qquad = 0 \text{ unless } n=k$$

- A formal change of bases

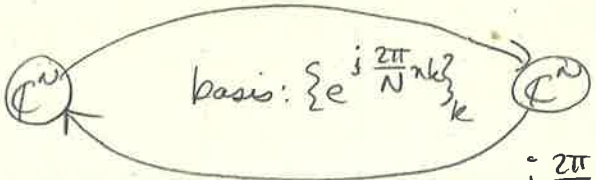
  - Formally DTFT is an inner product in $\mathbb{C}^\infty$:

$$\sum_{n=-\infty}^{\infty} x[n]\, e^{-j\omega n} = \langle e^{j\omega n}, x[n] \rangle$$

  - "basis" is an infinite, uncountable basis: $\{e^{j\omega n}\}_{\omega \in \mathbb{R}}$
  - something "breaks down": we start with sequences but the transform is a function
  - we used absolutely summable sequences but DTFT exists for all square-summable sequences (proof is rather technical)

- Review: DFT

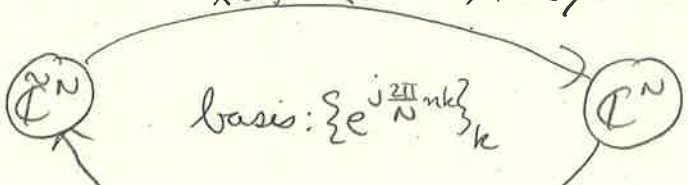
$$X[k] = \langle e^{j\frac{2\pi}{N}nk}, x[n] \rangle$$



$$x[n] = \frac{1}{N}\sum X[k]\, e^{j\frac{2\pi}{N}nk}$$

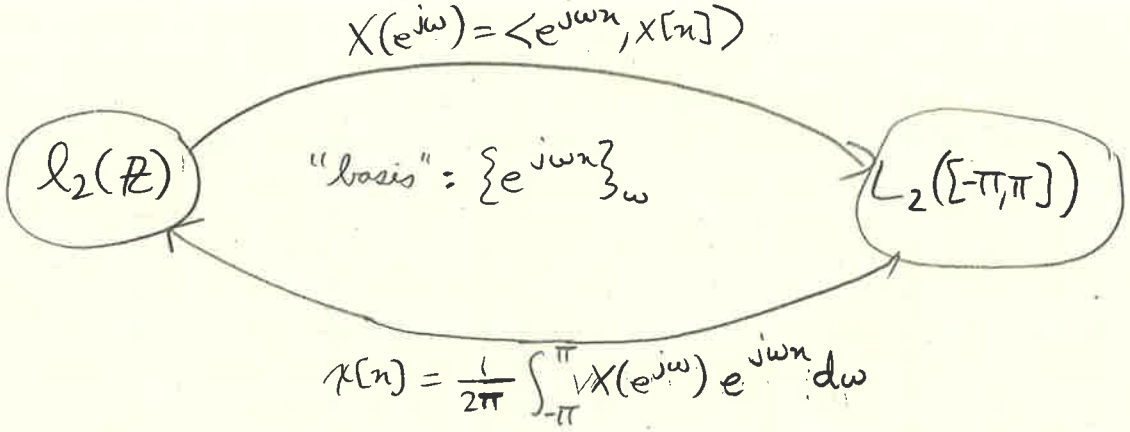
- Review: DFS

$$\tilde{X}[k] = \langle e^{j\frac{2\pi}{N}nk}, \tilde{x}[n] \rangle$$



$$\tilde{x}[n] = \frac{1}{N}\sum \tilde{X}[k]\, e^{j\frac{2\pi}{N}nk}$$

- DTFT



- DTFT properties
  - linearity: $\text{DTFT}\{\alpha x[n] + \beta y[n]\} = \alpha X(e^{j\omega}) + \beta Y(e^{j\omega})$
  - time shift: $\text{DTFT}\{x[n-M]\} = e^{-j\omega M} X(e^{j\omega})$
  - modulation (dual): $\text{DTFT}\{e^{j\omega_0 n} x[n]\} = X(e^{j(\omega-\omega_0)})$
  - time reversal: $\text{DTFT}\{x[-n]\} = X(e^{-j\omega})$
  - conjugation: $\text{DTFT}\{x^*[n]\} = X^*(e^{-j\omega})$
- Some particular cases:
  - if $x[n]$ is symmetric, the DTFT is symmetric:
    $$x[n] = x[-n] \iff X(e^{j\omega}) = X(e^{-j\omega})$$
  - if $x[n]$ is real, the DTFT is Hermitian-symmetric:
    $$x[n] = x^*[n] \iff X(e^{j\omega}) = X^*(e^{-j\omega})$$
  - special case: if $x[n]$ is real, the magnitude of the DTFT is symmetric:
    $$x[n] \in \mathbb{R} \Rightarrow |X(e^{j\omega})| = |X(e^{-j\omega})|$$
  - more special case: if $x[n]$ is real and symmetric, $X(e^{j\omega})$ is also real and symmetric.

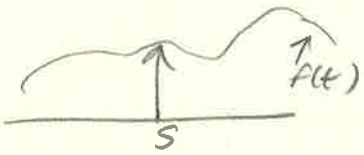
3.6c The DTFT as a change of basis

- DTFT as basis expansion
  - Some things are OK:
    - $\text{DFT}\{\delta[n]\} = 1$
    - $\text{DTFT}\{\delta[n]\} = \langle e^{j\omega n}, \delta[n] \rangle = 1$

- Some things aren't:
  - $\text{DFT}\{1\} = N\delta[k]$
  - $\text{DTFT}\{1\} = \sum_{n=-\infty}^{\infty} e^{-j\omega n} = ?$
  - problem: too many interesting sequences are not square-summable!

- The Dirac delta functional

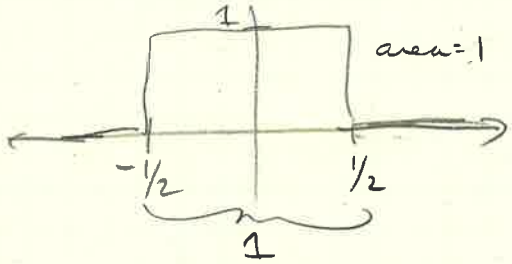
- Defined by the "sifting" property:



$$\int_{-\infty}^{\infty} \delta(t-s) f(t)\,dt = f(s), \quad \forall \text{ functions of } t \in \mathbb{R}, \; s \in \mathbb{R}$$

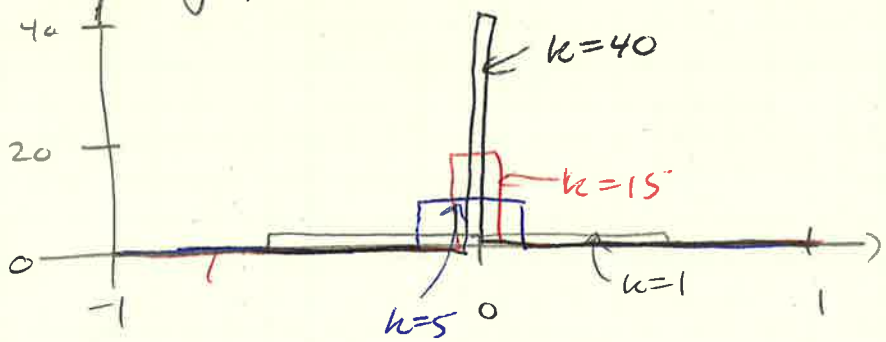
- Intuition
  - Family of localizing functions $r_k(t)$ with $k \in \mathbb{N}$ and $t \in \mathbb{R}$
  - Support inversely proportional to k
  - constant area

$$\text{rect}(t) = \begin{cases} 1, & |t| < \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}$$



- Consider the localizing family $r_k(t) = k\,\text{rect}(kt)$:
  - nonzero over $\left[-\frac{1}{2k}, \frac{1}{2k}\right]$, i.e., support is $\frac{1}{k}$
  - area is 1

- The family $r_k(t) = k\,\text{rect}(kt)$



- Extracting a point value

- By the mean value theorem:

$$\int_{-\infty}^{\infty} r_k(t) f(t)\,dt = k \int_{-\frac{1}{2k}}^{\frac{1}{2k}} f(t)\,dt = f(y)\Big|_{y \in \left[-\frac{1}{2k}, \frac{1}{2k}\right]}$$

and so:

$$\lim_{k\to\infty} \int_{-\infty}^{\infty} r_k(t) f(t)\,dt = f(0)$$

- The Dirac delta *functional*

"The delta functional shorthand. Instead of writing

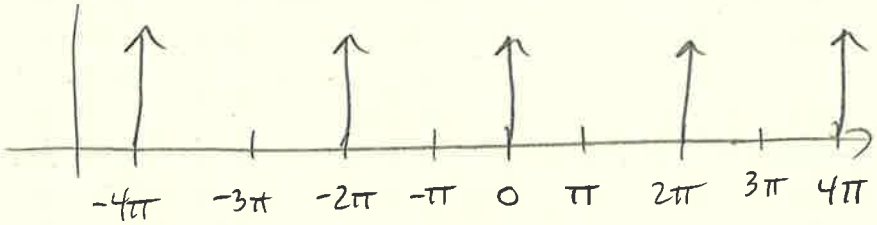
$$\lim_{k\to\infty} \int_{-\infty}^{\infty} r_k(t-s) f(t)\,dt$$

we write $\int_{-\infty}^{\infty} \delta(t-s) f(t)\,dt$, as if $\lim_{k\to\infty} r_k(t) = \delta(t)$.

- The "pulse train"

$$\tilde{\delta}(\omega) = 2\pi \sum_{k=-\infty}^{\infty} \delta(\omega - 2\pi k)$$

just a technicality to use the Dirac delta in the space of 2π-periodic functions



$$\text{IDTFT}\{\tilde{\delta}(\omega)\} = \frac{1}{2\pi}\int_{-\pi}^{\pi} \tilde{\delta}(\omega) e^{j\omega n}\,d\omega$$

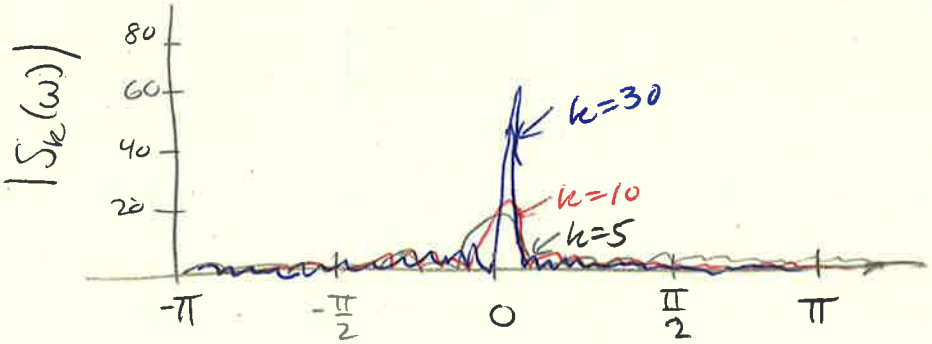
$$= \int_{-\pi}^{\pi} \delta(\omega)\, e^{-j\omega n}\,d\omega = e^{j\omega n}\Big|_{\omega=0} = 1$$

$$(\text{IDFT}\{N\delta[k]\} = 1)$$

$$\Rightarrow \text{DTFT}\{1\} = \tilde{\delta}(\omega)$$

- Does this make sense?

• Partial DTFT sum: $S_k(\omega) = \sum_{n=-k}^{k} e^{-j\omega n}$



- Using the same technique

$$\text{IDTFT}\{\tilde{\delta}(\omega-\omega_0)\} = e^{j\omega_0 n}$$

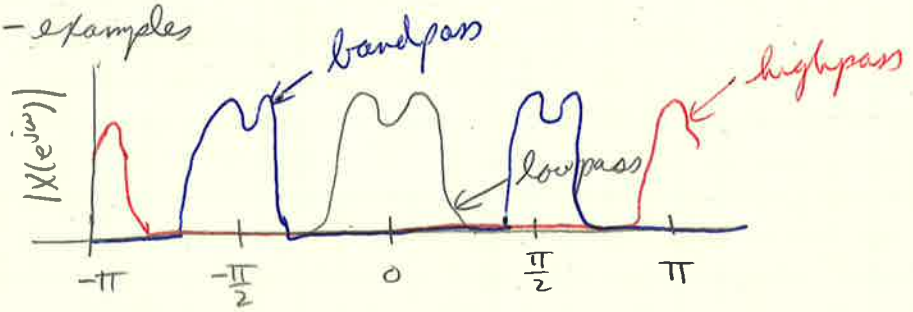
$\Rightarrow$
- $\text{DTFT}\{1\} = \tilde{\delta}(\omega)$
- $\text{DTFT}\{e^{j\omega_0 n}\} = \tilde{\delta}(\omega-\omega_0)$
- $\text{DTFT}\{\cos\omega_0 n\} = [\tilde{\delta}(\omega-\omega_0)+\tilde{\delta}(\omega+\omega_0)]/2$
- $\text{DTFT}\{\sin\omega_0 n\} = -j[\tilde{\delta}(\omega-\omega_0)-\tilde{\delta}(\omega+\omega_0)]/2$

3.7: Sinusoidal Modulation

3.7a Sinusoidal modulation

- Classifying signals in frequency
  - Three broad categories according to where most of the spectral energy resides:
    - lowpass signals (also known as "baseband" signals)
    - highpass signals
    - bandpass signals

- examples



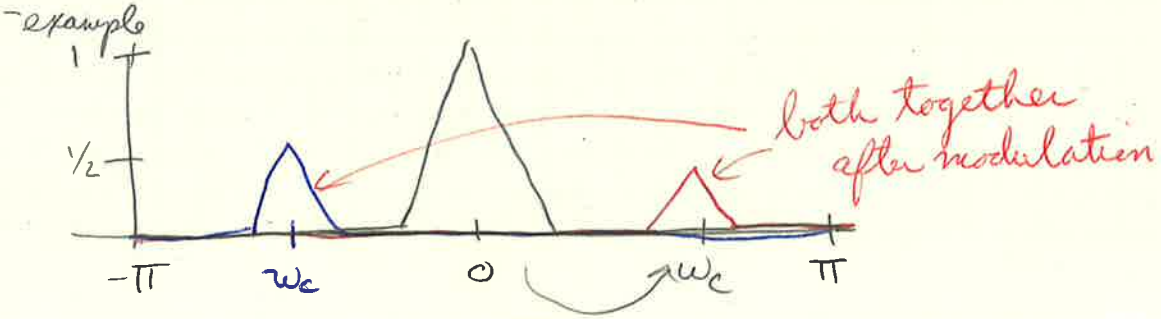
- Sinusoidal modulation

$$\text{DTFT}\{x[n]\cos(\omega_c n)\} = \text{DTFT}\left\{\frac{1}{2}e^{j\omega_c n}x[n] + \frac{1}{2}e^{-j\omega_c n}x[n]\right\}$$

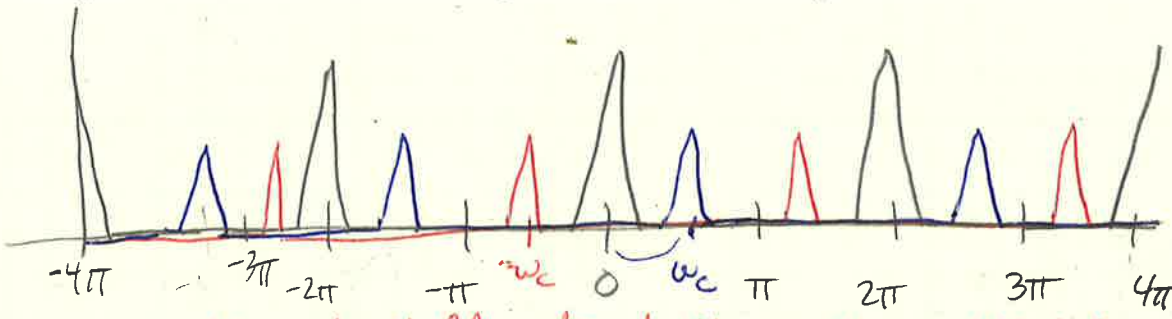
$$= \frac{1}{2}\left[X(e^{j(\omega-\omega_c)}) + X(e^{j(\omega+\omega_c)})\right]$$

- usually $x[n]$ baseband
- ω_c is the carrier frequency

- example

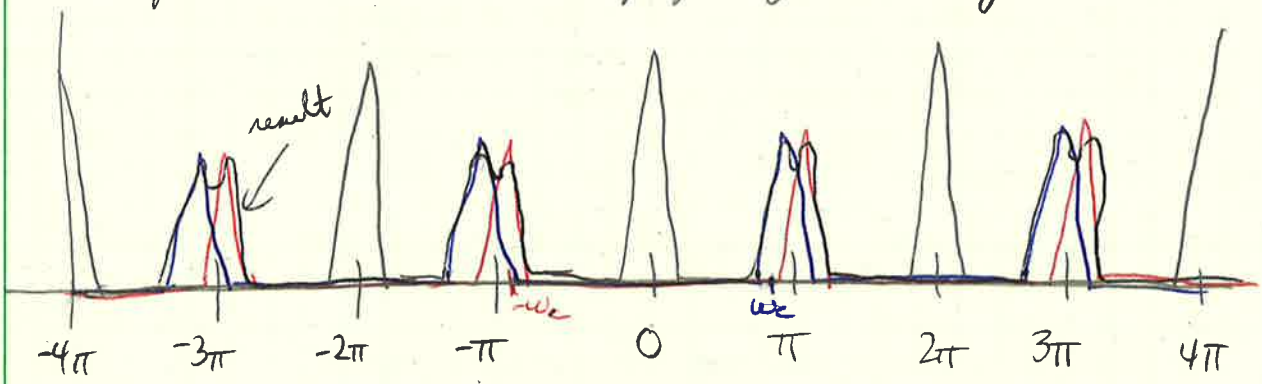


- Again, explicitly showing the periodicity of the spectrum



combine red and blue to get the resulting modulated spectrum

- Careful when the modulation frequency is too large!



- Sinusoidal modulation: applications

- voice and music are lowpass signals
- radio channels are bandpass, at much higher frequencies
- modulation brings the baseband signal in the transmission band
- demodulation at the receiver brings it back

- Sinusoidal demodulation
  - just multiply the received signal by the carrier again

$$y[n] = x[n]\cos(\omega_c n), \quad Y(e^{j\omega}) = \frac{1}{2}\left[X(e^{j(\omega-\omega_c)}) + X(e^{j(\omega+\omega_c)})\right]$$

$$\text{DTFT}\{y[n]\cdot 2\cos(\omega_c n)\} = Y(e^{j(\omega-\omega_c)}) + Y(e^{j(\omega+\omega_c)})$$

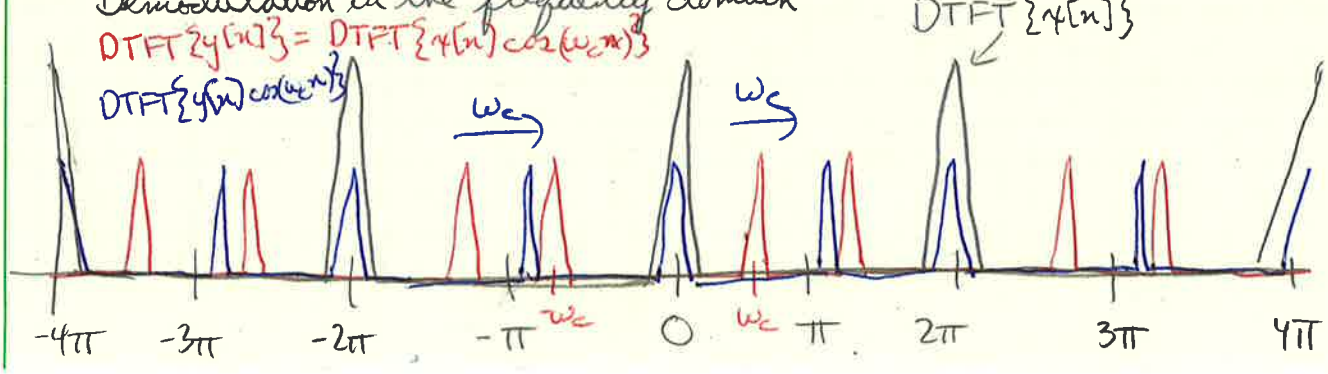
$$= \frac{1}{2}\left[X(e^{j(\omega-2\omega_c)}) + X(e^{j\omega}) + X(e^{j\omega}) + X(e^{j(\omega+2\omega_c)})\right]$$

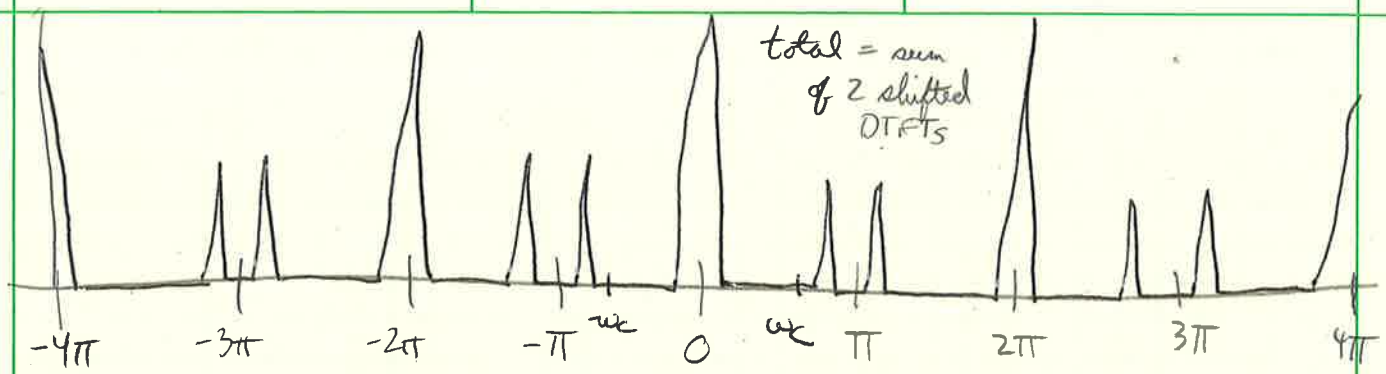
$$= X(e^{j\omega}) + \frac{1}{2}\left[X(e^{j(\omega-2\omega_c)}) + X(e^{j(\omega+2\omega_c)})\right]$$

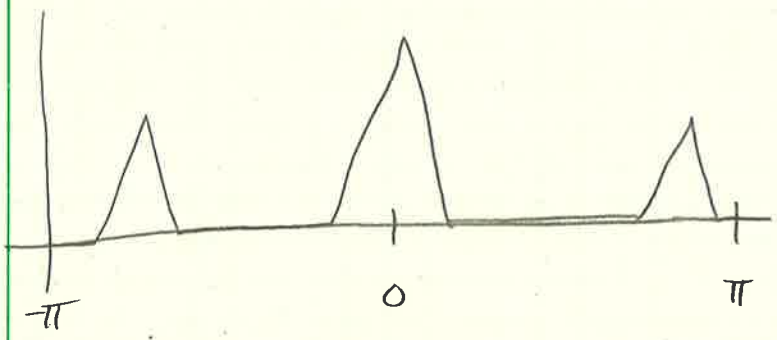
- Demodulation in the frequency domain

$\text{DTFT}\{y[n]\} = \text{DTFT}\{x[n]\cos(\omega_c n)\}$

$\text{DTFT}\{y[n]\cos(\omega_c n)\}$







DTFT of demodulated signal

- we recovered the baseband signal exactly ...
- but we have some spurious high-frequency components
- in the next Module we will learn how to get rid of them!

3.7b Tuning a guitar

- Problem (abstraction):
  - reference sinusoid at frequency ω_0
  - tunable sinusoid at frequency ω
  - make $\omega = \omega_0$ "by ear"

- The procedure
  1. bring ω close to ω_0 (easy)
  2. when $\omega \approx \omega_0$ play both sinusoids together
  3. trigonometry comes to the rescue

$$x[n] = \cos(\omega_0 n) + \cos(\omega n)$$
$$= 2\cos\left(\frac{\omega_0+\omega}{2}n\right)\cos\left(\frac{\omega_0-\omega}{2}n\right)$$
$$\approx 2\cos(\Delta_\omega n)\cos(\omega_0 n)$$

- Let's see what's happening

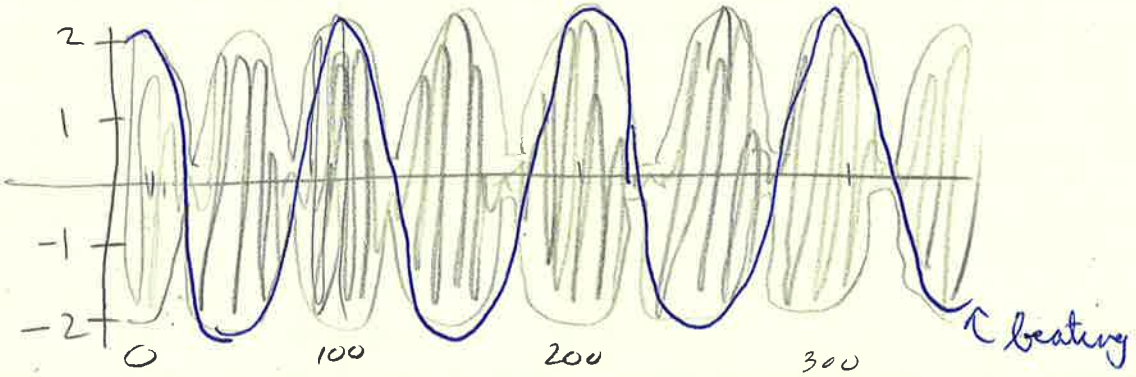
$$x[n] \approx \underbrace{2\cos(\Delta_\omega n)}\ \underbrace{\cos(\omega_0 n)}$$

"error" signal

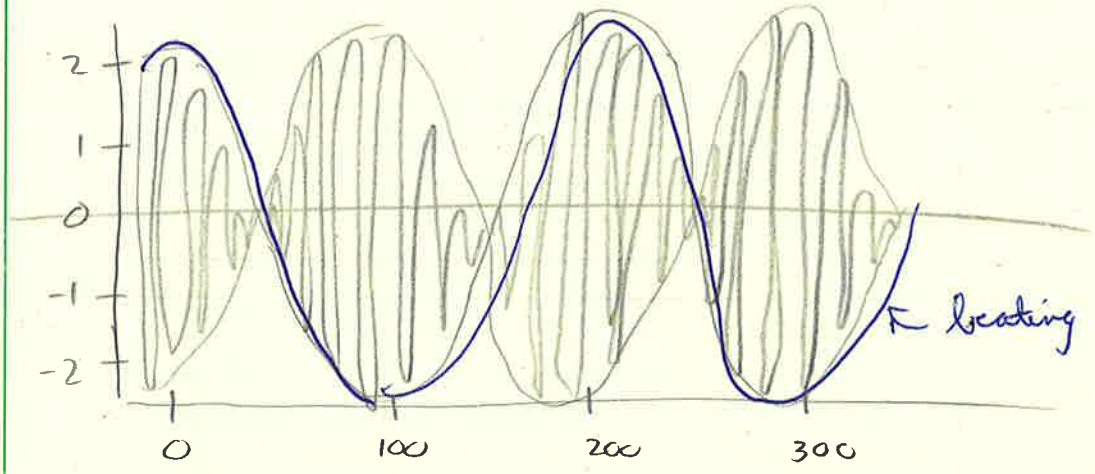
- modulation at ω_0

- when $\omega \approx \omega_0$, the error signal is too low to be heard; modulation brings it up to hearing range and we perceive it as amplitude oscillations of the carrier frequency

- In the time domain ...



$\omega_0 = 2\pi(0.2)$, $\omega = 2\pi(0.22)$, $\Delta\omega = 2\pi(0.0100)$



$\omega_0 = 2\pi(0.2)$, $\omega = 2\pi(0.21)$, $\Delta\omega = 2\pi(0.0050)$

- A Detour on Western Musical Conventions

reference note ↓

- Each note has a unique frequency, $A_3 = 220$ Hz, $A_4 = 440$ Hz, $A_5 = 880$ Hz.
- An octave corresponds to doubling/halving frequency
- Octaves are separated into 12 evenly spaced half-tones

$$f_h = f_0 \cdot 2^{h/12}$$

3.8 Relationship between transforms

- Overview
  - DFT, DFS, DTFT
  - DTFT of periodic sequences
  - DTFT of finite-support sequences
  - Zero padding

- Transforms
  - DFT, DFS: change of basis in $\mathbb{C}^N$
  - DTFT: "formal" change of basis in $l_2(\mathbb{Z})$
  - basis vectors are "building blocks" for any signal
  - DFT: numerical algorithm (computable)
  - DTFT: mathematical tool (proofs)

- Embedding finite-length signals
  - N-tap signal $x[n]$
  - natural spectral representation: DFT $X[k]$
  - two ways to embed $x[n]$ into an infinite sequence:
    - periodic extension: $\tilde{x}[n] = x[n \bmod N]$
    - finite support extension: $\bar{x}[n] = \begin{cases} x[n], & 0 \le n < N \\ 0, & \text{otherwise} \end{cases}$
  - how does $X[k]$ relate to the DTFT of the embedded signals?

- DTFT of periodic signals

$$\tilde{x}[n] = x[n \bmod N]$$

$$\tilde{X}(e^{j\omega}) = \sum_{n=-\infty}^{\infty} \tilde{x}[n] e^{-j\omega n}$$

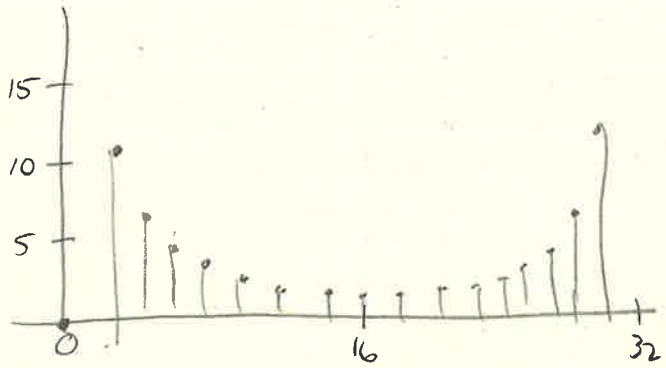
$$= \sum_{n=-\infty}^{\infty} \left(\frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}[k] e^{j\frac{2\pi}{N}nk} \right) e^{-j\omega n}$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} X[k] \left(\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi}{N}nk} e^{-j\omega n} \right)$$

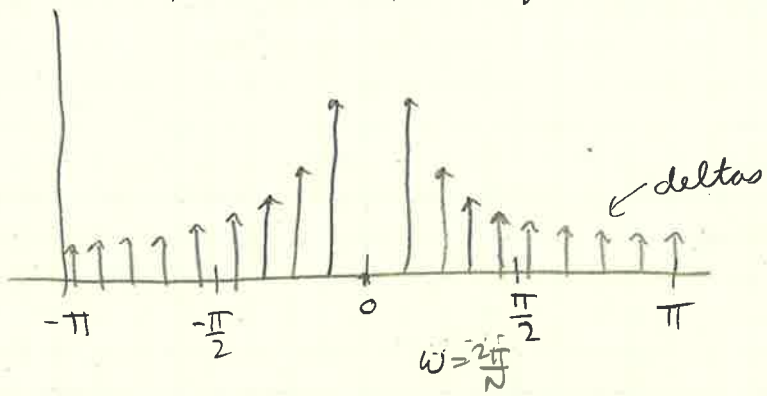
$$\left(\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi}{N}nk} e^{-j\omega n} = \text{DTFT}\left\{ e^{j\frac{2\pi}{N}nk} \right\} = \tilde{\delta}\left(\omega - \tfrac{2\pi}{N}k\right) \right)$$

$$\tilde{X}(e^{j\omega}) = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \, \tilde{\delta}\left(\omega - \tfrac{2\pi}{N}k\right)$$

- DFT of 32-tap sawtooth



- DTFT of periodic extension of 32-tap sawtooth



- DTFT of finite-support signals

$$\bar{x}[n] = \begin{cases} x[n] & , \ 0 \le n < N \\ 0 & , \ \text{otherwise} \end{cases}$$

$$\bar{X}(e^{j\omega}) = \sum_{n=-\infty}^{\infty} \bar{x}[n] e^{-j\omega n} = \sum_{n=0}^{N-1} x[n] e^{-j\omega n}$$

$$= \sum_{n=0}^{N-1} \left(\frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}nk} \right) e^{-j\omega n}$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} X[k] \left(\sum_{n=0}^{N-1} e^{-j(\omega - \frac{2\pi}{N}k)n} \right)$$

$$\left[\sum_{n=0}^{N-1} e^{-j(\omega - \frac{2\pi}{N}k)n} = \bar{R}\left(e^{j(\omega - \frac{2\pi}{N}k)}\right) \text{, where } \bar{R}(e^{j\omega}) \text{ is the DTFT of } r[n], \text{ the interval indicator signal: } r[n] = \begin{cases} 1 & , \ 0 \le n < N \\ 0 & , \ \text{otherwise} \end{cases} \right]$$

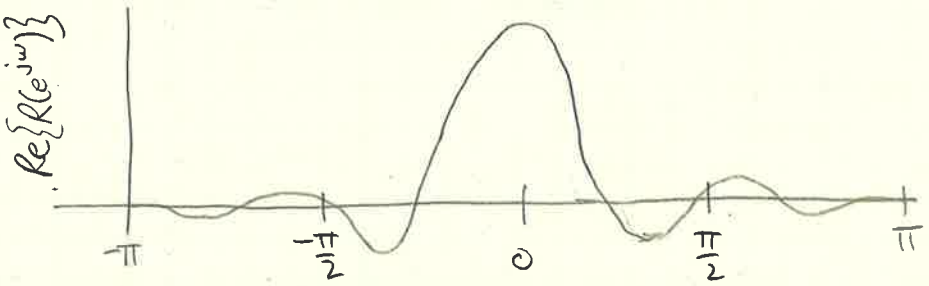
- DTFT of interval signal

$$\overline{R}(e^{j\omega}) = \sum_{n=0}^{N-1} e^{-j\omega n} = \frac{1 - e^{-j\omega N}}{1 - e^{-j\omega}}$$

$$= \frac{e^{-j\frac{\omega N}{2}}\left[e^{j\frac{\omega N}{2}} - e^{-j\frac{\omega N}{2}}\right]}{e^{-j\frac{\omega}{2}}\left[e^{j\frac{\omega}{2}} - e^{-j\frac{\omega}{2}}\right]}$$

$$= \frac{\sin\left(\frac{\omega}{2}N\right)}{\sin\left(\frac{\omega}{2}\right)} e^{-j\frac{\omega}{2}(N-1)}$$

- DTFT of interval signal ($N=9$)

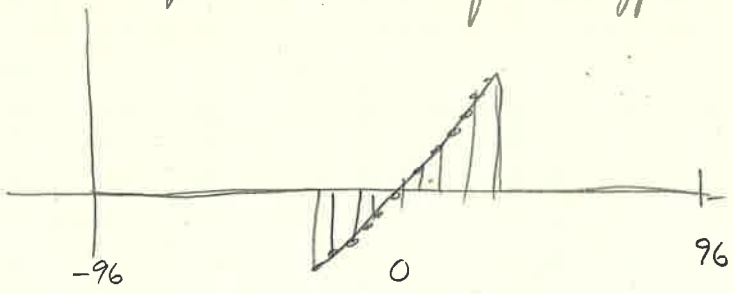


- DTFT of finite-support signals cont'd

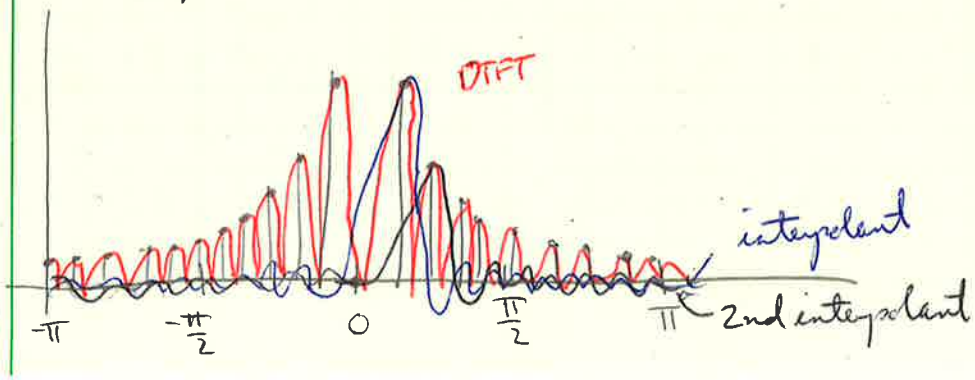
$$\overline{X}(e^{j\omega}) = \sum_{k=0}^{N-1} X[k]\,\Lambda\left(\omega - \frac{2\pi}{N}k\right), \text{ where } \Lambda(\omega) = \frac{1}{N}\overline{R}(e^{j\omega}):$$

smooth interpolation of DFT values.

- 32 tap sawtooth with finite support extension



- DTFT of finite support extension (sketch)



- About zero-padding

"When computing the DFT numerically, one may "pad" the data vector with zeros to obtain "nicer" plots"

$$X_M[k] = \sum_{n=0}^{M-1} x'[n]\, e^{-j\frac{2\pi}{M}nk} = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{M}nk}$$

($x'[n]$: extension of $x[n]$)

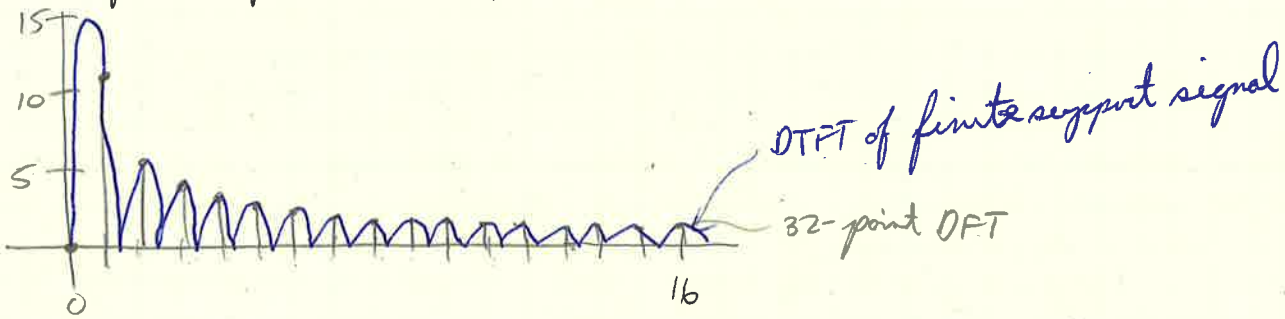
$$= \sum_{n=0}^{N-1}\left(\frac{1}{N}\sum_{k=0}^{N-1} X_N[k]\, e^{j\frac{2\pi}{N}nk}\right) e^{-j\frac{2\pi}{M}nk}$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} X_N[k]\left(\sum_{n=0}^{N-1} e^{-j\left(\frac{2\pi}{M}k - \frac{2\pi}{N}k\right)n}\right)$$

$$= \overline{X}(e^{j\omega})\Big|_{\omega=\frac{2\pi}{M}k}$$

- zero padding does not add information
- a zero-padded DFT is simply a sampled DTFT of the finite-support extension

- DFT of 32-tap sawtooth, zero-padded



4.1.a Linear time-invariant filters

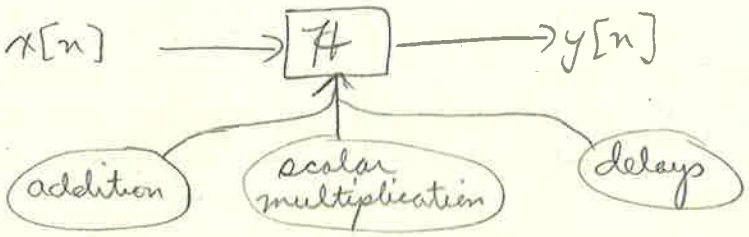
- A generic signal processing device

$$x[n] \longrightarrow \boxed{\mathcal{H}} \longrightarrow y[n]$$

$$y[n] = \mathcal{H}\{x[n]\}$$

- Linearity: $\mathcal{H}\{\alpha x_1[n] + \beta x_2[n]\} = \alpha \mathcal{H}\{x_1[n]\} + \beta \mathcal{H}\{x_2[n]\}$
- Time Invariance: $y[n] = \mathcal{H}\{x[n]\} \Leftrightarrow \mathcal{H}\{x[n-n_0]\} = y[n-n_0]$

- Linear, time-invariant systems (LTI)



$y[n] = H(x[n], x[n-1], x[n-2], \ldots, y[n-1], y[n-2], \ldots)$ (causal LTI)
with $H(\cdot)$ a linear function of its arguments

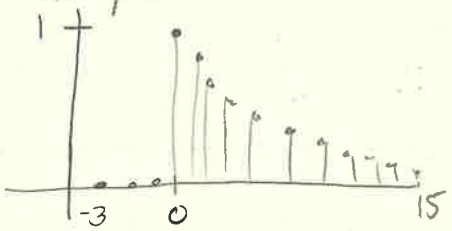
4.1.b Convolution

- Impulse response

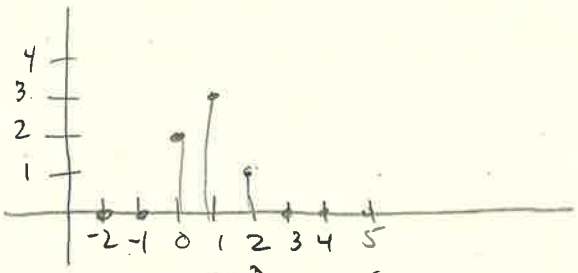
$$h[n] = \mathcal{H}\{\delta[n]\}$$

- Fundamental result: impulse response fully characterizes the LTI system!

- Example



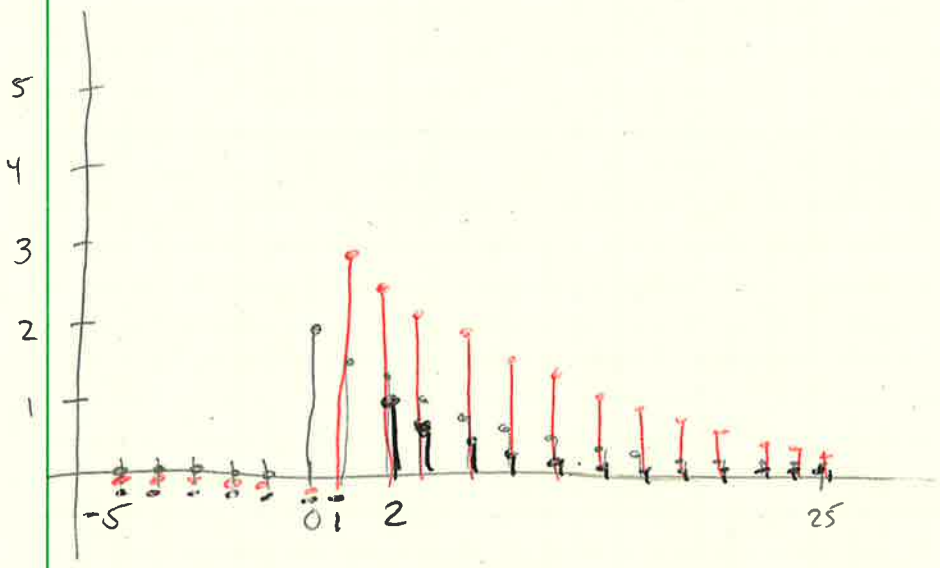
$h[n] = \alpha^n u[n]$



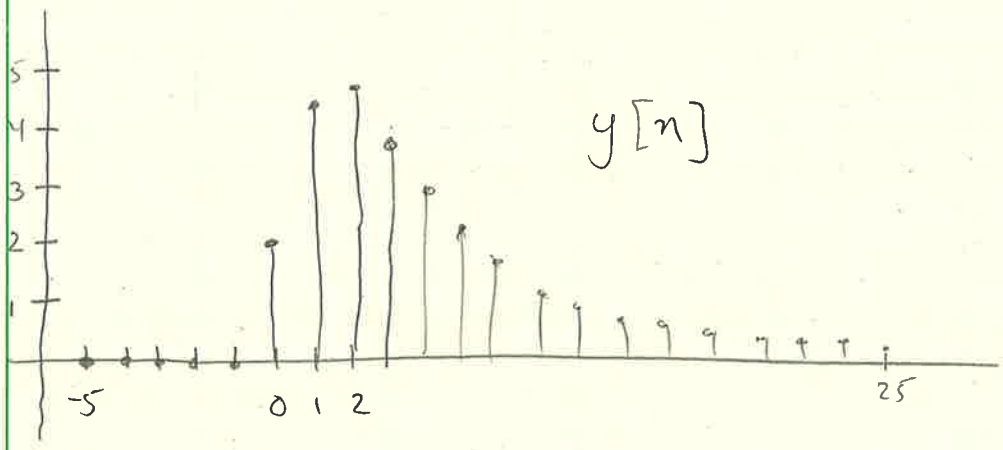
$$x[n] = \begin{cases} 2, & n=0 \\ 3, & n=1 \\ 1, & n=2 \\ 0, & \text{otherwise} \end{cases}$$

- $x[n] = 2\delta[n] + 3\delta[n-1] + \delta[n-2]$
- we know the impulse response $h[n] = \mathcal{H}\{\delta[n]\}$
- compute $y[n] = \mathcal{H}\{x[n]\}$ exploiting linearity and time-invariance

$$y[n] = \mathcal{H}\{2\delta[n] + 3\delta[n-1] + \delta[n-2]\}$$
$$= 2\mathcal{H}\{\delta[n]\} + 3\mathcal{H}\{\delta[n-1]\} + \mathcal{H}\{\delta[n-2]\}$$
$$= 2h[n] + 3h[n-1] + h[n-2]$$



- $2h[n]$
- $3h[n-1]$
- $h[n-2]$



- Convolution

We can always write (Module 3.2):

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

by linearity and time invariance: $y[n] = \sum_{k=-\infty}^{\infty} x[k]\,h[n-k] = x[n] * h[n]$

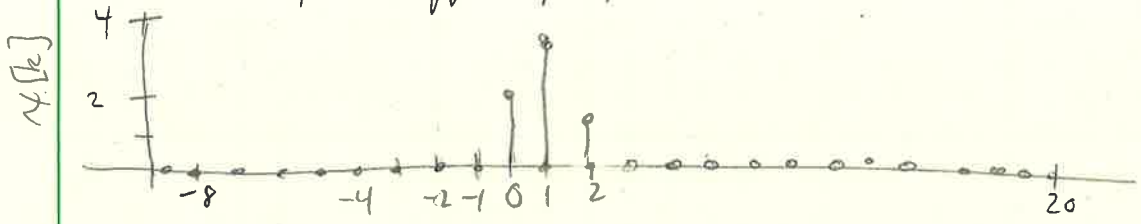
- Performing the convolution algorithmically

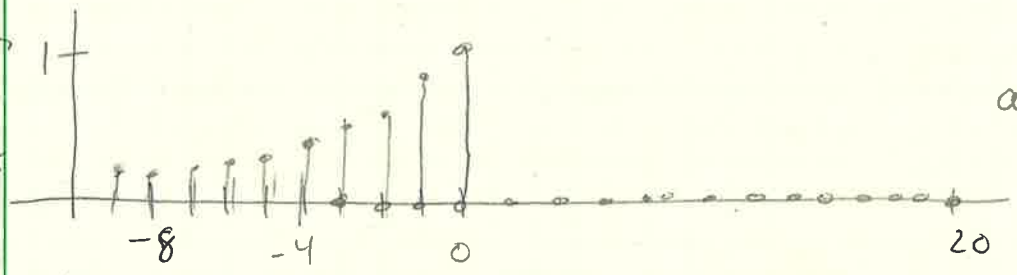
$$x[n] * y[n] = \sum_{k=-\infty}^{\infty} x[k]\,h[n-k]$$

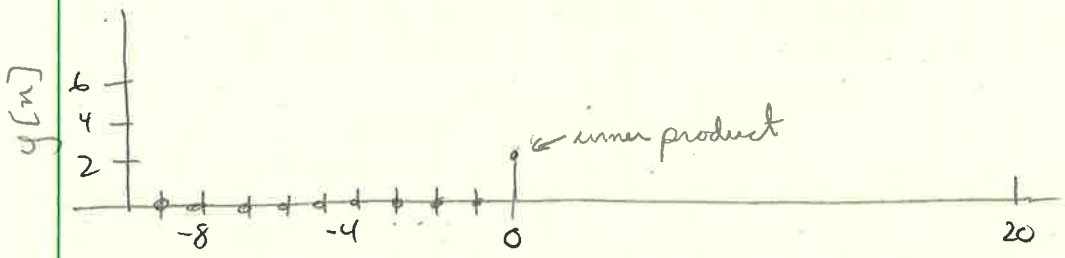
Ingredients:
- a sequence $x[n]$
- a second sequence $h[n]$

The recipe:
- time reverse $h[n]$
- at each step n (from $-\infty$ to ∞)
  - center the time-reversed $h[n]$ in n (i.e. shift by $-n$)
  - Compute the inner product

- Same example, different perspective







- Convolution properties
  - linearity and time invariance (by definition)
  - commutativity: $(x*h)[n] = (h*x)[n]$
  - associativity for absolutely and square-summable sequences:
    $((x*h)*w)[n] = (x*(h*w))[n]$

$$x[n] \longrightarrow \boxed{h[n]} \longrightarrow \boxed{w[n]} \longrightarrow y[n]$$

$$x[n] \rightarrow \boxed{(h*w)[n]} \longrightarrow y[n]$$

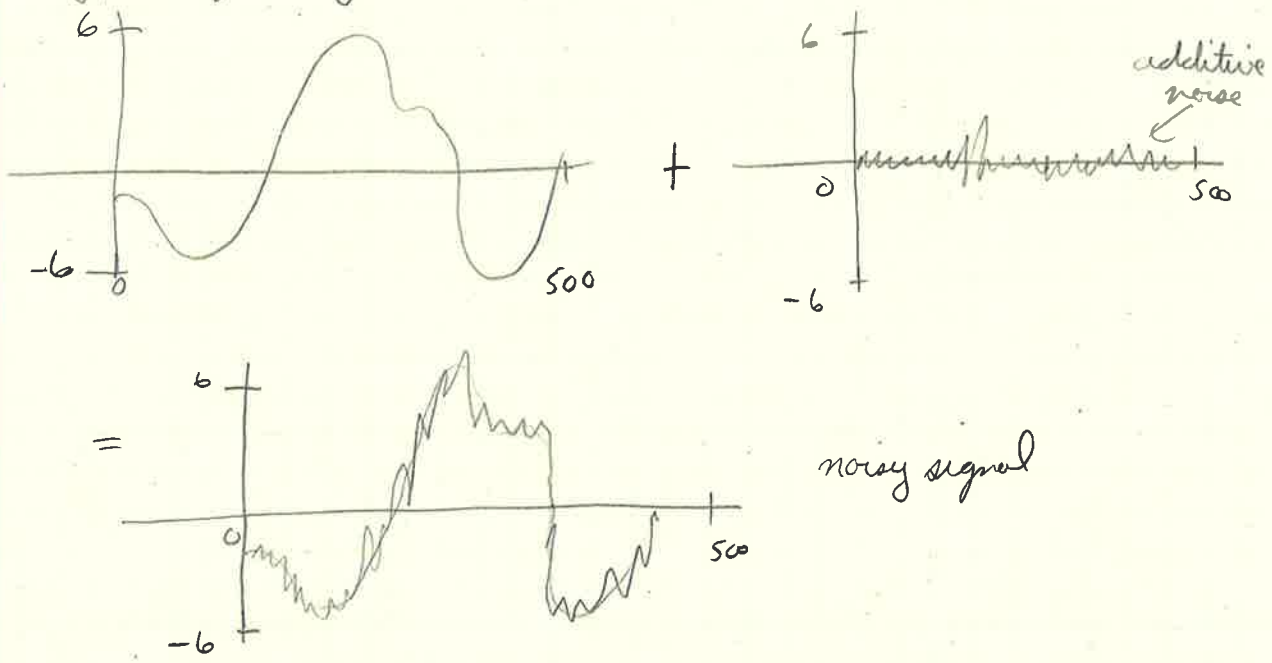
Signal of the Day: Can one hear the shape of a room?

- Our Model: Room Impulse Response (RIR)
  - Linear model
    - Sound level in room acoustics is low
    - Linear model is good approximation
    - Entirely characterized by impulse response, i.e., response to Dirac impulse
  - Room impulse response
    - Describe audio channel between sender S and receiver R
    - Sum up effect of direct path transmission and subsequent attenuated echoes.

4.2 Filtering by Example

4.2.a The moving average filter

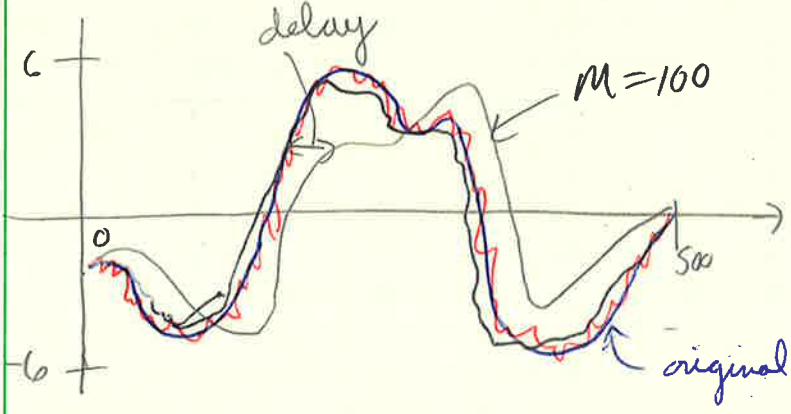
- Typical filtering scenario: denoising



noisy signal

- Denoising by Moving Average (MA)
  - idea: replace each sample by the local average
  - for instance: $y[n] = (x[n] + x[n-1])/2$
  - more generally:

$$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k]$$

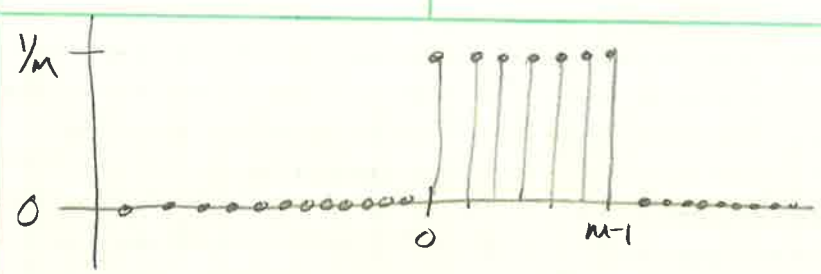


- $M = 2$
- $M = 12$

Smoother as M gets larger, but output gets delayed

- MA: impulse response

$$h[n] = \frac{1}{M} \sum_{k=0}^{M-1} \delta[n-k] = \begin{cases} 1/M, & 0 \le n < M \\ 0, & \text{otherwise} \end{cases}$$



- MA: analysis
  - smoothing effect proportional to M
  - number of operations and storage also proportional to M

- From the MA to a first-order recursion

$$y_M[n] = \frac{1}{M}\left(x[n] + x[n-1] + \dots + x[n-M+1]\right)$$ – Moving average over M points

$$y_M[n] = \frac{1}{M}x[n] + \frac{1}{M}\underbrace{\left(x[n-1] + \dots + x[n-M+1]\right)}$$

"almost" $y_{M-1}[n-1]$, i.e. moving average over M-1 points, delayed by one

Formally:

$$y_M[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k]$$

$$y_M[n-1] = \frac{1}{M}\sum_{k=0}^{M-1} x[(n-1)-k] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-(k+1)]$$

$$= \frac{1}{M}\sum_{k=0+1}^{M-1+1} x[n-k] = \frac{1}{M}\sum_{k=1}^{M} x[n-k]$$

$$y_{M-1}[n] = \frac{1}{M-1}\sum_{k=0}^{M-2} x[n-k]$$

$$y_{M-1}[n-1] = \frac{1}{M-1}\sum_{k=1}^{M-1} x[n-k]$$

$$y_M[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k] \qquad y_{M-1}[n-1] = \frac{1}{M-1}\sum_{k=1}^{M-1} x[n-k]$$

$$\sum_{k=0}^{M-1} x(n-k) = x[n] + \sum_{k=1}^{M-1} x[n-k]$$

$$M y_M[n] = x[n] + (M-1)\, y_{M-1}[n-1]$$

$$\Leftrightarrow y_M[n] = \frac{M-1}{M} y_{M-1}[n-1] + \frac{1}{M}x[n]$$

$$y_M[n] = \lambda\, y_{M-1}[n-1] + (1-\lambda)\, x[n], \quad \lambda = \frac{M-1}{M}$$

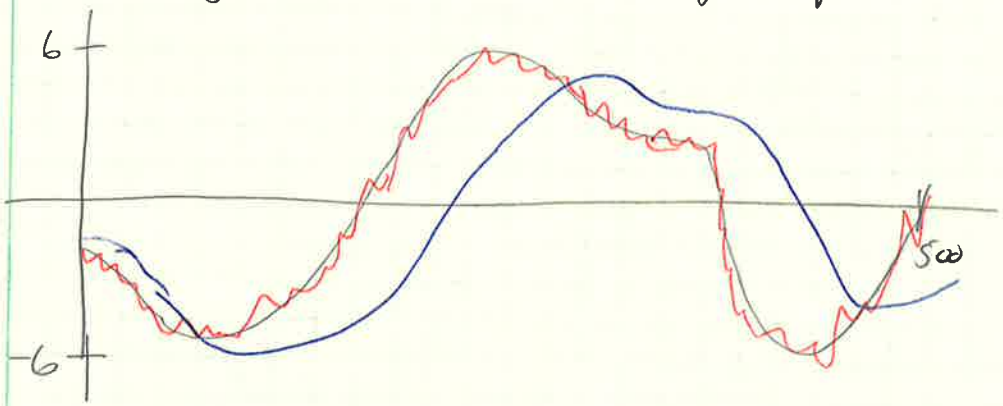
4.2.6 The leaky integrator

- From the MA to a first-order recursion

$$y_M[n] = \frac{M-1}{M} y_{M-1}[n-1] + \frac{1}{M} x[n]$$

$$y_M[n] = \lambda y_{M-1}[n-1] + (1-\lambda) x[n], \quad \lambda = \frac{M-1}{M}$$

- The Leaky Integrator
  - When M is large, $y_{M-1}[n] \approx y_M[n]$ (and $\lambda \approx 1$)
  - try the filter
    $$y[n] = \lambda y[n-1] + (1-\lambda) x[n]$$
  - filter is now recursive, since it uses its previous output value
- Denoising recursively with the Leaky Integrator



- $\lambda = 0.2$
- $\lambda = 0.98$

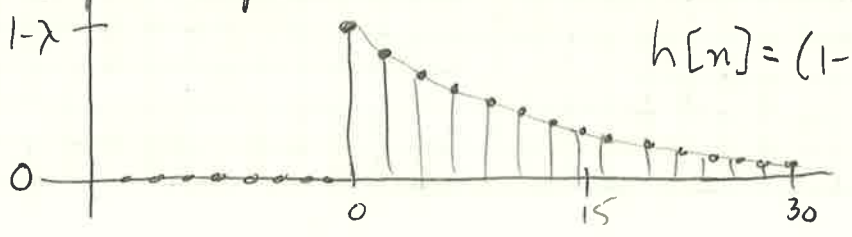
smoothness increases while delay increases

- What about the impulse response?

$$y[n] = \lambda y[n-1] + (1-\lambda)\delta[n]$$

- $y[n] = 0, \ \forall n < 0$
- $y[0] = \lambda y[-1] + (1-\lambda)\delta[0] = 1-\lambda$
- $y[1] = \lambda y[0] + (1-\lambda)\delta[1] = \lambda(1-\lambda)$
- $y[2] = \lambda y[1] + (1-\lambda)\delta[2] = \lambda^2(1-\lambda)$
- $\vdots$

- Impulse Response



$$h[n] = (1-\lambda)\lambda^n u[n]$$

- Leaky Integrator : why the name?

Discrete-time integrator is a boundless accumulator:

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

We can rewrite the integrator as

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

To prevent "explosion", pick $\lambda < 1$

$$y[n] = \lambda y[n-1] + (1-\lambda)\, x[n]$$

keep only a fraction λ of the accumulated value so far and forget ("leak") a fraction $1-\lambda$

add only a fraction $1-\lambda$ of the current value to the accumulator

Summary of Lesson 4.2

In this lesson we have studied two examples of LTI filters: the moving average and the leaky integrator. The moving average is just a local average computed over the last M observations, including the current one. We have seen the formula for its impulse response and derived a simple recursive formula to compute it efficiently.

The leaky integrator is a special case of the moving average filter when M becomes large. It is also defined by a recursive formula. The basic idea is to add a portion λ of the past accumulated value so far and a fraction $1-\lambda$ of the current observation. By setting the value of $\lambda < 1$, we ensure that the system never blows up.

4.3 Filter Stability

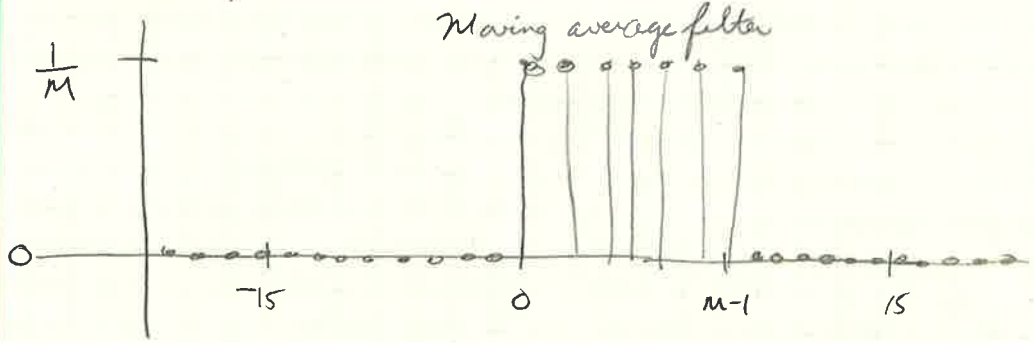
4.3.a Filter classification in the time domain

- Filter types according to impulse response
  - Finite Impulse Response (FIR)
  - Infinite Impulse Response (IIR)
    - causal
    - non causal

- FIR

  - impulse response has finite support
  - only a finite number of samples are involved in the computation of each output sample

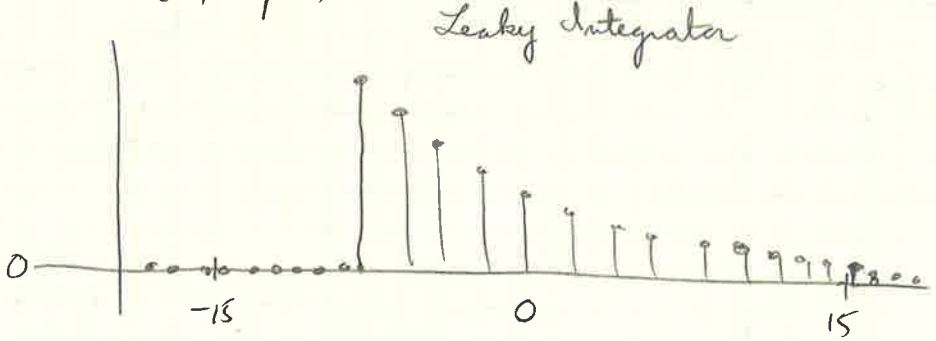
- FIR (example)



- IIR

  - impulse response has infinite support
  - a potentially infinite number of samples are involved in the computation of each output sample
  - surprisingly, in many cases the computation can still be performed in a finite amount of steps.

- IIR (example)

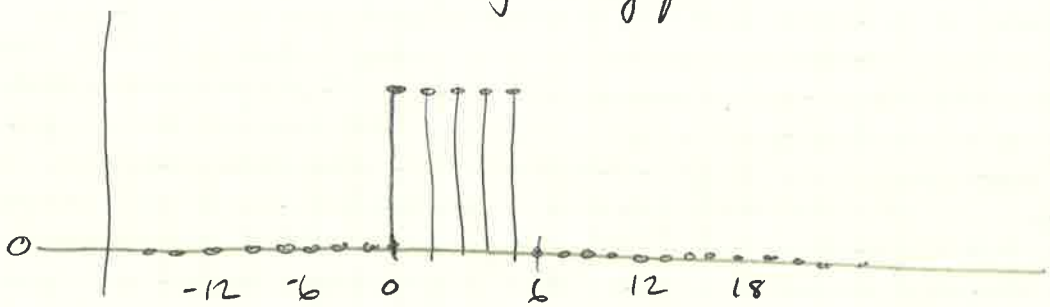


- Causal vs Noncausal

  - causal:
    - impulse response is zero for $n < 0$
    - only past samples (with respect to the present) are involved in the computation of each output sample
    - causal filters can work "on line" since they only need the past
  - non causal:
    - impulse response is nonzero for some (or all) $n < 0$
    - can still be implemented in an offline fashion (when all input data are available on storage, e.g., in Image Processing)

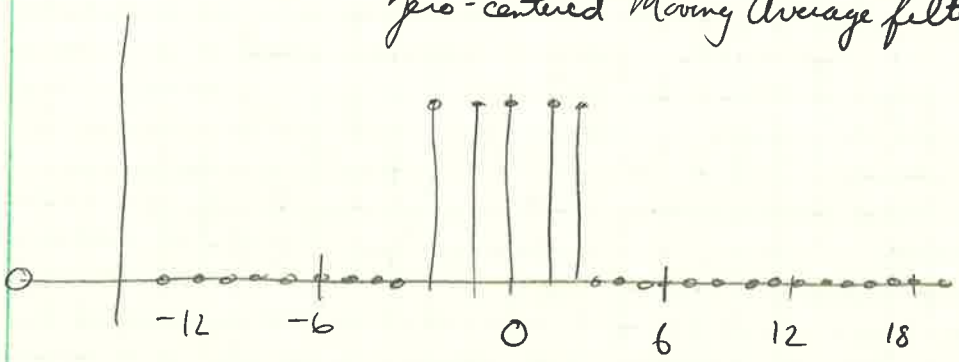
- Causal example

Moving Average filter



- Noncausal example

Zero-centered Moving Average filter



4.3.6 Filter Stability

- Stability
  - key concept: avoid "explosions" if the input is nice
  - a nice signal is a bounded signal: $|x[n]| < M, \forall n$
  - Bounded-Input-Bounded-Output (BIBO) stability: if the input is nice the output should be nice

- Fundamental Stability Theorem

A filter is BIBO stable $\Leftrightarrow$ its impulse response is absolutely summable

proof ($\Rightarrow$) Hypotheses: $|x[n]| < M$, $\sum_n h[n] = L < \infty$
Thesis: $|y[n]|$ bounded

proof: $$|y[n]| = \left| \sum_{k=-\infty}^{\infty} h[k]\, x[n-k] \right| \leq \sum_{k=-\infty}^{\infty} |h[k]\, x[n-k]|$$

$$\leq M \sum_{k=-\infty}^{\infty} |h[k]| < ML$$

proof (⇐): Hypotheses: $|x[n]| < M$, $|y[n]| < P$

Thesis: $h[n]$ is absolutely summable

proof by contradiction:

- assume $\sum_n |h[n]| = \infty$
- build $x[n] = \begin{cases} +1, & h[-n] \geq 0 \\ -1, & h[n] < 0 \end{cases}$
- clearly, $x[n]$ is bounded
- however

$$y[0] = (x*h)[0] = \sum_{k=-\infty}^{\infty} h[k]\,x[-k] = \sum_{k=-\infty}^{\infty} |h[k]| = \infty$$

- The good news: FIR filters are always stable
- Checking the stability of IIRs
  - Let's check the Leaky Integrator:

$$\sum_{n=-\infty}^{\infty} |h[n]| = |1-\lambda| \sum_{n=0}^{\infty} |\lambda|^n$$

$$= \lim_{n\to\infty} |1-\lambda| \frac{1-|\lambda|^{n+1}}{1-|\lambda|} < \infty \quad \text{for } |\lambda| < 1$$

  - Stability is guaranteed for $|\lambda| < 1$
  - We will study indirect methods for filter stability later in this Module.

4.4 Frequency Response

4.4.a The convolution theorem

- A remarkable result

$$e^{j\omega_0 n} \longrightarrow \boxed{H} \longrightarrow ?$$

$$y[n] = e^{j\omega_0 n} * h[n] = h[n] * e^{j\omega_0 n}$$

$$= \sum_{k=-\infty}^{\infty} h[k]\, e^{j\omega_0 (n-k)}$$

$$= e^{j\omega_0 n} \sum_{k=-\infty}^{\infty} h[k] e^{-j\omega_0 k}$$

$$= H(e^{j\omega_0})\, e^{j\omega_0 n}$$

$$e^{j\omega_0 n} \longrightarrow \boxed{H} \longrightarrow H(e^{j\omega_0}) e^{j\omega_0 n}$$

- complex exponentials are eigensequences of LTI systems, i.e., linear filters cannot change the frequency of sinusoids
- DTFT of impulse response determines the frequency character of a filter

- Magnitude and phase

If $H(e^{j\omega_0}) = Ae^{j\theta}$, then $H\{e^{j\omega_0 n}\} = Ae^{j(\omega_0 n + \theta)}$, $A \in \mathbb{R}$, $\theta \in [-\pi, \pi]$

amplitude: amplification ($A>1$) or attenuation ($0 \le A < 1$)

phase shift: delay ($\theta < 0$) or advancement ($\theta > 0$)

- The convolution theorem

- In general: $\text{DTFT}\{x[n] * h[n]\} = ?$
- Intuition: the DTFT reconstruction formula tells us how to build $x[n]$ from a set of complex exponential "basis" functions of the form $X(e^{j\omega})e^{j\omega n}$.

$$X(e^{j\omega})e^{j\omega n} \longrightarrow \boxed{H} \longrightarrow X(e^{j\omega})H(e^{j\omega})e^{j\omega n}$$

$$\text{DTFT}\{x[n] * h[n]\} = \sum_{n=-\infty}^{\infty} (x*h)[n]\, e^{-j\omega n}$$

$$= \sum_{n=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} x[k]\, h[n-k]\, e^{-j\omega n}$$

$$= \sum_{n=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} x[k]\, h[n-k]\, e^{-j\omega(n-k)}\, e^{-j\omega k}$$

$$= \sum_{k=-\infty}^{\infty} x[k] e^{-j\omega k} \sum_{n=-\infty}^{\infty} h[n-k] e^{-j\omega(n-k)}$$

$$= H(e^{j\omega}) X(e^{j\omega})$$

- Frequency response

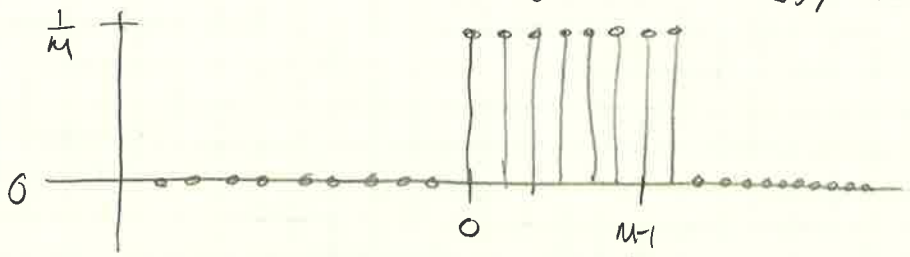
$$H(e^{j\omega}) = \text{DTFT}\{h[n]\}$$

- Two effects:
  - magnitude: amplification ($|H(e^{j\omega})| > 1$) or attenuation ($|H(e^{j\omega})| < 1$) of input frequencies
  - phase: overall delay and shape changes

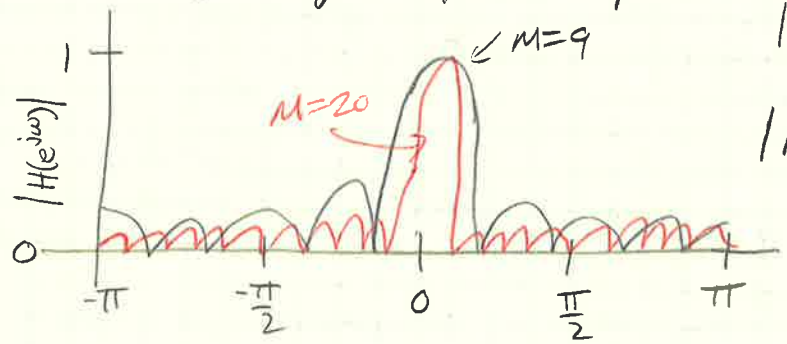
4.4.b Examples of frequency response

- Moving Average revisited

$$h[n] = (u[n] - u[n-M])/M$$



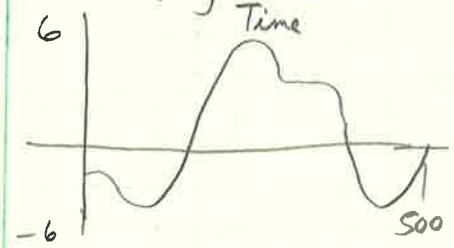
- Moving Average, magnitude response



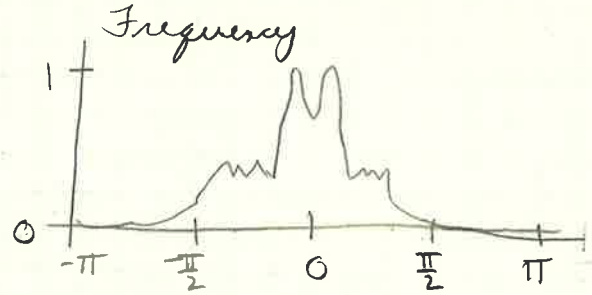
$$|H(e^{j\omega})| = \frac{1}{M}\left|\frac{\sin(\frac{\omega}{2}M)}{\sin(\frac{\omega}{2})}\right|$$

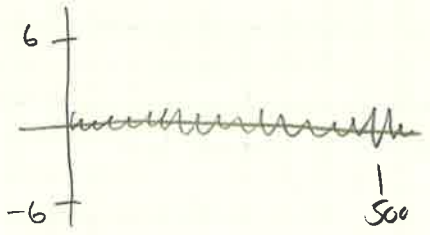
$$|H(e^{j\omega})| = 0, \quad \omega = \frac{2\pi}{M}k, \quad k \neq 0$$

- Denoising revisited

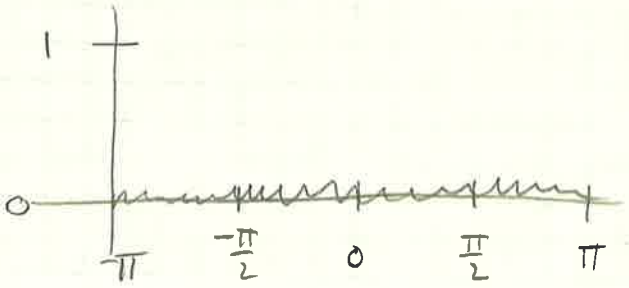


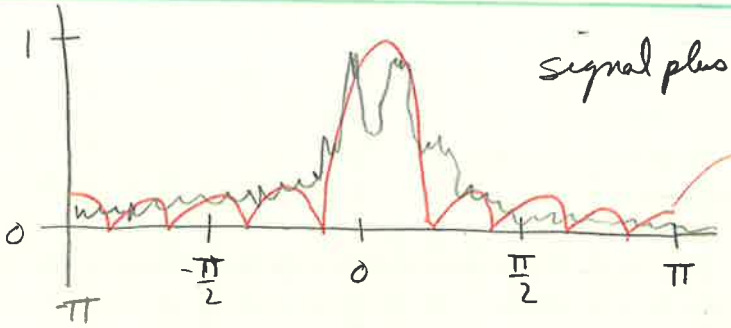
Signal



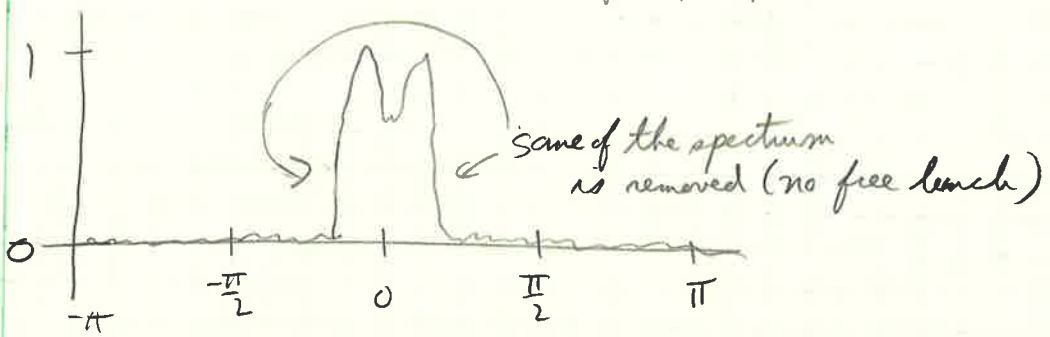


noise



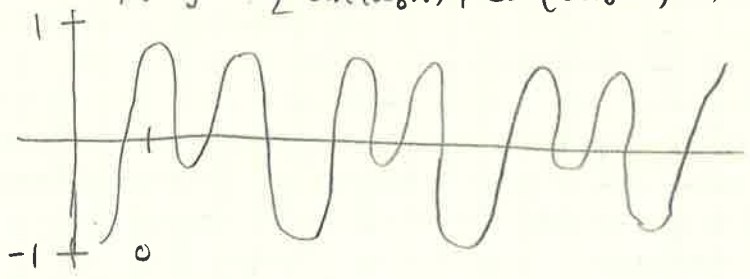


⇓ product of signal plus noise and 9-point MA filter

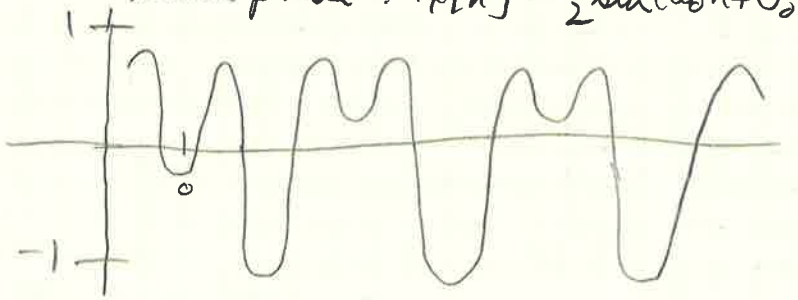


- What about the phase?
  - Assume $|H(e^{j\omega})| = 1$
    - zero phase: $\angle H(e^{j\omega}) = 0$ (spectrum is real)
    - linear phase: $\angle H(e^{j\omega}) = d\omega$, $d \in \mathbb{R}$
    - nonlinear phase
- Phase and signal shape

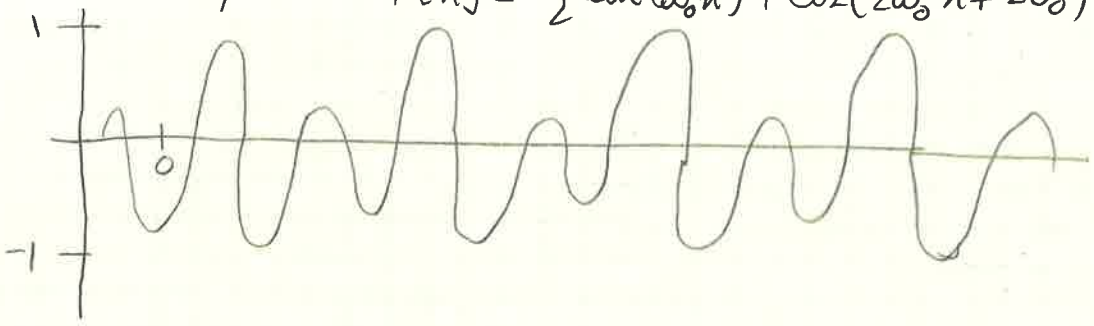
$x[n] = \frac{1}{2}\sin(\omega_0 n) + \cos(2\omega_0 n)$, $\omega_0 = \frac{2\pi}{40}$ (0-phase signal)



- linear phase: $x[n] = \frac{1}{2}\sin(\omega_0 n + \theta_0) + \cos(2\omega_0 n + 2\theta_0)$, $\theta_0 = \frac{\delta\pi}{5}$



- nonlinear phase: $x[n] = \frac{1}{2}\sin(\omega_0 n) + \cos(2\omega_0 n + 2\theta_0)$



- Linear phase

$x[n] \rightarrow \boxed{D} \longrightarrow x[n-d]$

- $y[n] = x[n-d]$
- $Y(e^{j\omega}) = e^{-j\omega d} X(e^{j\omega})$
- $H(e^{j\omega}) = e^{-j\omega d}$
- linear phase term

In general, if $H(e^{j\omega}) = A(e^{j\omega}) e^{-j\omega d}$, $A(e^{j\omega}) \in \mathbb{R}$

$x[n] \rightarrow \boxed{A(e^{j\omega})} \rightarrow \boxed{\text{delay by } d} \longrightarrow y[n]$

- Moving average is linear phase

$$H(e^{j\omega}) = \frac{1}{M} \frac{\sin\left(\frac{\omega}{2}M\right)}{\sin\left(\frac{\omega}{2}\right)} e^{-j\frac{M-1}{2}\omega}$$

$\frac{M-1}{2}$ ↖ $d = \frac{M-1}{2}$

- Leaky Integrator revisited: $h[n] = (1-\lambda)\lambda^n u[n]$

$$H(e^{j\omega}) = \frac{1-\lambda}{1-\lambda e^{j\omega}}$$

Finding magnitude and phase requires a little algebra...

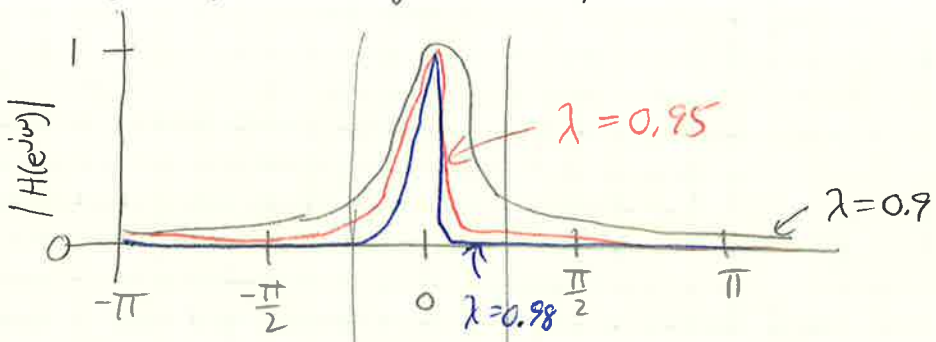
Recall from complex algebra: $\frac{1}{a+jb} = \frac{a-jb}{a^2+b^2}$

So that if $x = \frac{1}{a+jb}$, $|x|^2 = \frac{1}{a^2+b^2}$, $\angle x = \tan^{-1}\left[\frac{-b}{a}\right]$

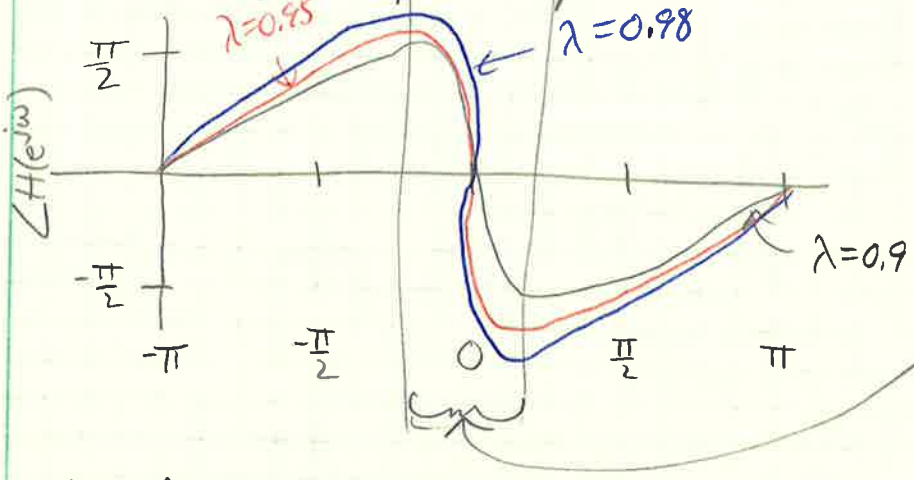
$$H(e^{j\omega}) = \frac{1-\lambda}{(1-\lambda\cos\omega) - j\sin\omega}$$ So that:

$$|H(e^{j\omega})|^2 = \frac{(1-\lambda)^2}{1-2\lambda\cos\omega+\lambda^2}, \quad \angle H(e^{j\omega}) = \tan^{-1}\left[\frac{\lambda\sin\omega}{1-\lambda\cos\omega}\right]$$

- Leaky Integrator, magnitude response

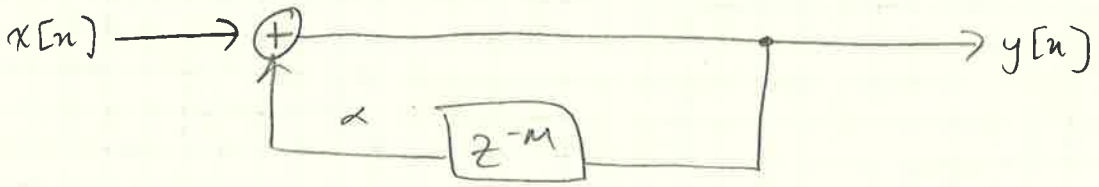


- Leaky Integrator, phase response



Phase is sufficiently linear where it matters!

Karplus-Strong revisited, again!



$$y[n] = \alpha y[n-M] + x[n]$$

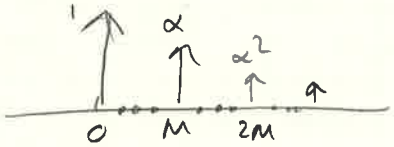
$$y[n] = \underbrace{\bar{x}[0], \bar{x}[1], \ldots, \bar{x}[M-1]}_{1^{st}\text{ period}}, \underbrace{\alpha\bar{x}[0], \alpha\bar{x}[1], \ldots, \alpha\bar{x}[M-1]}_{2^{nd}\text{ period}}, \underbrace{\alpha^2\bar{x}[0], \alpha^2\bar{x}[1], \ldots}_{\ldots}$$

- DTFT of KS signal, using the convolution theorem

Key observation:

$$y[n] = \bar{x}[n] * w[n], \quad w[n] = \begin{cases} \alpha^k, & n = kM \\ 0, & \text{otherwise} \end{cases}$$

$$Y(e^{j\omega}) = \bar{X}(e^{j\omega})\, W(e^{j\omega})$$

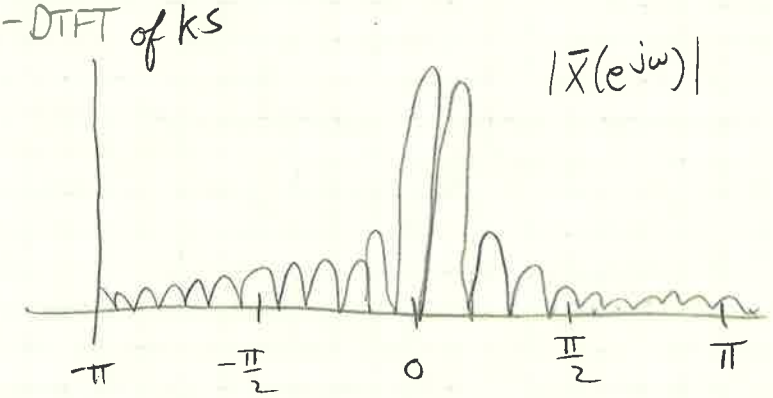


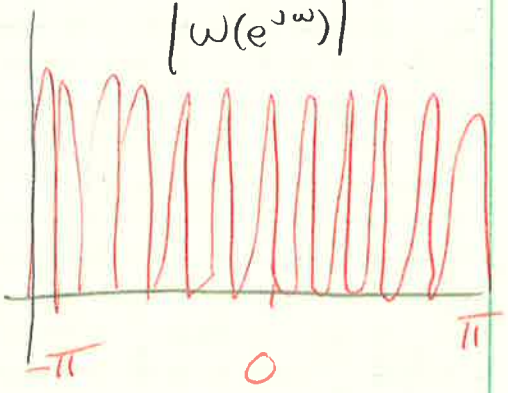
$$\bar{X}(e^{j\omega}) = e^{-j\omega}\left(\frac{M+1}{M-1}\right)\frac{1-e^{-j(M-1)\omega}}{(1-e^{-j\omega})^2} - \frac{1-e^{-j(M+1)\omega}}{(1-e^{-j\omega})^2}$$

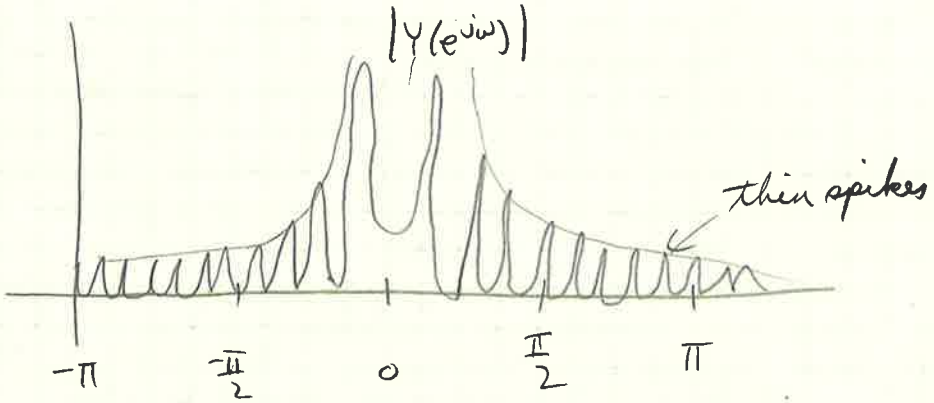
(sawtooth wave)

$$W(e^{j\omega}) = \frac{1}{1-\alpha e^{-j\omega M}}$$

- DTFT of ks







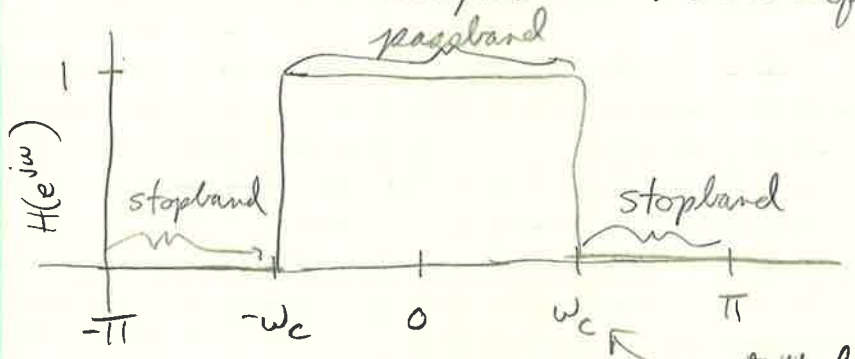
4.5 Ideal Filters

4.5.a Filter classification in the frequency domain

- Filter types according to magnitude response
  - Lowpass
  - Highpass
  - Bandpass
  - Allpass
    - Moving average and Leaky Integrator are lowpass filters
- Filter types according to phase response
  - Linear phase
  - Nonlinear phase

4.5.b The ideal lowpass filter

- What is the best lowpass we can think of?



$H(e^{j\omega}) \in \mathbb{R}$
$\Rightarrow$ no phase

$\omega_b = 2\omega_c$ (bandwidth)

- Ideal lowpass filter

$$H(e^{j\omega}) = \begin{cases} 1, & |\omega| \le \omega_c \\ 0, & \text{otherwise} \end{cases} \quad (2\pi\text{-periodicity implicit})$$

- perfectly flat passband
- infinite attenuation in stopband
- zero-phase (no delay)

- Ideal lowpass filter: impulse response

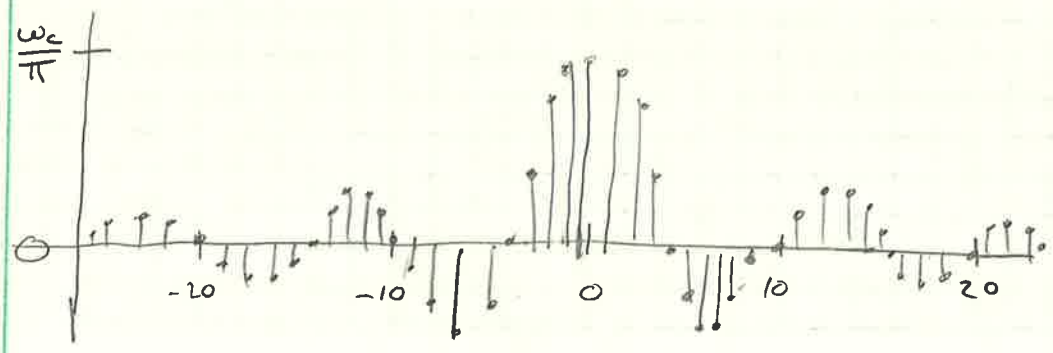
$$h[n] = \text{IDTFT}\{H(e^{j\omega})\}$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega}) e^{j\omega n}\, d\omega$$

$$= \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{j\omega n}\, d\omega$$

$$= \frac{1}{\pi n}\,\frac{e^{j\omega_c n} - e^{-j\omega_c n}}{2j}$$

$$= \frac{\sin \omega_c n}{\pi n}$$



- The bad news
  - impulse response has infinite support, two-sided
    ⇒ cannot compute the output in a finite amount of time
    ⇒ that's why it's called "ideal"
  - impulse response decays slowly in time, $O(\frac{1}{n})$
    ⇒ we need a lot of samples for a good approximation

- Nevertheless ...

The sinc-rect pair:

$$\text{rect}(x) = \begin{cases} 1, & |x| \le \frac{1}{2} \\ 0, & |x| > \frac{1}{2} \end{cases}$$

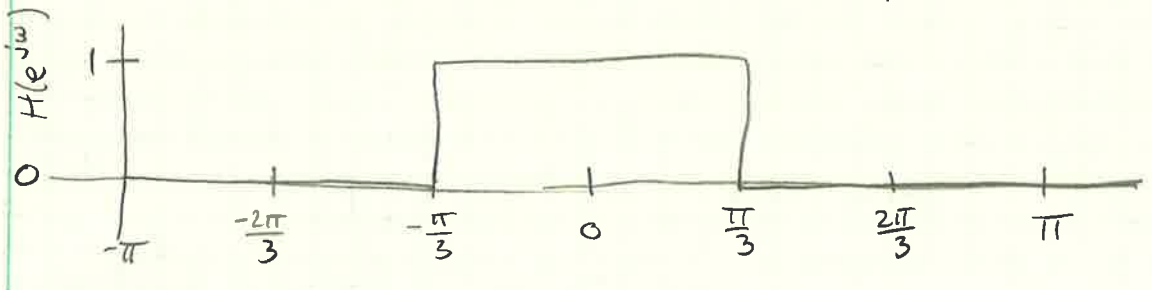
$$\text{sinc}(x) = \begin{cases} \frac{\sin(\pi x)}{\pi x}, & x \neq 0 \\ 1, & x = 0 \end{cases}$$

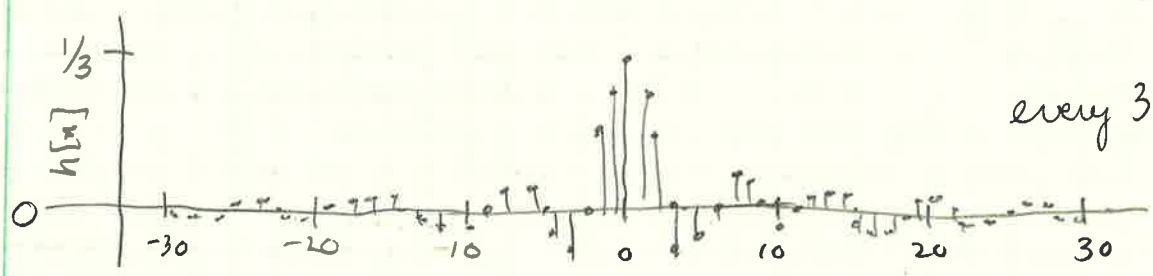
(note that $\text{sinc}(x) = 0$ when x is a nonzero integer)

- The ideal lowpass in canonical form

$$\underbrace{\text{rect}\left(\frac{\omega}{\omega_c}\right)}_{H(e^{j\omega})} \overset{\text{DTFT}}{\longleftrightarrow} \underbrace{\frac{\omega_c}{\pi}\,\text{sinc}\left(\frac{\omega_c}{\pi} n\right)}_{\text{Impulse response}}$$

- Example $\omega_c = \frac{\pi}{3}$: $H(e^{j\omega}) = \text{rect}\left(\frac{3\omega}{2\pi}\right)$, $h[n] = \frac{1}{3}\,\text{sinc}\frac{n}{3}$





every 3rd sample = 0

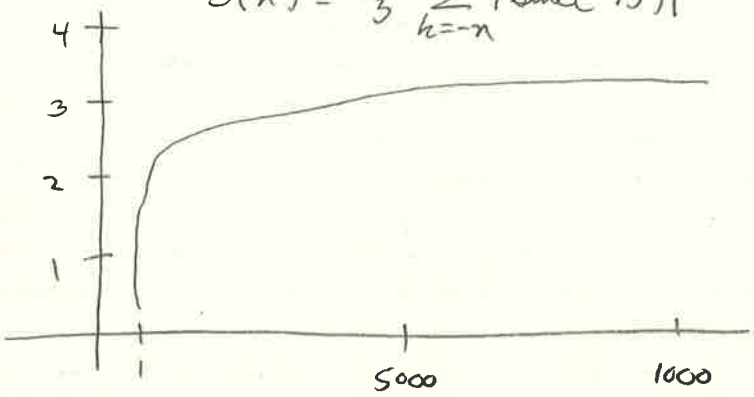
- Little-known fact

  - the sinc is not absolutely summable
  - the ideal lowpass is not BIBO stable!
  - example for $\omega_c = \frac{\pi}{3}$: $h[n] = \frac{1}{3}\,\mathrm{sinc}\,\frac{n}{3}$
  - take $x[n] = \mathrm{sign}\left\{\mathrm{sinc}\left(\frac{-n}{3}\right)\right\}$ and

$$y[0] = (x*h)[0] = \frac{1}{3}\sum_{k=-\infty}^{\infty} \left|\mathrm{sinc}\left(\frac{k}{3}\right)\right| = \infty$$

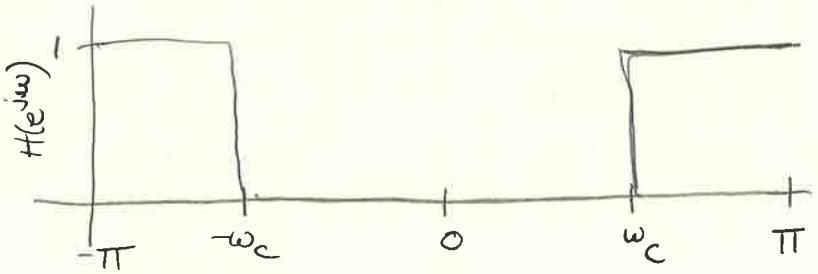
- Divergence, however, is very slow...

$$S(n) = \frac{1}{3}\sum_{k=-n}^{n} \left|\mathrm{sinc}(k/3)\right|$$



4.5.c Ideal filters derived from the ideal lowpass filter

- Ideal highpass filter

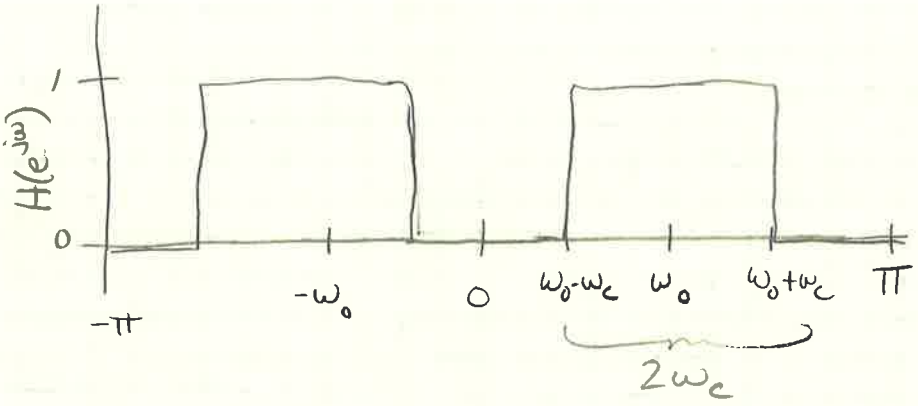


$$H_{hp}(e^{j\omega}) = \begin{cases} 1, & \pi \geq |\omega| \geq \omega_c \\ 0, & \text{otherwise} \end{cases} \quad (2\pi\text{-periodicity implicit})$$

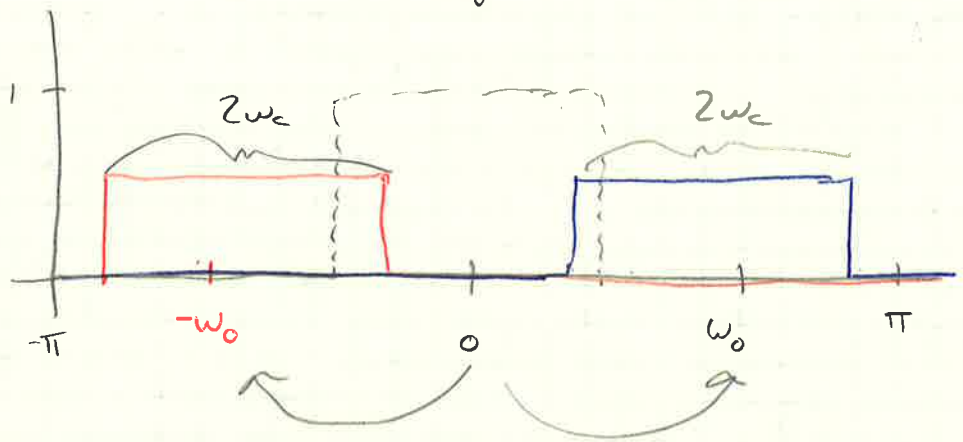
$$H_{hp}(e^{j\omega}) = 1 - H_{lp}(e^{j\omega})$$

$$h_{hp}[n] = \delta[n] - \frac{\omega_c}{\pi}\,\mathrm{sinc}\left(\frac{\omega_c}{\pi}n\right) \qquad \text{impulse response in time domain}$$

- Ideal bandpass filter



- can be obtained by modulating the ideal lowpass filter



2 copies from cosine modulation

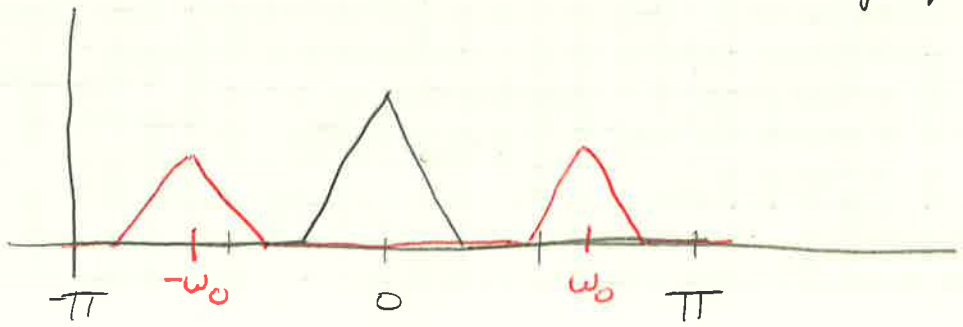
$$H_{bp}(e^{j\omega}) = \begin{cases} 1, & |\omega \pm \omega_0| \le \omega_c \\ 0, & \text{otherwise} \end{cases} \quad (2\pi\text{-periodicity implicit})$$

$$h_{bp}[n] = \underset{\uparrow}{2}\cos(\omega_0 n)\frac{\omega_c}{\pi}\,\mathrm{sinc}\left(\frac{\omega_c}{\pi}n\right) \quad \text{(impulse response)}$$

↑ for unit amplitude

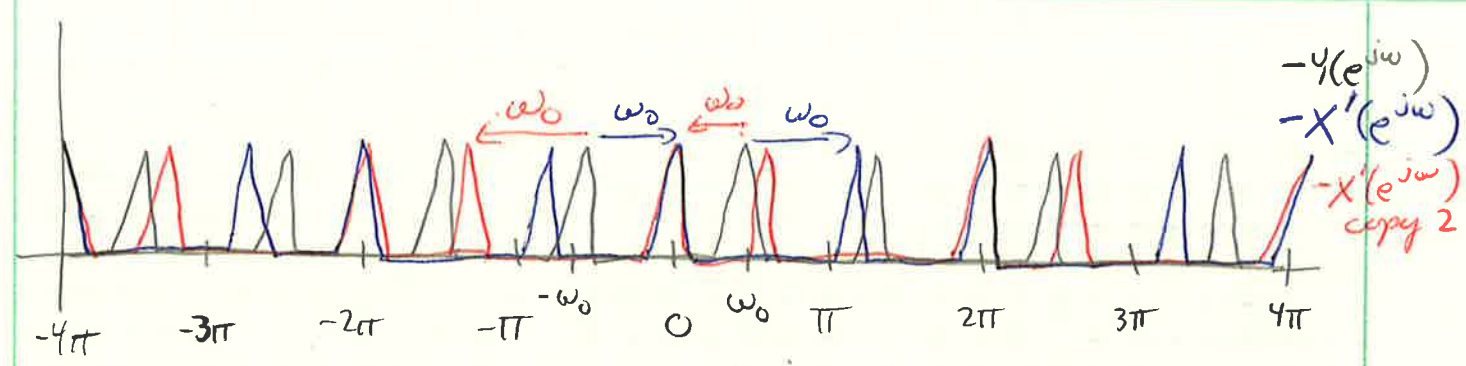
4.5.d Demodulation revisited

- apply sinusoidal modulation to $x[n]$: $y[n] = x[n]\cos\omega_0 n$
- demodulate by multiplying by the carrier $x'[n] = y[n]\cos\omega_0 n$
- demodulated signal contains unwanted high-frequency components

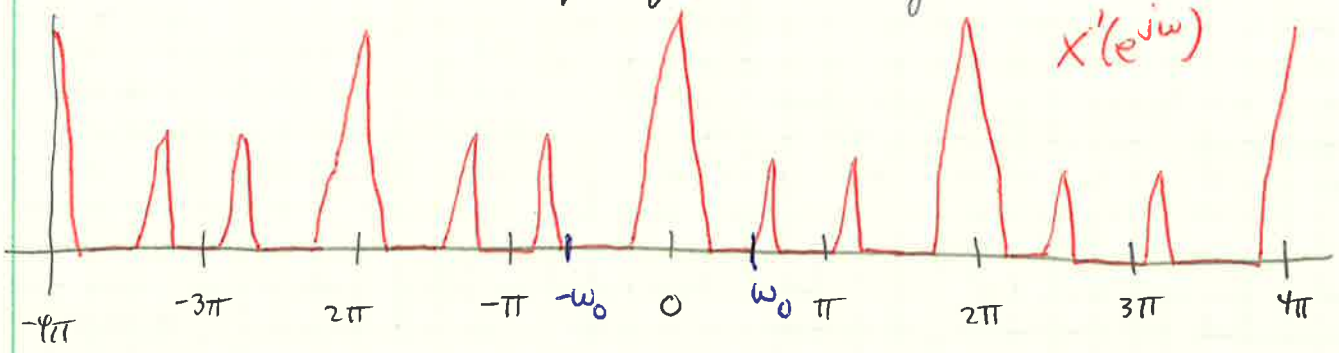


$-X(e^{j\omega})$

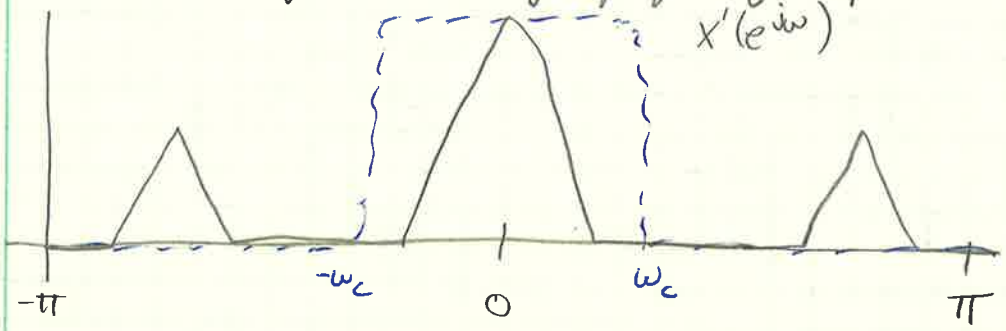
$-Y(e^{j\omega})$



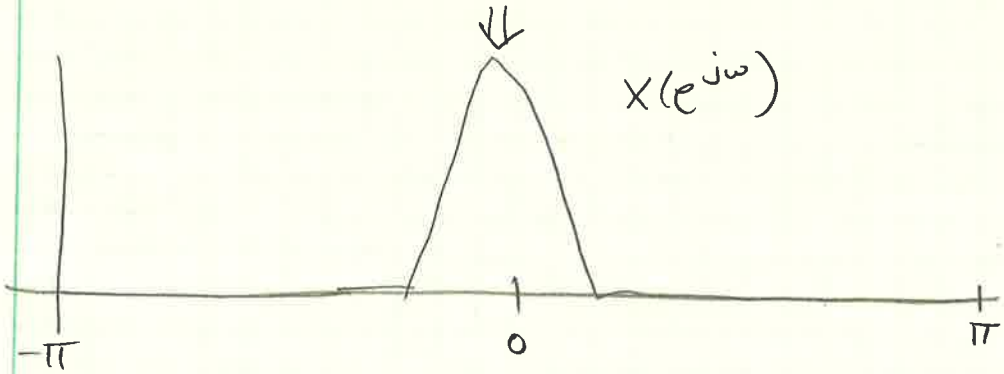
add copies of $X'(e^{j\omega})$ together



To get rid of spurious high frequency components, use lowpass filtering



- lowpass filter



Summary to Lesson 4.5

We have started this lesson with a filter classification based on characteristics of filters in the frequency domain. Hence, we distinguish lowpass (resp. highpass) filters, i.e., filters that let low (resp. high) frequencies go through and kill high (resp. low) frequencies. A bandpass filter is a filter that lets a certain range of frequency go through and a allpass filter one with constant magnitude over all frequencies.

We have studied the ideal lowpass filter, in some sense the best lowpass filter we can think of. Its frequency response is characterised by

- a passband centered around zero with cutoff frequency ω_c
- an infinite attenuation in the stop band
- no phase term (aka no delay).

Its corresponding impulse response is a sinc function whose support is infinite. Thus it is not possible to implement this filter with a finite number of operations. This is the reason why we call this filter ideal and we will spend the following lessons to derive good approximations. Furthermore, from the ideal lowpass filters, we have also derived ideal highpass and bandpass filters.

We have concluded this lesson by revisiting the demodulation problem. When we demodulate a signal, we multiply it by a cosine of the same carrier frequency and subsequently apply a lowpass filter to get rid off the spurious high frequency components.

✓ Complete

Module 4 Part 2

4.6 Filter Design Part 1

4.6.a Impulse truncation and Gibbs phenomenon

- How can we approximate an ideal lowpass?
  - Idea #1
    - pick ω_c
    - compute ideal impulse response $h[n]$
    - truncate $h[n]$ to a finite-support $\hat{h}[n]$
    - $\hat{h}[n]$ defines an FIR filter
  - Approximation by truncation
    - FIR approximation of length $M = 2N+1$:

$$\hat{h}[n] = \begin{cases} \frac{\omega_c}{\pi} \operatorname{sinc}\left(\frac{\omega_c}{\pi} n\right), & |n| \le N \\ 0, & \text{otherwise} \end{cases}$$

    - Why it could be a good idea

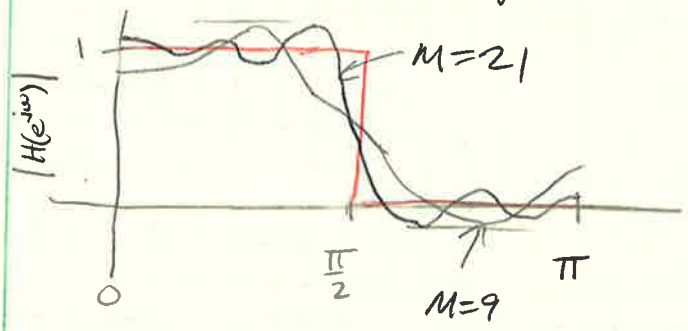
$$MSE = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| H(e^{j\omega}) - \hat{H}(e^{j\omega}) \right|^2 d\omega = \left\| H(e^{j\omega}) - \hat{H}(e^{j\omega}) \right\|^2$$

$$= \left\| h[n] - \hat{h}[n] \right\|^2 \quad \text{(Parseval)}$$

$$= \sum_{n=-\infty}^{\infty} |h[n] - \hat{h}[n]|^2$$

MSE is minimized by symmetric impulse truncation around zero

    - Why it is not such a good idea



- ideal lowpass filter ($\omega_c = \frac{\pi}{2}$)

  - The Gibbs phenomenon

The maximum error around the cutoff frequency is around 9% of the height of the jump regardless of N

4.6.6 Window method

- Understanding the Gibbs phenomenon

$$\hat{h}[n] = h[n]w[n]$$

$$w[n] = \begin{cases} 1, & |n| \le N \\ 0, & \text{otherwise} \end{cases} \qquad \hat{H}(e^{j\omega}) = ?$$

- The modulation theorem

$$\text{DTFT}\{(x*y)[n]\} = X(e^{j\omega})Y(e^{j\omega}) \quad \text{- convolution theorem}$$

$$\text{DTFT}\{x[n]\,y[n]\} = (X*Y)(e^{j\omega}) \quad \text{- modulation theorem}$$

- Convolution of DTFTs

in $\mathcal{C}^\infty$: $(x*y)[n] = \langle x^*[k], y[n-k] \rangle$

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

in $L_2([-\pi,\pi])$: $(X*Y)(e^{j\omega}) = \langle X^*(e^{j\sigma}), Y(e^{j(\omega-\sigma)}) \rangle$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})Y(e^{j(\omega-\sigma)})\,d\sigma$$

- Modulation theorem: proof

$$\text{IDTFT}\{(X*Y)(e^{j\omega})\} = \frac{1}{2\pi}\int_{-\pi}^{\pi}(X*Y)(e^{j\omega})e^{j\omega n}\,d\omega$$

$$= \frac{1}{(2\pi)^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})Y(e^{j(\omega-\sigma)})e^{j\omega n}\,d\sigma\,d\omega$$

$$= \frac{1}{(2\pi)^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})Y(e^{j(\omega-\sigma)})e^{j\sigma n}e^{j(\omega-\sigma)n}\,d\sigma\,d\omega$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})e^{j\sigma n}\,d\sigma \;\frac{1}{2\pi}\int_{-\pi}^{\pi} Y(e^{j(\omega-\sigma)n})e^{j(\omega-\sigma)n}\,d\omega$$

$$= x[n]\,y[n]$$

- Aside: sinusoidal modulation revisited

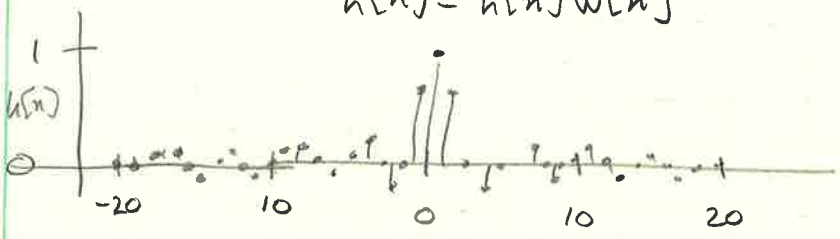
$$\text{DTFT}\{x[n]\cos\omega_c n\} = X(e^{j\omega}) * \frac{1}{2}[\tilde{\delta}(\omega-\omega_c)+\tilde{\delta}(\omega+\omega_c)]$$

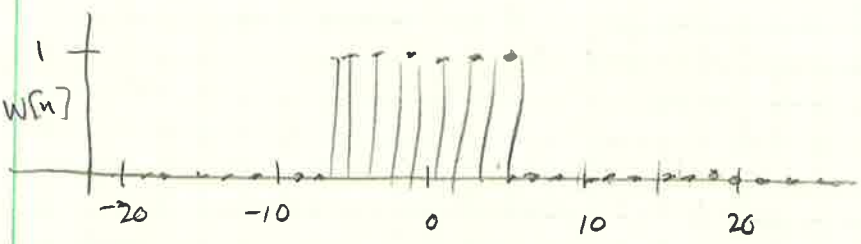
$$= \frac{1}{4\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})\tilde{\delta}(\sigma-\omega-\omega_c)\,d\sigma + \frac{1}{4\pi}\int_{-\pi}^{\pi} X(e^{j\sigma})\tilde{\delta}(\sigma-\omega-\omega_c)\,d\sigma$$

$$= \frac{1}{2}\left[X(e^{j(\omega-\omega_c)}) + X(e^{j(\omega+\omega_c)})\right]$$

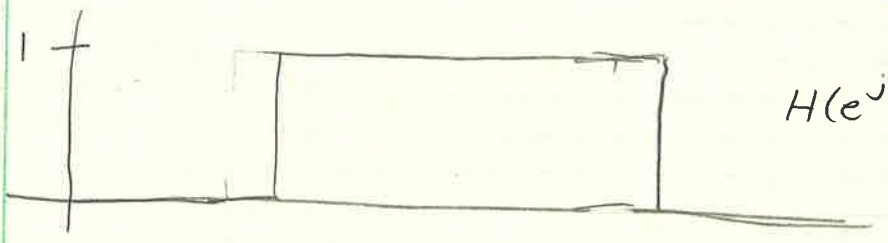
- Understanding the Gibbs phenomenon

$$\hat{h}[n] = h[n]\,w[n]$$

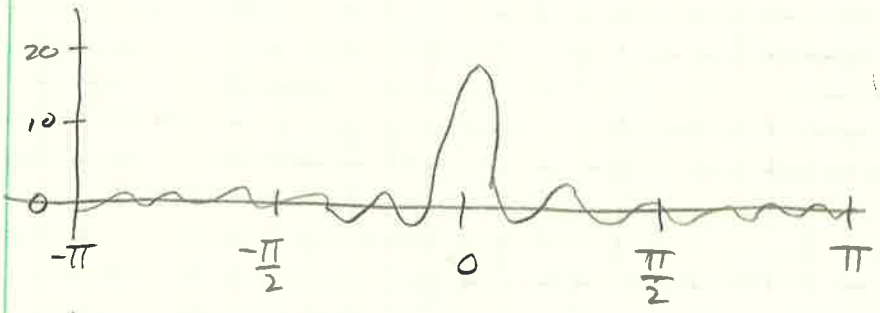




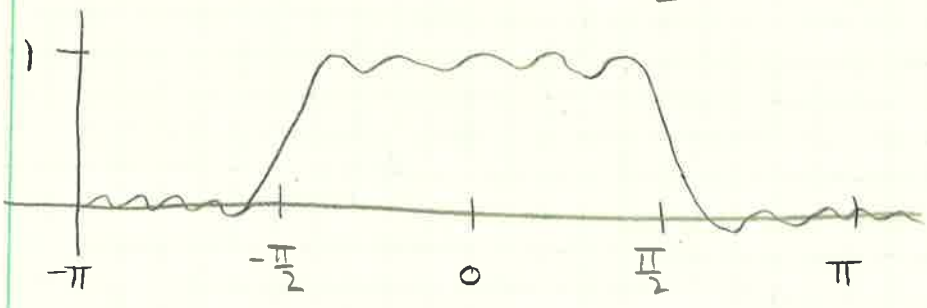
$$\hat{H}(e^{j\omega}) = (H * W)(e^{j\omega}), \quad W(e^{j\omega}) = \frac{\sin(\omega(2N+1)/2)}{\sin(\omega/2)}$$



$H(e^{j\omega})$

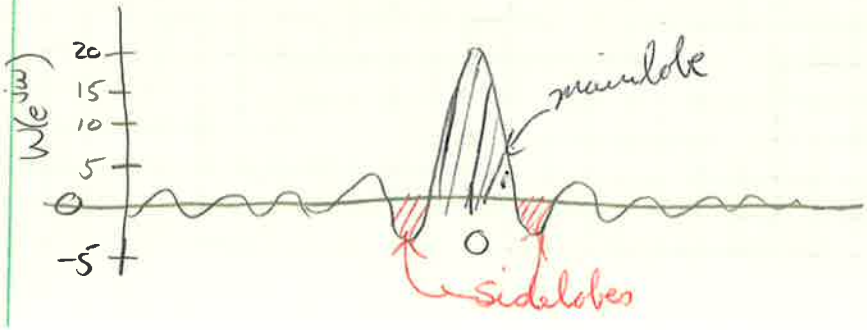


$W(e^{j\omega})$



$\hat{H}(e^{j\omega})$

- Mainlobe and sidelobes

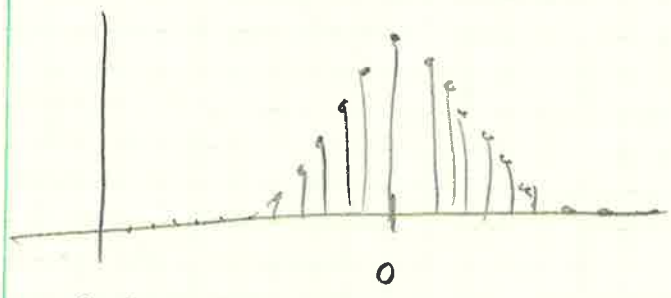


- What if we change the window?

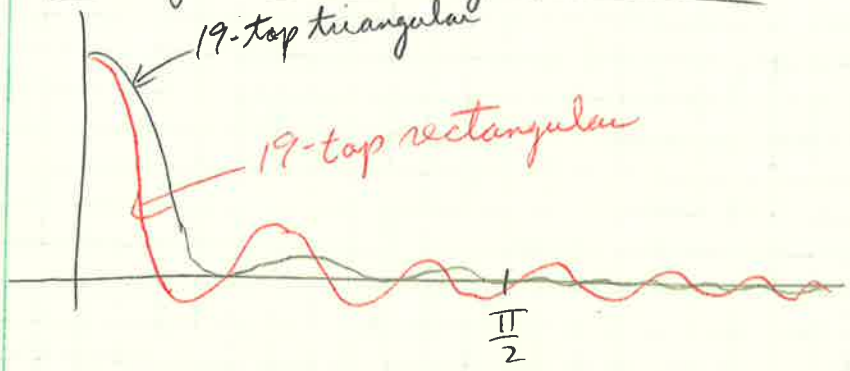
· We want:
  - narrow mainlobe so that transition is sharp
  - small sidelobe so Gibbs error is small
  - short window so FIR is efficient

} very conflicting requirements!

- Triangular Window



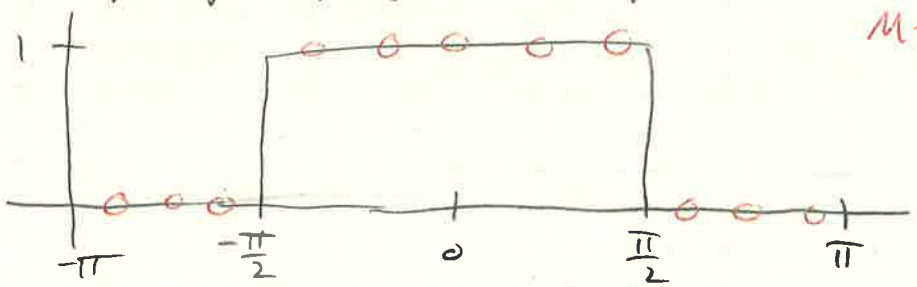
- Rectangular vs. Triangular Window



4.6.c Frequency sampling

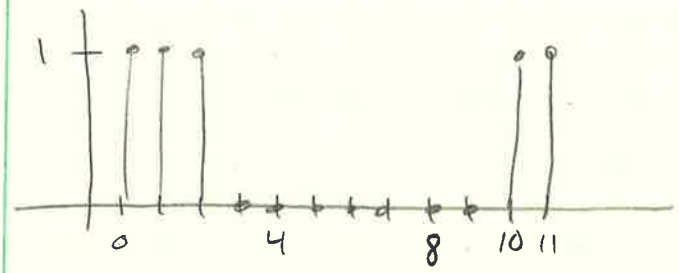
· Idea #2
  - draw desired frequency response $H(e^{j\omega})$
  - take M values at $\omega_k = \left(\frac{2\pi}{M}\right)k$
  - compute IDFT of values
  - use result as an M-tap impulse response $\hat{h}[n]$

- Frequency sampling: desired response

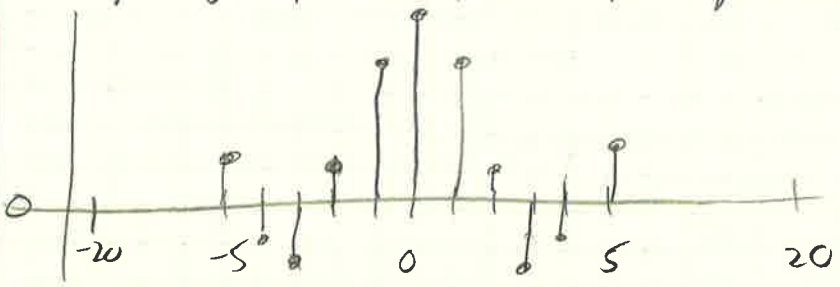


M = 11 samples

- Frequency sampling: DFT samples



- Frequency sampling: impulse response from IDFT



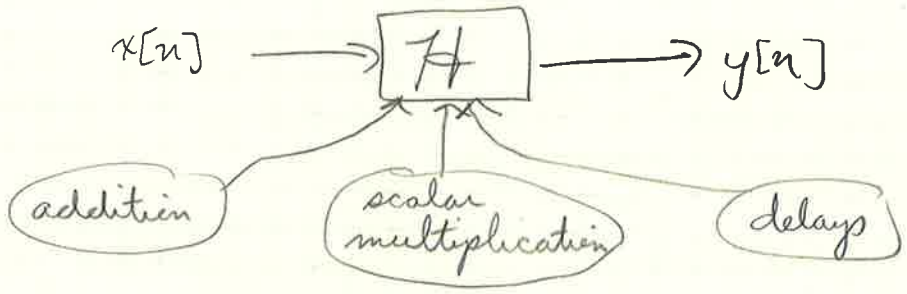
- Why it's not such a good idea
  - frequency response is DTFT of finite-support, whose DFT we know
  - frequency response is interpolation of frequency samples
  - interpolator is transform of N-tap rectangular window
  - again, no control over mainlobe and sidelobes

4.7 Realizable Filters

4.7.a The z-transform

- The most general LTI systems
  - ideal filters cannot be implemented
  - what is the most general, realizable LTI transformation?
    - linearity: only sums and multiplications
    - time-invariance: only multiplications by constants
    - realizability: only finite number of past and future samples

- Linear, time-invariant systems



- Constant-Coefficient Difference Equation (CCDE)

$$\sum_{k=0}^{N-1} a_k y[n-k] = \sum_{k=0}^{M-1} b_k x[n-k]$$

- uses M input and N output values
- how do we compute the frequency response?
- we need a new tool!

- The z-transform

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}, \quad z \in \mathbb{C}$$

- think of it as a formal operator...
- ...or as the extension of the DTFT to the whole complex plane:

$$X(z)\Big|_{z=e^{j\omega}} = \text{DTFT}\{x[n]\}$$

- Key properties

a) linearity: $\mathcal{Z}\{\alpha x[n] + \beta y[n]\} = \alpha X(z) + \beta Y(z)$

b) time shift: $\mathcal{Z}\{x[n-N]\} = z^{-N} X(z)$

- Applying the z-transform to CCDE's

$$\sum_{k=0}^{N-1} a_k y[n-k] = \sum_{k=0}^{M-1} b_k x[n-k]$$

$$Y(z) \sum_{k=0}^{N-1} a_k z^{-k} = X(z) \sum_{k=0}^{M-1} b_k z^{-k} \iff Y(z) = H(z) X(z)$$

- Rational Transfer Function

$$H(z) = \frac{\sum_{k=0}^{M-1} b_k z^{-k}}{\sum_{k=0}^{N-1} a_k z^{-k}}$$

, by setting $z = e^{j\omega}$ we have the frequency response!

(and now the notation $X(e^{j\omega})$ should make more sense)

- BTW, remember the delay block?

$x[n] \longrightarrow \boxed{z^{-1}} \longrightarrow x[n-1]$

now the notation should make more sense!

- Leaky Integrator revisited

$$y[n] = (1-\lambda)x[n] + \lambda y[n-1]$$

$$Y(z) = (1-\lambda)X[z] + \lambda z^{-1} Y(z)$$

$$H(z) = \frac{1-\lambda}{1-\lambda z^{-1}}$$

$$H(e^{j\omega}) = \frac{1-\lambda}{1-\lambda e^{-j\omega}}$$

- Region of convergence (ROC)

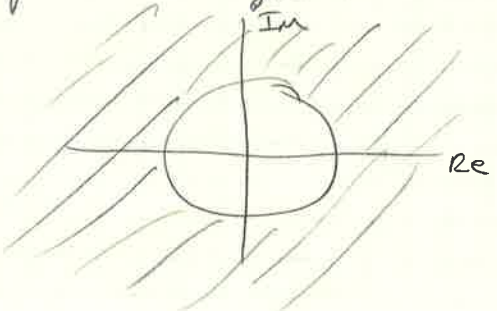
- The z-transform is a power series, so convergence is always absolute

$$|X(z)| < \infty \Leftrightarrow \sum_{n=-\infty}^{\infty} |x[n] z^{-n}| < \infty$$

- ROC is whole complex plane for finite-support sequences
- ROC has circular symmetry (depends only on $|z|$)
- ROC of causal sequences extends from a circle to infinity

4.7.b Region of convergence and stability

- ROC for causal systems



Consider the transfer function for an LTI system:

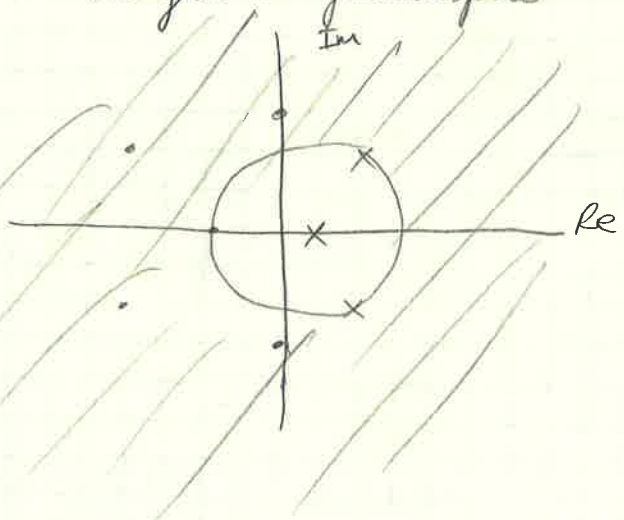
$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{M-1} z^{-(M-1)}}{a_0 + a_1 z^{-1} + \cdots + a_{N-1} z^{-(N-1)}}$$

It can always be factored as:

$$H(z) = C \frac{\prod_{n=1}^{M-1} (1 - z_n z^{-1})}{\prod_{n=1}^{N-1} (1 - p_n z^{-1})}$$

- z_n's: zeros of the transfer function
- p_n's: poles of the transfer function
- only trouble spots for ROC are the poles

- We know:
  - ROC extends outwards
  - ROC cannot include poles
    - ROC extends outward from a circle touching the largest-magnitude pole

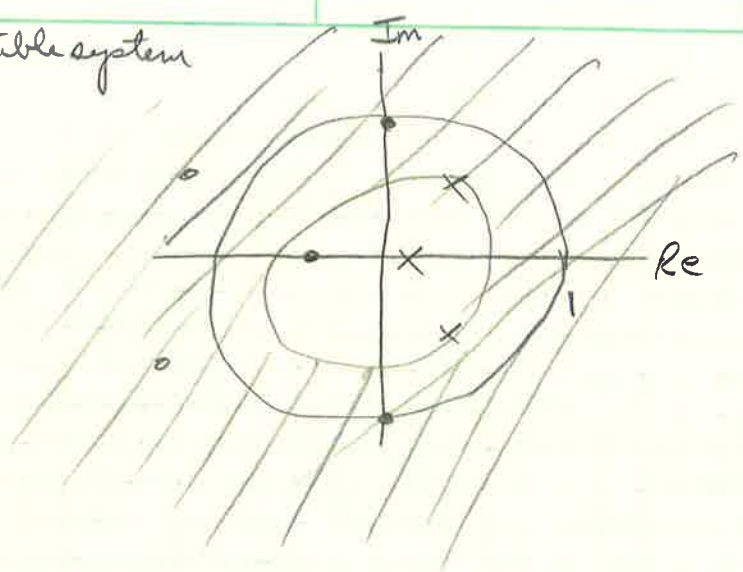


X - pole
o - zero

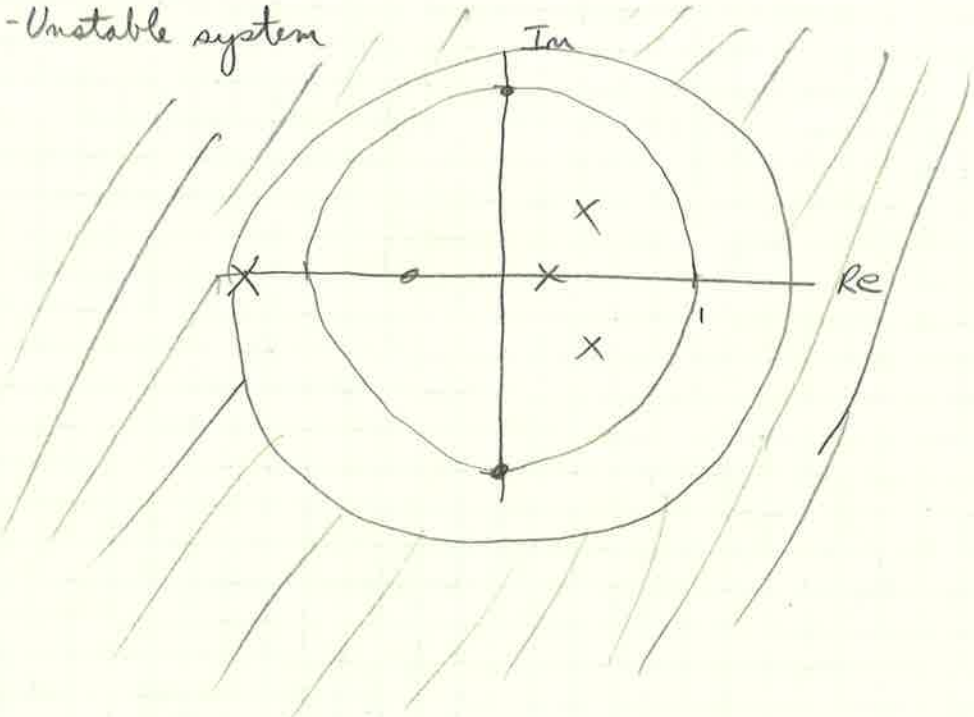
- System stability criterion

- Consider a filter with impulse response $h[n]$
  - BIBO stability $\Leftrightarrow \sum_{n=-\infty}^{\infty} |h[n]| < \infty$
  - $H(z)\big|_{|z|=1} < \infty \Leftrightarrow \sum_{n=-\infty}^{\infty} |h[n]| < \infty$ (absolute convergence of z-transform)
  - **System is stable iff ROC includes the unit circle!**

- Stable system



- Unstable system



- Estimating the frequency response from the pole-zero plot

· The "circus tent" method:

- magnitude of Z-transform is like a rubber sheet over the complex plane
- zeros glue the sheet to the ground
- poles are like... poles, pushing it up
- frequency response (in magnitude) is sheet profile around the unit circle

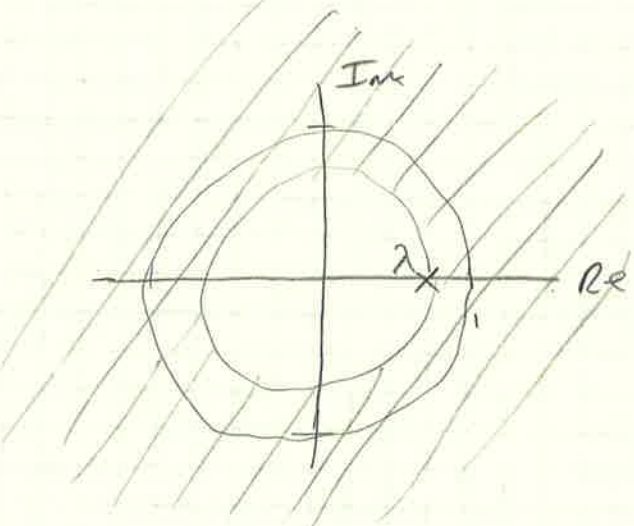
4.8 Filter Design Part 2

4.8.a Intuitive IIR designs

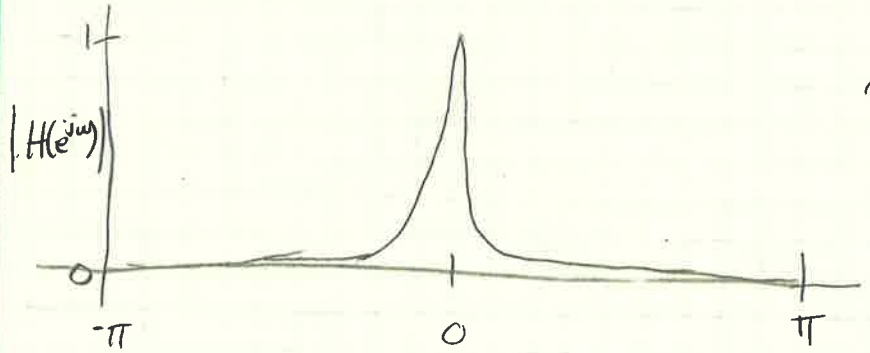
- Simple, useful filters
  - Many signal processing problems can be solved using simple filters
  - We have seen simple lowpass filters already (MA, Leaky Integrator)
  - Simple (low order) transfer functions allow for intuitive design steering
- Simple lowpass
  - Let only low frequencies pass
  - Used to remove high frequency components (e.g. noise)
  - Useful in audio, communication, control systems
  - We know a simple answer: Leaky Integrator
- Leaky Integrator

$$H(z) = \frac{1-\lambda}{1-\lambda z^{-1}}$$

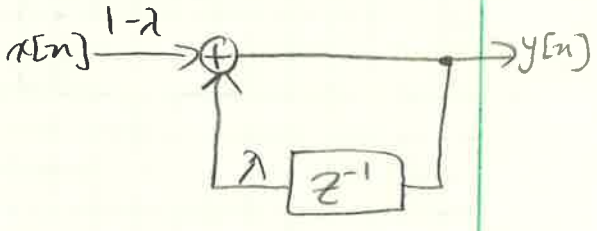
$$y[n] = (1-\lambda)x[n] + \lambda y[n-1]$$



$\lambda = 0.98$



Filter Structure

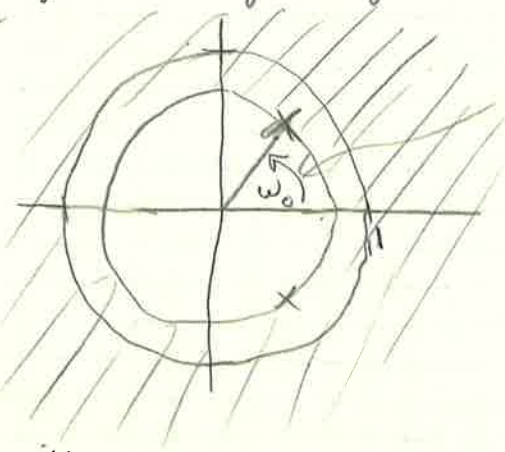


- Resonator

- A resonator is a narrow bandpass filter
- Used to detect the presence of a sinusoid of a given frequency
- Useful in communication systems and telephony (DTMF) dual-tone multi-freq
- Idea: shift the passband of the Leaky Integrator!

$$H(z) = \frac{G_0}{(1-pz^{-1})(1-p^*z^{-1})} \quad \text{(Gain factor)}$$

$$p = \lambda e^{j\omega_0}$$



move ZP pole by ω_0

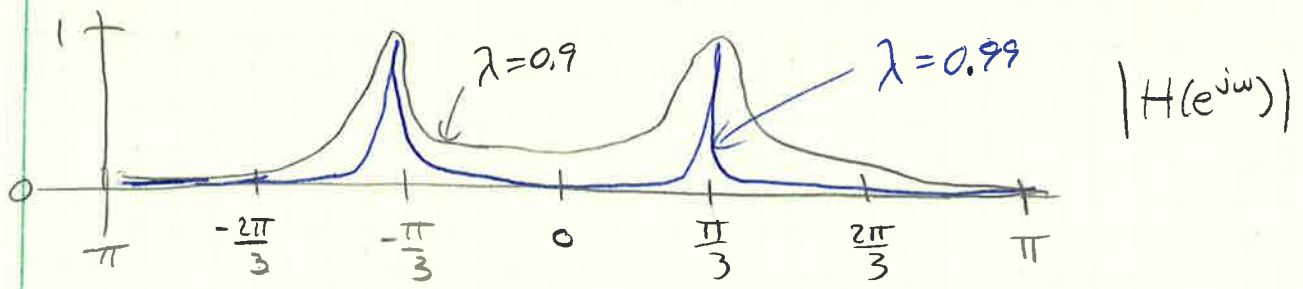
$$y[n] = G_0 x[n] - a_1 y[n-1] - a_2 y[n-2]$$

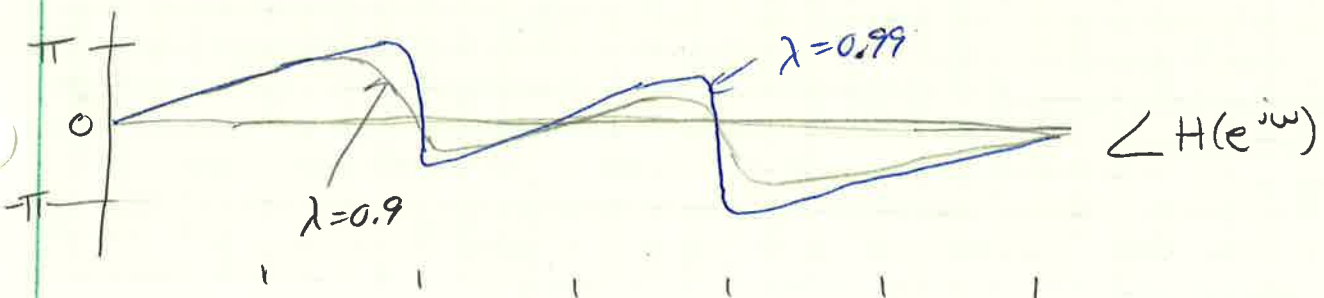
$$H(z) = \frac{G_0}{(1-pz^{-1})(1-p^*z^{-1})}, \quad p = \lambda e^{j\omega_0}$$

$$= \frac{G_0}{1 - 2\,\mathrm{Re}(p)z^{-1} + |p|^2 z^{-2}}$$

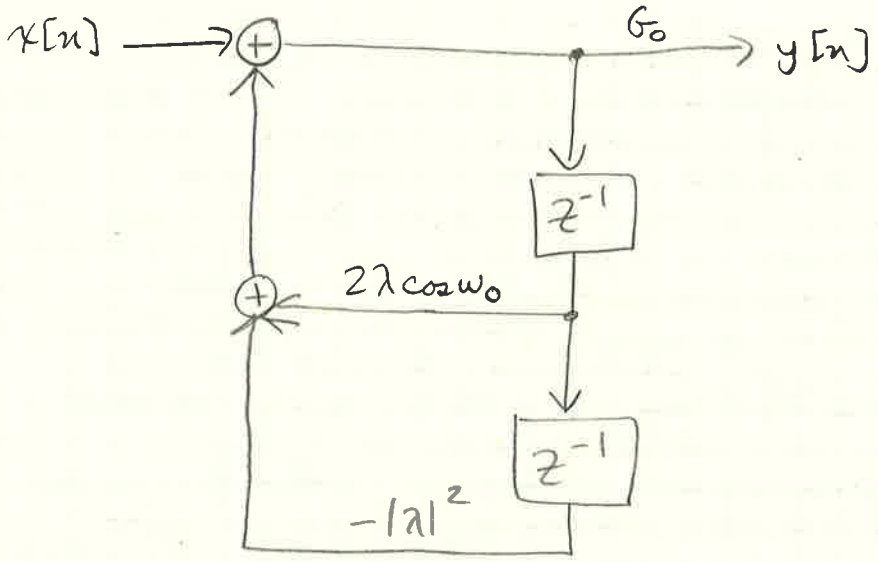
$$= \frac{G_0}{1 - 2\lambda\cos\omega_0\, z^{-1} + |\lambda|^2 z^{-2}} \Rightarrow \begin{aligned} a_1 &= 2\lambda\cos\omega_0 \\ a_2 &= -|\lambda|^2 \end{aligned}$$

$\omega_0 = \frac{\pi}{3}$





filter structure

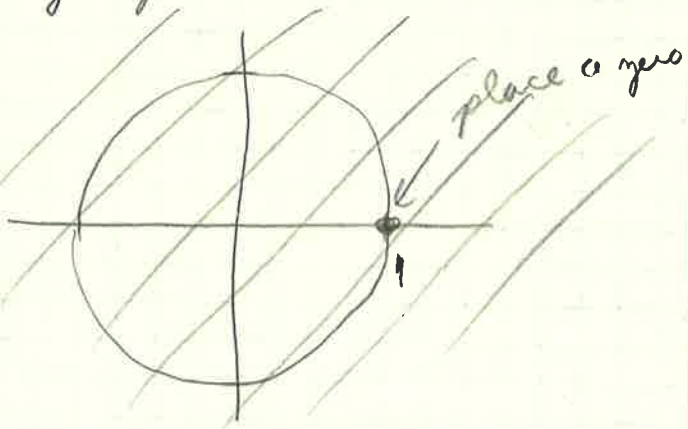


- DC removal

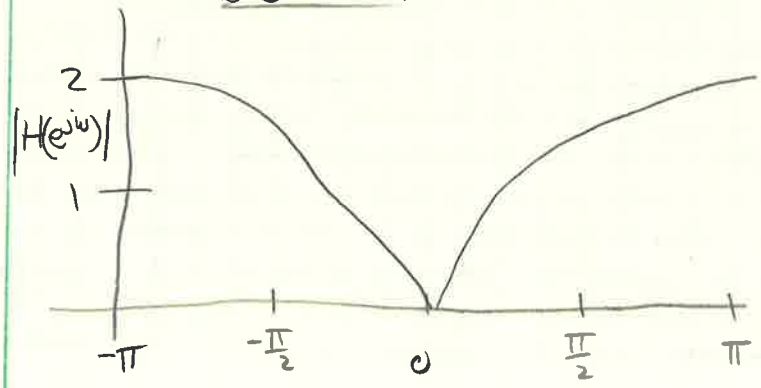
- a DC-balanced signal has zero sum: $\lim_{N\to\infty} \sum_{n=-N}^{N} x[n] = 0$
  i.e. there is no Direct Current component
- Its DTFT value at zero is zero
- We want to remove the DC bias from a non zero-centered signal
- We want to kill the frequency component at $\omega = 0$

$H(z) = 1 - z^{-1}$

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



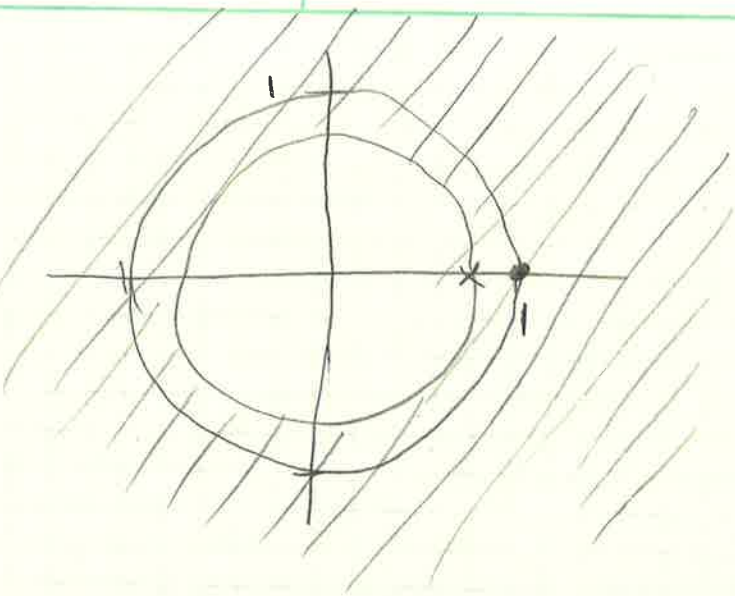
DC Notch



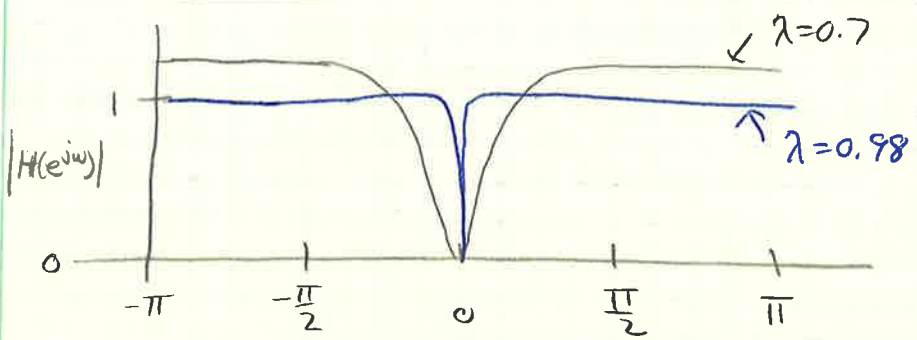
- DC removal, improved

$$H(z) = \frac{1 - z^{-1}}{1 - \lambda z^{-1}}$$

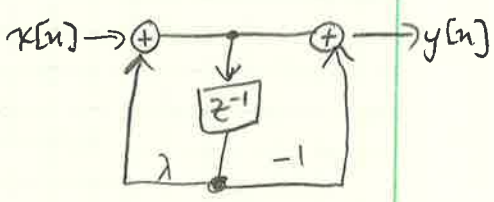
$$y[n] = \lambda y[n-1] + x[n] - x[n-1]$$



DC Notch



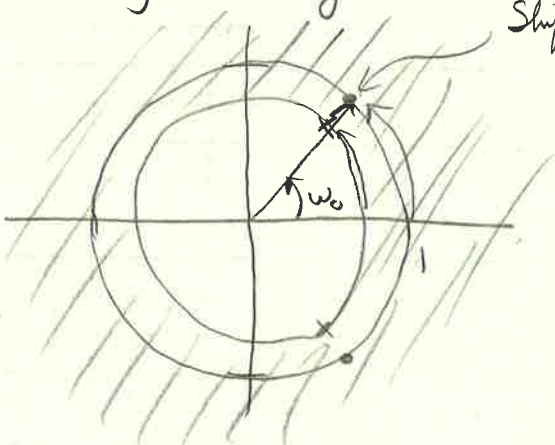
filter structure

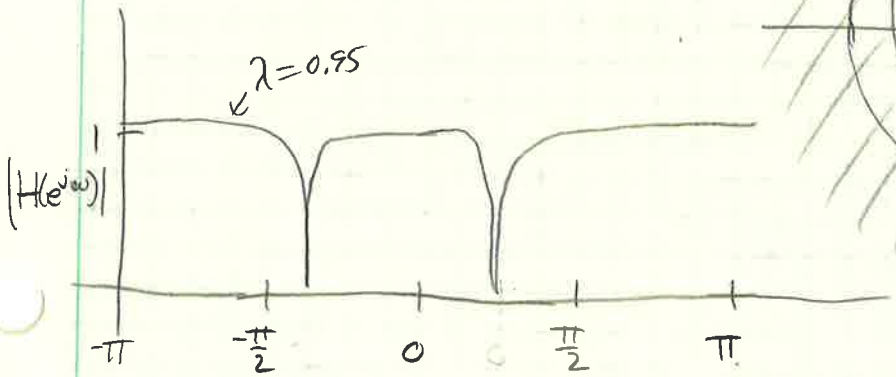


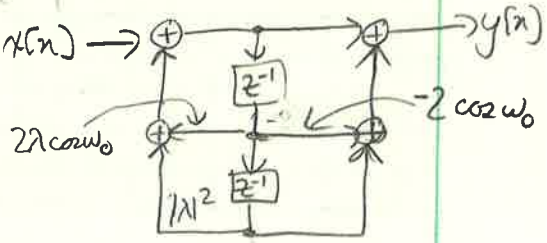
- Hum removal

- Similar to DC removal but we want to remove a specific nonzero frequency
- Very useful for musicians: amplifiers for electric guitars pick up the hum from the electric mains (50 Hz in Europe and 60 Hz in North America)
- We need to tune the hum removal according to country

$$H(z) = \frac{(1 - e^{j\omega_0} z^{-1})(1 - e^{-j\omega_0} z^{-1})}{(1 - \lambda e^{j\omega_0} z^{-1})(1 - \lambda e^{-j\omega_0} z^{-1})}$$







4.9 Filter Design Part 3

4.9.a Filter Specifications

- The filter design problem

- You are given a set of requirements
  - frequency response: passband(s) and stopband(s)
  - phase: overall delay, linearity
  - some limit on computational resources and/or numerical precision

You must determine N, M, a_k's and b_k's in

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{M-1} z^{-(M-1)}}{a_0 + a_1 z^{-1} + \cdots + a_{N-1} z^{-(N-1)}}$$

in order to best fulfill the requirements

- Example: lowpass specs

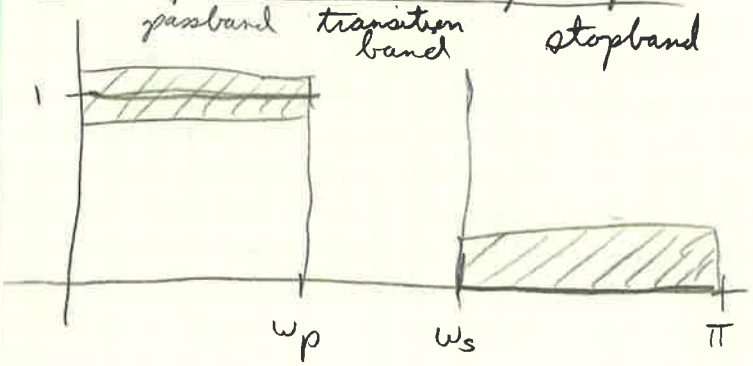
Want cutoff frequency ω_c and magnitude gain

- Practical limitations

- passband/stopband transitions cannot be infinitely sharp
  ⇒ use transition bands
- magnitude response cannot be constant over an interval
  ⇒ specify magnitude tolerances over bands

- in general:
  - smaller transition bands ⇒ higher filter order
  - smaller error tolerances ⇒ higher filter order
  - higher filter order ⇒ more expensive, larger delay

- Example: realistic lowpass specs



- Why we can't have a flat response

$H(z) = B(z)/A(z)$, with A and B polynomials

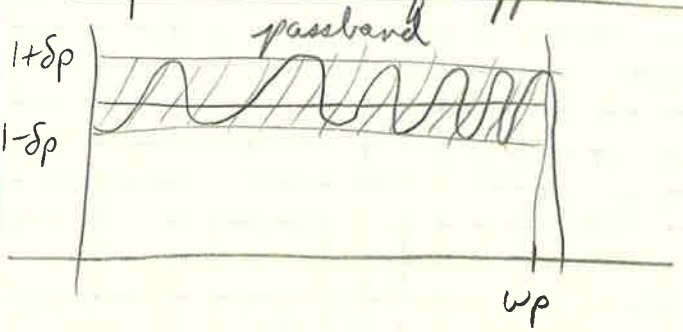
$H(e^{j\omega}) = c$ over an interval $\Rightarrow B(z) - cA(z) = 0$ over an interval

$\Rightarrow B(z) - cA(z)$ has an infinite number of roots

$\Rightarrow B(z) - cA(z) = 0$ for all values of z

$\Rightarrow H(e^{j\omega}) = c$ over the entire $[-\pi, \pi)$ interval.

- Important case: equiripple error



- The big questions
  - IIR or FIR
  - how to determine the coefficients?
  - how to evaluate the performance?

- IIRs: pros and cons
  - Pros:
    - computationally efficient
    - strong attenuation easy
    - good for audio
  - Cons:
    - stability issues
    - difficult to design for arbitrary response
    - nonlinear phase

- FIRs: pros and cons
  - Pros:
    - always stable
    - optimal design techniques exist
    - can be designed with linear phase
  - Cons:
    - computationally much more expensive
    - may "sound" harsh

- The design methods

  - finding N, M, a_k's and b_k's from specs is a hard nonlinear problem
  - established methods
    - IIR: conversion of analog design
    - FIR: optimal minimax filter design

4.9.6 IIR design

- IIR: conversion of analog design

Filter design was an established art long before digital processing appeared
- lots of nice analog filters exist
- methods exist to "translate" the analog design into a rational transfer function
- most numerical packages (Matlab, etc.) provide ready-made routines
- design involves specifying some parameters and testing that the specs are fulfilled (trial and error)

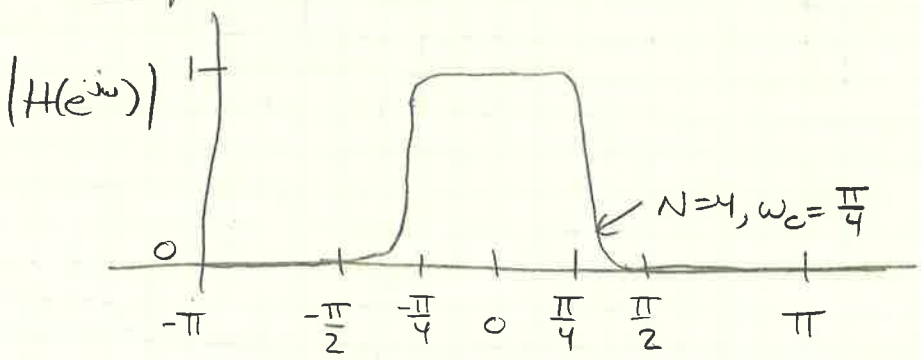
- Butterworth lowpass

Magnitude response:
- maximally flat
- monotonic over $[0, \pi]$

Design parameters:
- order N
- cutoff frequency

Test values:
- width of transition band
- passband error

- example



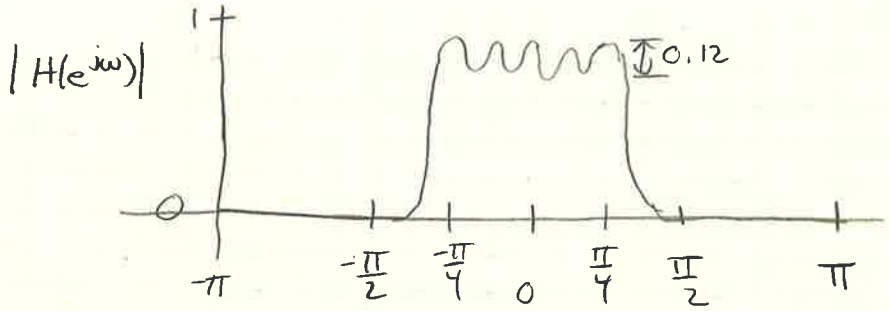
- Chebyshev lowpass

Magnitude response:
- equiripple in passband
- monotonic in stopband

Test values:
- width of transition band
- stopband error

Design parameters:
- order N
- passband max error
- cutoff frequency

- example : $N=4$, $\omega_c = \frac{\pi}{4}$, $e_{max} = 12\%$



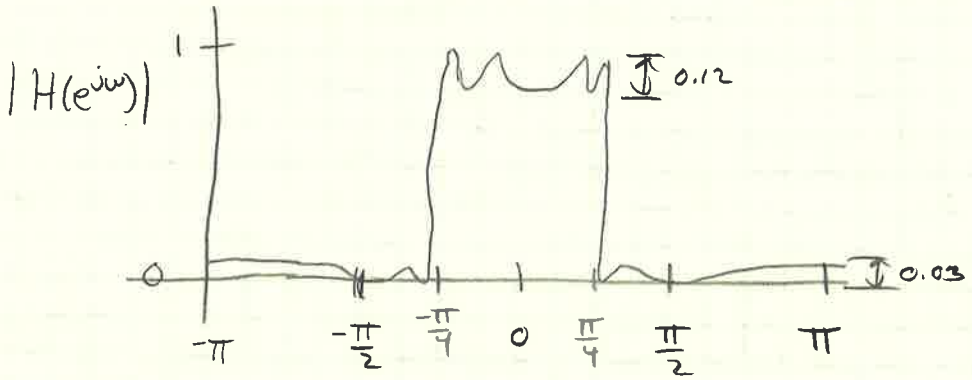
- Elliptic lowpass

Magnitude response:
- equiripple in passband and stopband

Design parameters:
- order N
- cutoff frequency
- passband maperror
- stopband min attenuation

Test value:
- width of transition band

- example : $N=4$, $\omega_c = \frac{\pi}{4}$, $e_{max} = 12\%$, $att_{min} = 0.03$



4.9.c FIR design

- FIR: optimal minimax design

FIR filters are a digital signal processing "exclusivity". In the 70s, Parks and McClellan developed an algorithm to design optimal FIR filters:

- linear phase
- equiripple error in passband and stopband

Algorithm proceeds by minimizing the maximum error in passband and stopband

- Linear phase

Linear phase derives from a symmetric or antisymmetric impulse response

- Linear phase (Type 1)

$h[C+n] = h[C-n]$

$h'[n] = h[n+C]$ centered on 0

$h'[n] = h'[n]$

$H(z) = z^{-C} H'(z)$

$$H'(z) = \sum_{n=-M}^{M} h'[n] z^{-n} = h'[0] + \sum_{n=1}^{M} h'[n](z^{n} + z^{-n})$$

$$H'(e^{j\omega}) = h'[0] + \sum_{n=1}^{M} h'[n](e^{j\omega n} + e^{-j\omega n})$$

$$= h'[0] + 2\sum_{n=1}^{M} h'[n] \cos \omega n \quad \in \mathbb{R} \quad (0\text{-phase})$$

$$H(e^{j\omega}) = \underbrace{\left[h[C] + 2\sum_{n=1}^{M} h[n] \cos(n-C)\omega\right]}_{\in \mathbb{R}} e^{-j\omega C}$$

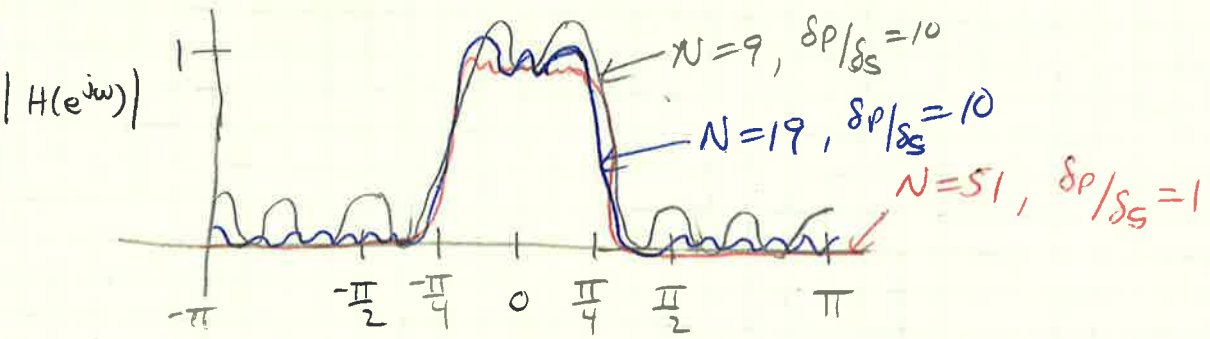
- Minimax lowpass

Magnitude response:
- equiripple in passband and stopband

Design parameters:
- order N (number of taps)
- passband edge ω_p
- stopband edge ω_s
- ratio of passband to stopband error δ_p/δ_s

Test value:
- passband max error
- stopband max error

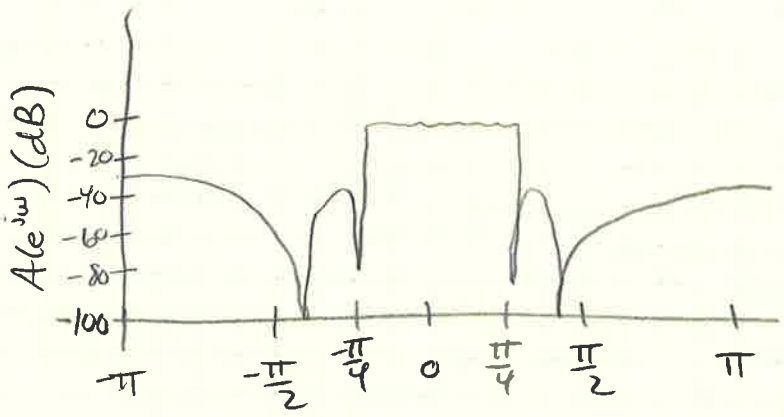
- example: $\omega_s = 0.2\pi$, $\omega_p = 0.3\pi$



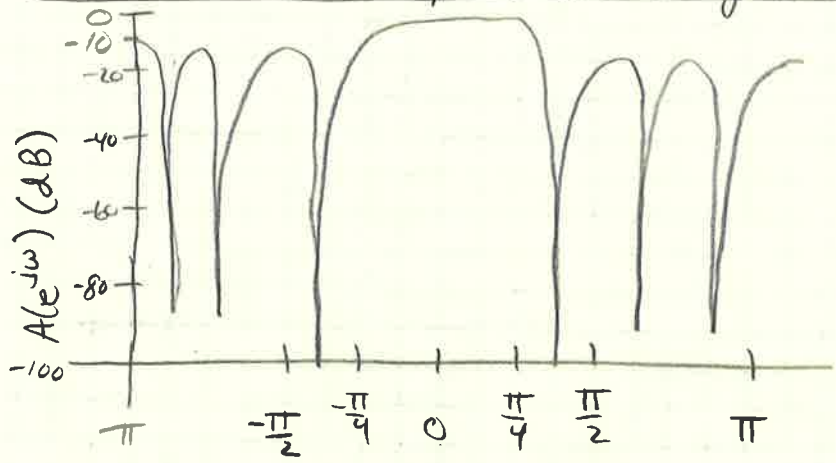
- Magnitude response in decibels

- filter max passband magnitude G
- filter attenuation expressed in decibels as $A_{dB} = 20 \log_{10}(|H(e^{j\omega})|/G)$
- useful to compare attenuations between filters

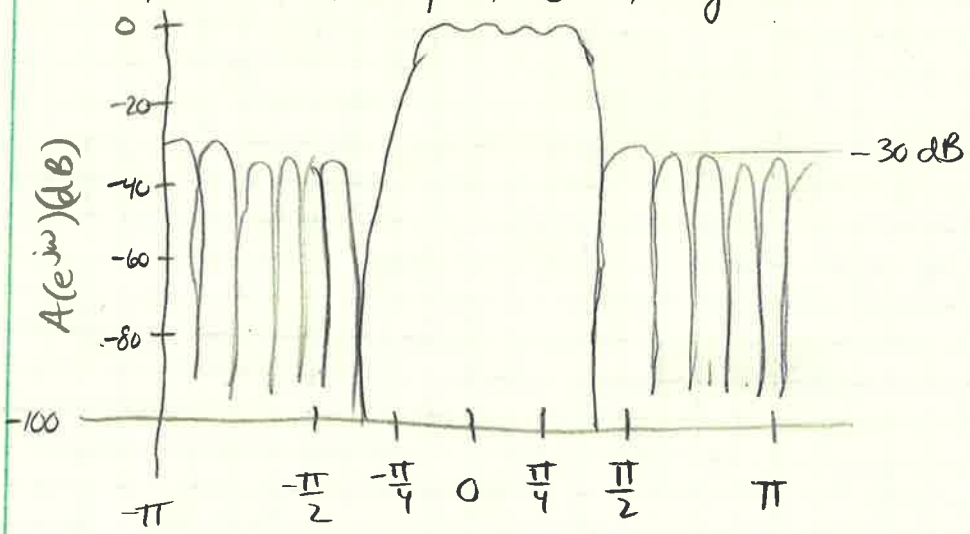
- 4th-order elliptic lowpass, $\omega_c = \pi/4$, log scale



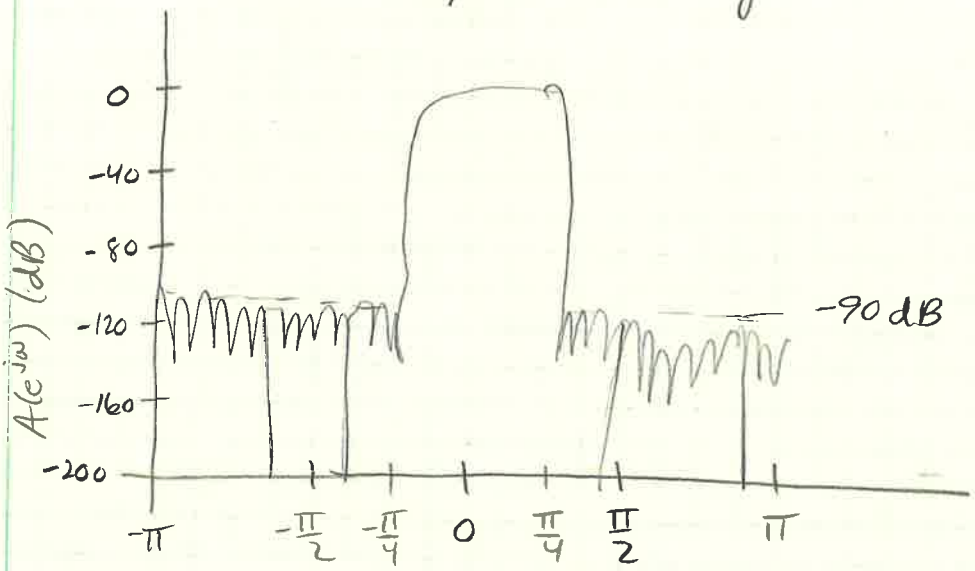
- 9-tap minimax lowpass, $\omega_c = \frac{\pi}{4}$, log scale



- 19-tap minimax lowpass, $\omega_c = \frac{\pi}{4}$, log scale



-51- tap minimax lowpass, $\omega_c = \frac{\pi}{4}$, log scale



-Life beyond lowpass

The IIR and FIR methods we just described can be used to design more general filter types than lowpass, with only minor modifications

- IIR bandpass and highpass can be obtained by modulating the lowpass response
- optimal FIR bandpass and highpass can be designed by the Parks-McClellan algorithm
- optimal FIR can be designed with piecewise linear magnitude response
- the literature on filter design is vast: this is just the tip of the iceberg!

4.8.5* Fractional delay and Hilbert filter

-The fractional delay

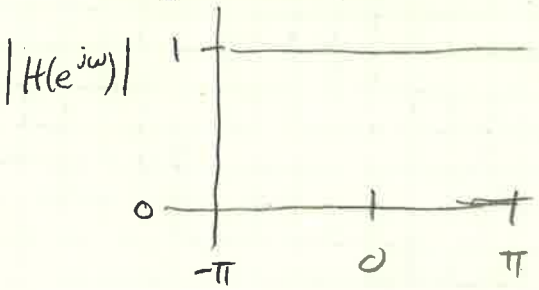
Consider a simple delay...

$$x[n] \longrightarrow \boxed{z^{-d}} \longrightarrow x[n-d] \qquad X(e^{j\omega}) \rightarrow e^{-j\omega d} X(e^{j\omega})$$

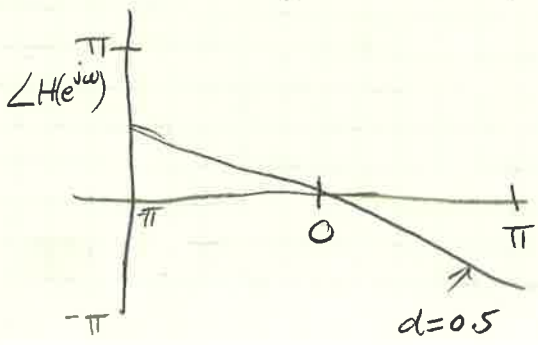
$$H(e^{j\omega}) = e^{-j\omega d}, \quad d \in \mathbb{Z}$$

What happens if, in $H(e^{j\omega})$, we use a non-integer $d \in \mathbb{R}$

Magnitude Response



Phase Response



impulse response

$$h[n] = \text{IDTFT}\{e^{-j\omega d}\}$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{-j\omega d} e^{j\omega n} d\omega$$

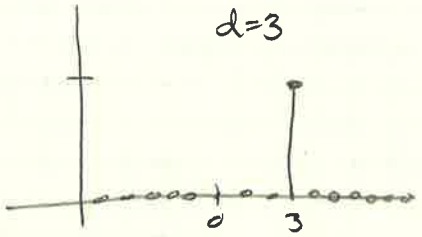
$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\omega(n-d)} d\omega$$

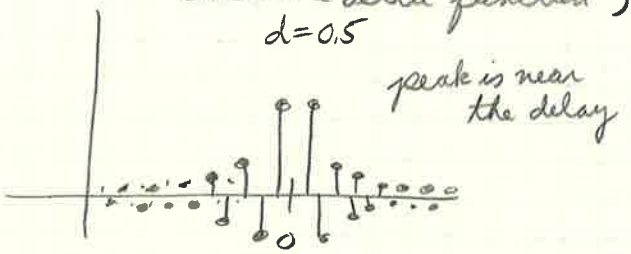
$$= \frac{1}{\pi(n-d)} \frac{e^{j\pi(n-d)} - e^{-j\pi(n-d)}}{2j}$$

$$= \frac{\sin\pi(n-d)}{\pi(n-d)}$$

$$= \text{sinc}(n-d)$$

(if argument is always an integer ⇒ sinc is a delta function)





– The Hilbert filter

a quirky machine

$$\cos(\omega_0 n) \longrightarrow \boxed{H(z)} \longrightarrow \sin(\omega_0 n)$$

can we build such a thing?

In the frequency domain,

$$H(e^{j\omega})\left[\tilde{\delta}(\omega-\omega_0)+\tilde{\delta}(\omega+\omega_0)\right] = -j\left[\tilde{\delta}(\omega-\omega_0)-\tilde{\delta}(\omega+\omega_0)\right]$$

we can derive two values: $\begin{cases} H(e^{j\omega_0}) = -j \\ H(e^{-j\omega_0}) = j \end{cases}$

For the machine to work at all frequencies:

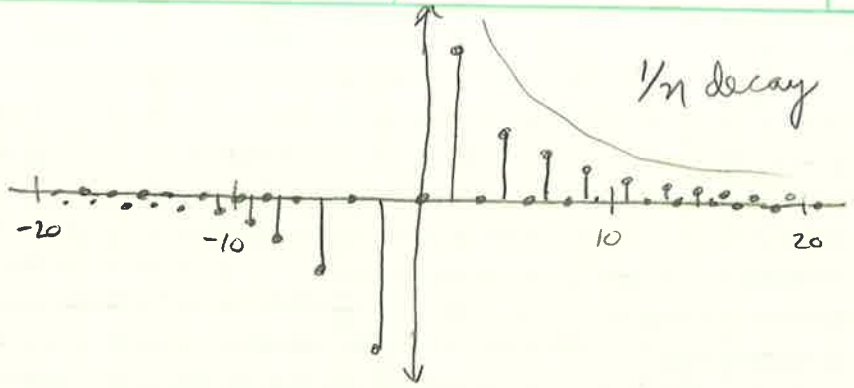
$$H(e^{j\omega_0}) = \begin{cases} -j, & 0 \le \omega < \pi \\ j, & -\pi \le \omega < 0 \end{cases} \quad (2\pi\text{-periodic})$$

$|H(e^{j\omega})| = 1, \forall\omega \Leftrightarrow$ Hilbert filter is allpass

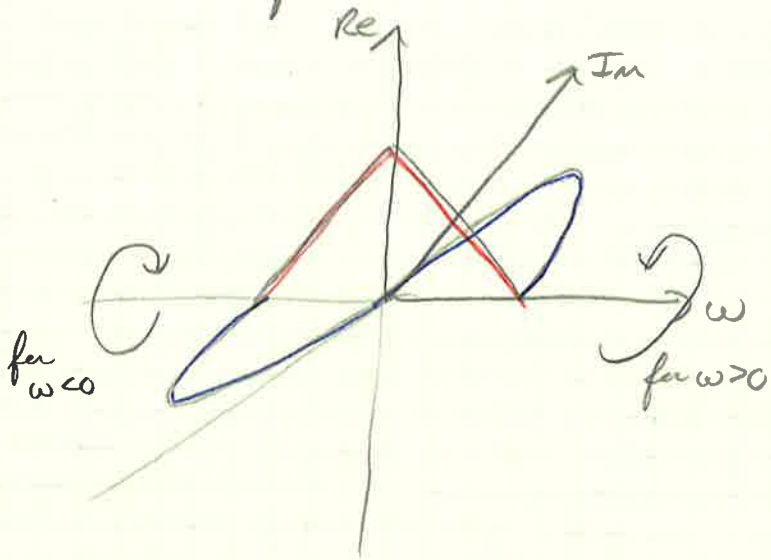
impulse response

$$h[n] = \frac{1}{2\pi}\int_{-\pi}^{0} je^{j\omega n} d\omega + \frac{1}{2\pi}\int_{0}^{\pi} -je^{j\omega n} d\omega = \frac{1}{2\pi n}\left[1 - e^{-j\pi n} - e^{-j\pi n} + 1\right]$$

$$= \begin{cases} \frac{2}{\pi n}, & n \text{ odd} \\ 0, & n \text{ even} \end{cases}$$



What does the Hilbert filter do?

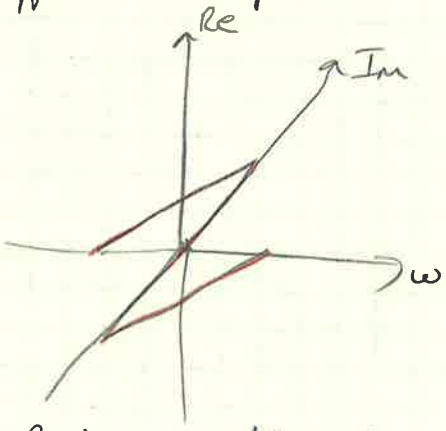


in Re-ω plane

in Im-ω plane

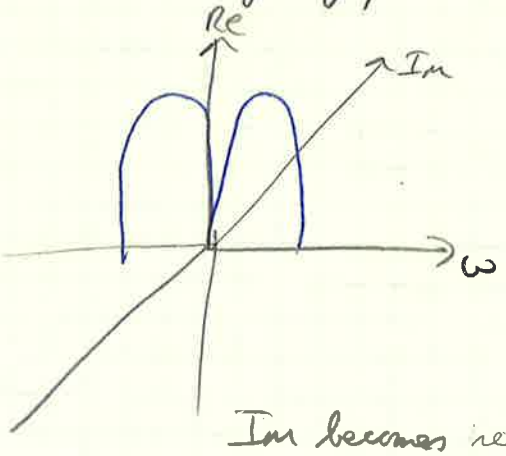
Hilbert filter imparts 90° rotations as shown

effect on the real part



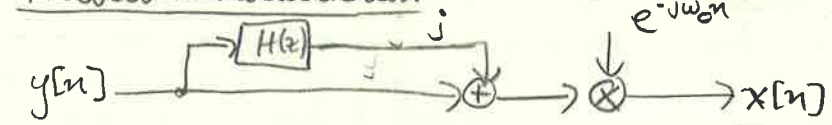
Re becomes antisymmetric and imaginary

effect on the imaginary part



Im becomes real and symmetric

- Hilbert demodulation



$x[n]\cos(\omega_0 n)$ $\quad (j y[n] * h[n] + y[n]) e^{-j\omega_0 n}$

4.10* Implementation of digital filters

- Algorithms for CCDE's

• C implementation of Leaky Integrator

$$y[n] = \lambda y[n-1] + (1-\lambda)x[n]$$

```
double Leaky(double x) {
  static const double lambda = 0.9;
  static double y = 0;

  y = lambda * y + (1 - lambda) * x;
  return y;
}
```

`static` gives memory of y's previous value

Testing the code

```
int main() {
  int n;
  for(n=0; n<20; n++)
    printf("%.4f ", Leaky(n==0 ? 1.0 : 0.0));
}
```

(n==0 ? 1.0 : 0.0) = δ[n]

Key points

- we need a "memory cell" to store previous output
- we need to initialize the storage before first use
- we need 2 multiplications and one addition per output sample

Python OOP

```
class Leaky:
  def __init__(self, lmb):
    self.lmb = lmb
    self.y = 0

  def compute(self, x):
    res = []
    for n in x:
      self.y = self.lmb * self.y + (1-self.lmb)*n
      res.append(self.y)
    return res
```

Testing the code:

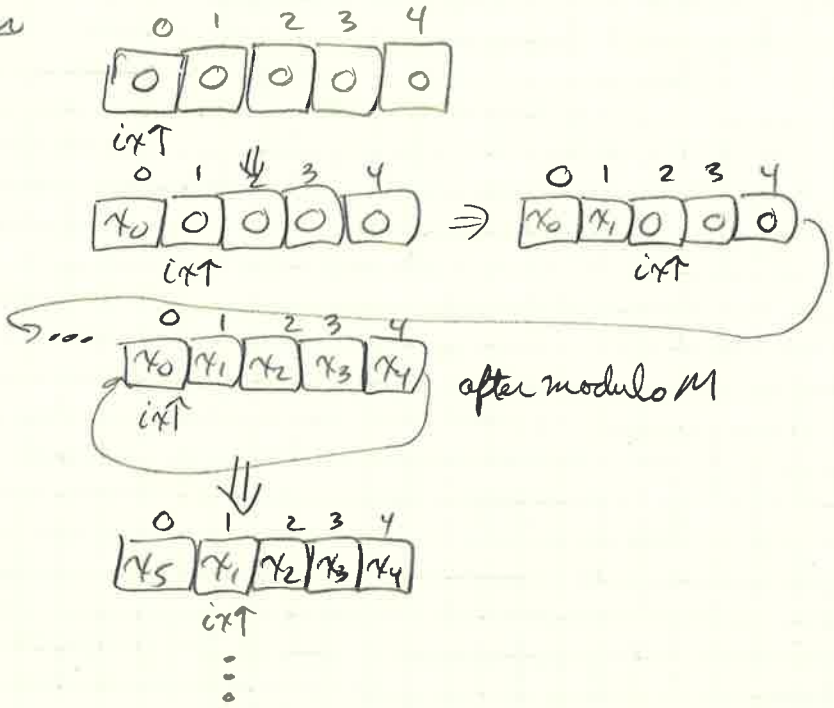
```
>> from leaky import Leaky
>> L = Leaky(0.95)
>> print(L.compute([0,0,0,0,1,0,0,0,0,0,0]))
```

n=0 ; [0,0,0,0,1,0,0,0,0,0,0] = δ[n]

Another old friend: MA

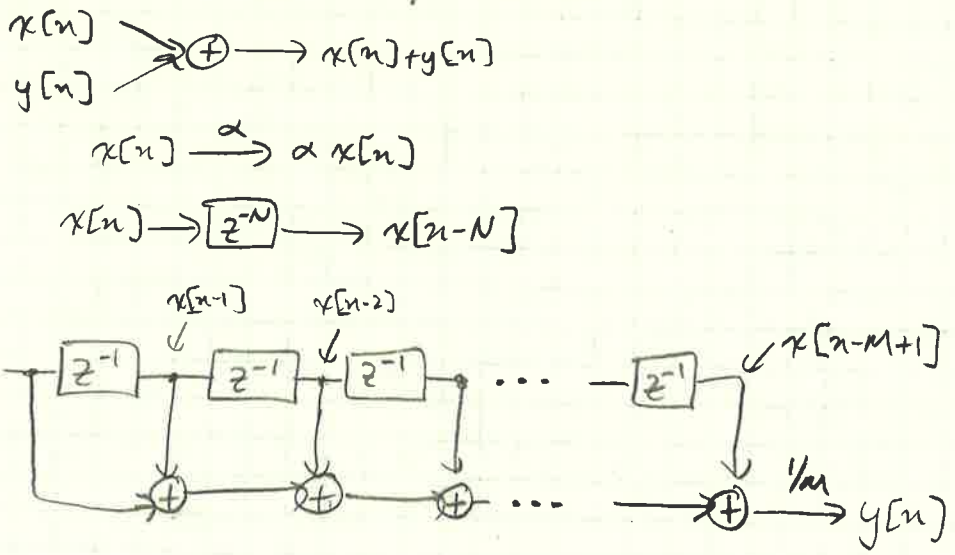
$$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k]$$

- Circular buffer



- key point
  - we now need M memory cells to store previous input values
  - we need to initialize the storage before use
  - we need 1 division and M additions per output sample

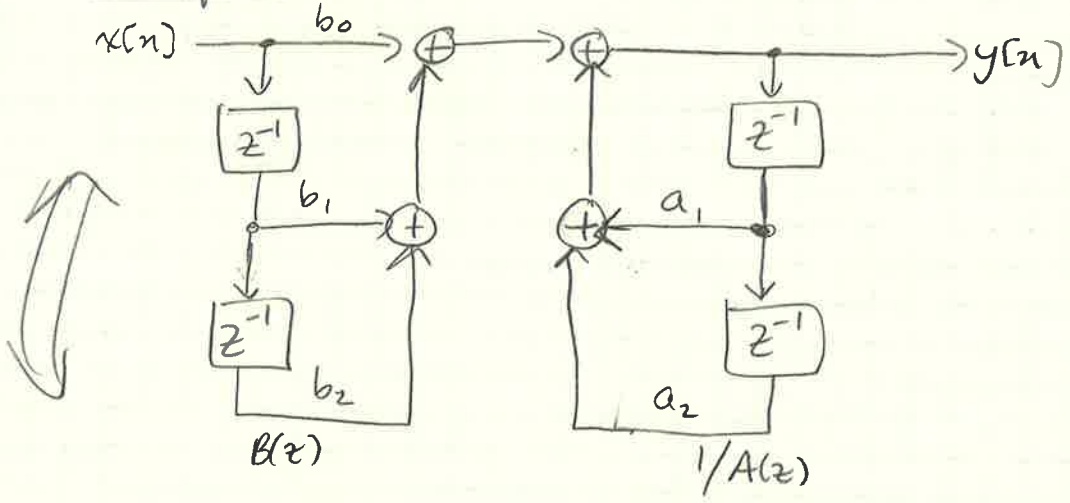
- We can abstract from the implementation



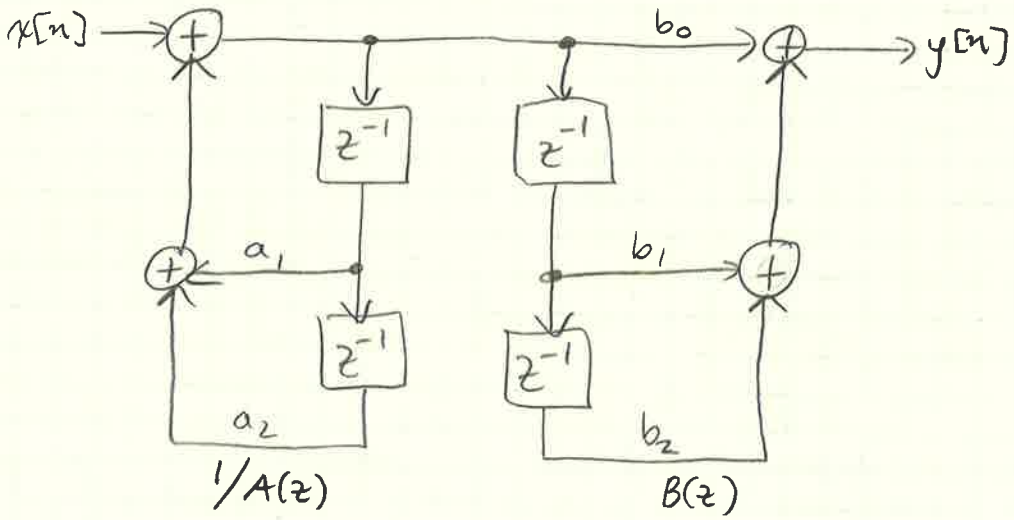
-The second-order section

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2}} = \frac{B(z)}{A(z)}$$

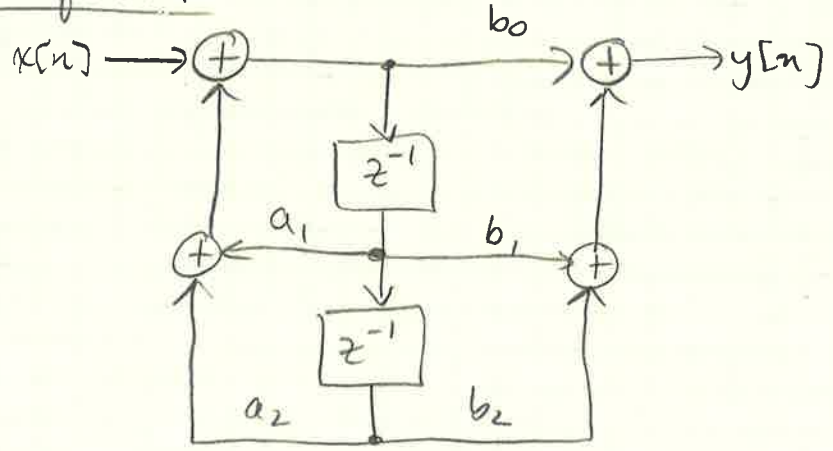
direct form I



direct form I, inverted order



direct form II



- Real-time processing - speed

If input samples arrive every T seconds, $\left(T = \frac{1}{44000} \text{ sec in audio}\right)$
- each output sample must be computed in at most T seconds
- number of operations becomes important (IIR vs FIR)
- some common tricks:
  - circular buffers are size 2^K (mod operation faster)
  - exploit the parallelism for some processor operations (fetch and compute)

- Real-time processing - accuracy

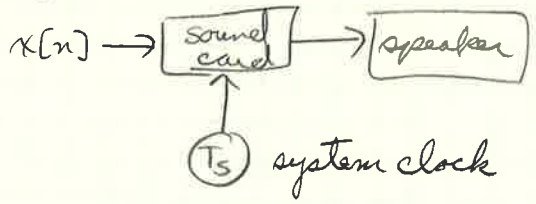
Processing units have finite precision arithmetic:
- overflow or underflow are a real problem
- more complex structures are more resilient (but less efficient)
- some common tricks:
  - use double precision in the accumulator
  - split the filter into low-order sections
  - use floating point

4.11* Real-time processing

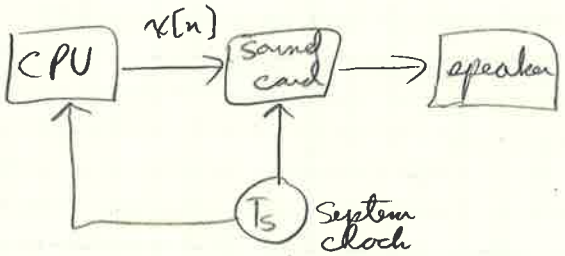
Everything works in sync with a system clock of period T_s:
- "record" a value $x_i[n]$
- process the value in a causal filter
- "play" the output $x_o[n]$

everything needs to happen in at most T_s seconds!

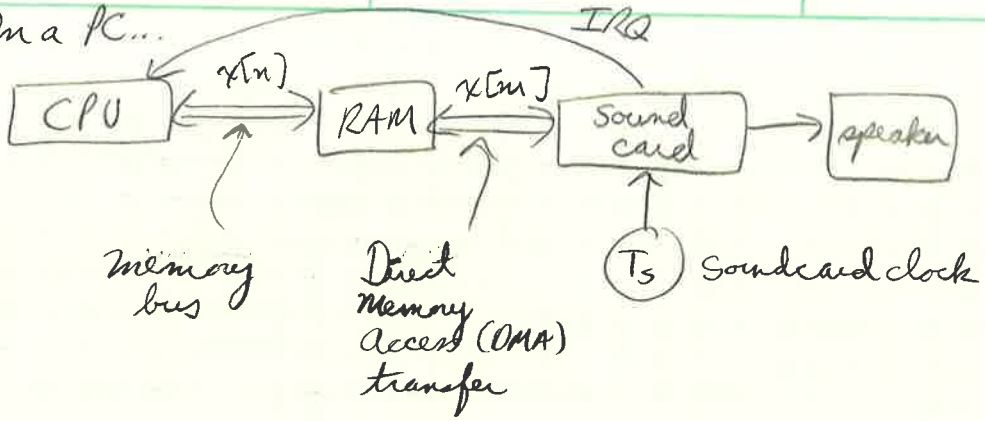
- Playing a sound



On dedicated hardware..



On a PC...

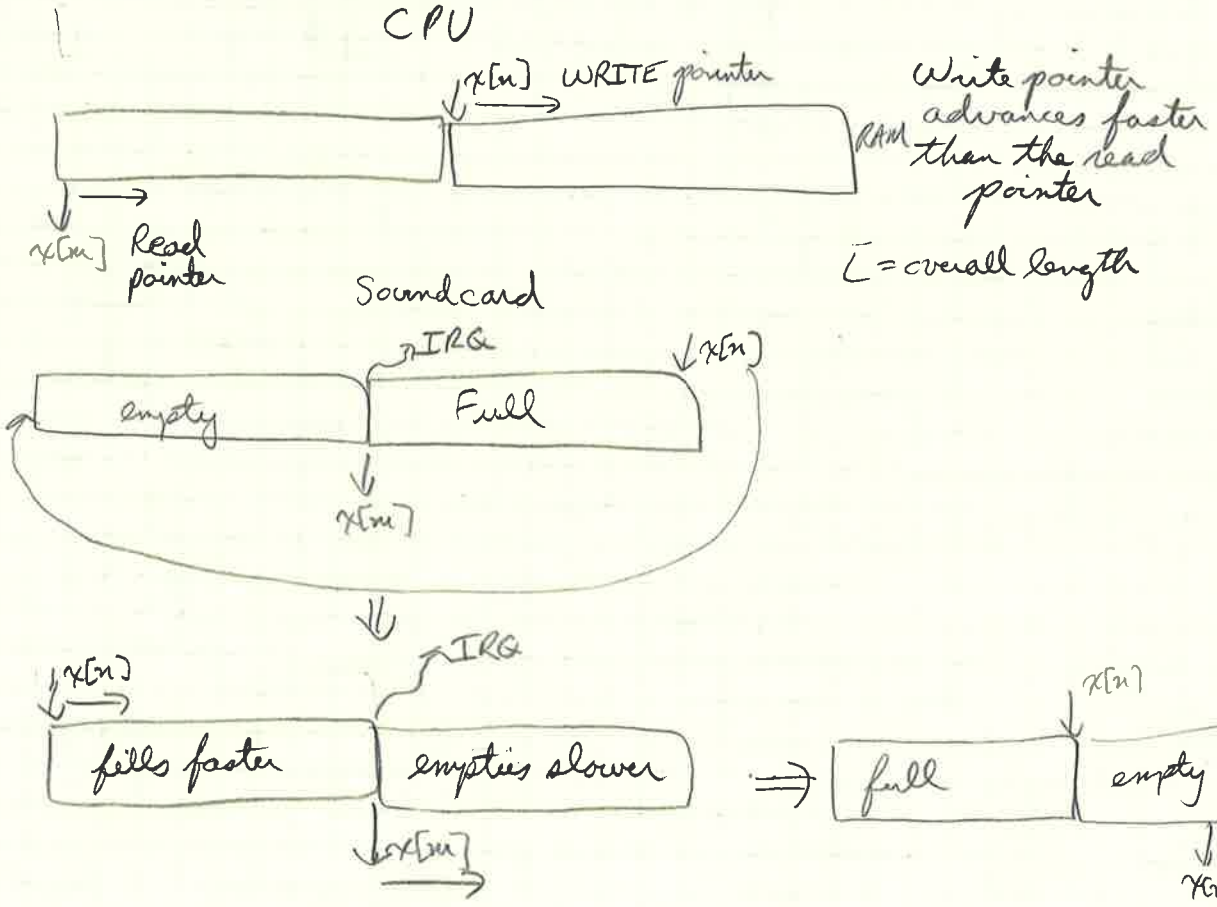


- Buffering

- Interrupt for each sample would be too much overhead
- Soundcard consumes sample in buffers
- Soundcard notifies when buffer ~~is~~ used up
- CPU can fill a buffer in less time than soundcard can empty it

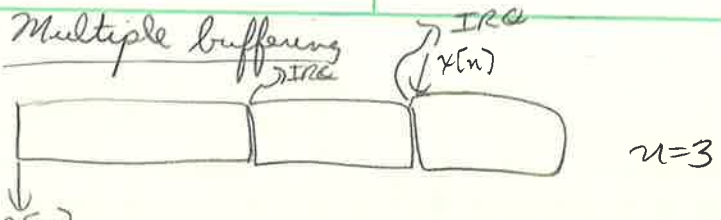
buffering introduces delay !

Example: double buffering (output)



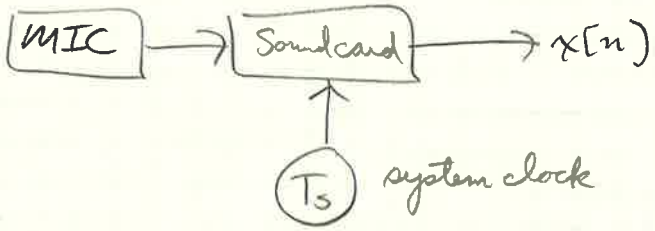
- double buffering introduces a delay $d = T_S \frac{L}{2}$ seconds
- If CPU doesn't fill the buffer fast enough: underflow

- Multiple buffering

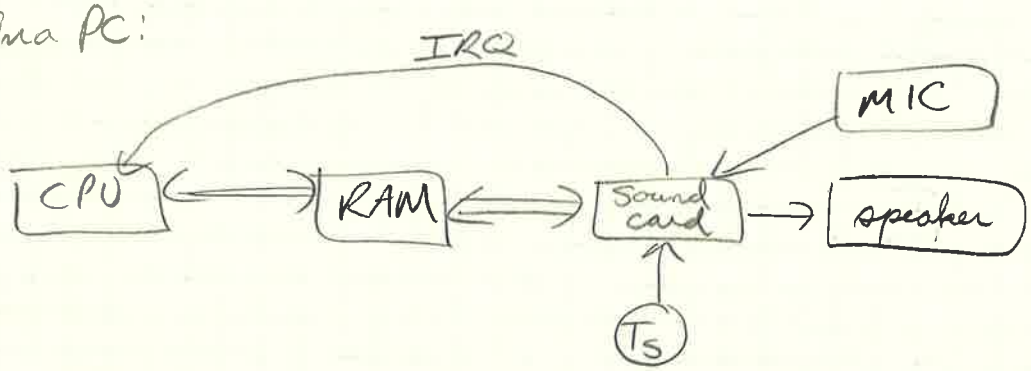


- Call the CPU more often (balance load)
- Keep reasonable overflow protection

- What about the input?



On a PC:

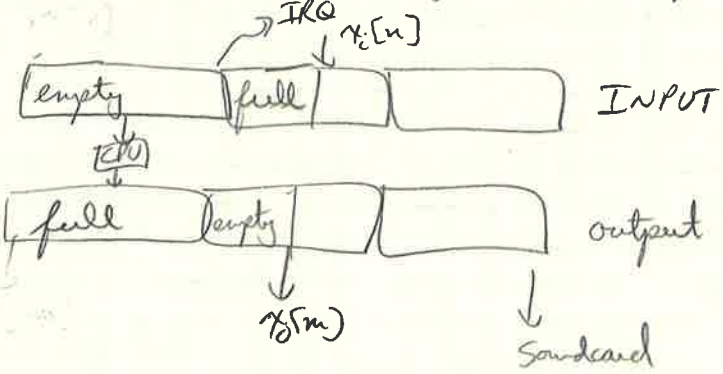


Double buffering — roles are reversed. The CPU takes samples out of the buffer.

- Putting it all together

- Multiple input buffers and output buffers
- Equal chunk sizes
- Input IRQ drives processing

- Real-time I/O processing with multiple buffering



$x_i(n)$ fills at same rate as $x_o[m]$ empties

- total delay $d = T - L$ seconds
- usually start output process first
- buffers can be collapsed

- Implementation
  - low level:
    - study soundcard data sheet (each one is different)
    - write code to program soundcard via writes to IO ports
    - write an interrupt handler
    - write the code to handle the data
  - high level:
    - choose a good API
    - write a callback function to handle the data

- Callback prototype

```
int Callback ( const void *input, void *output,
               unsigned long samples, ... );
```

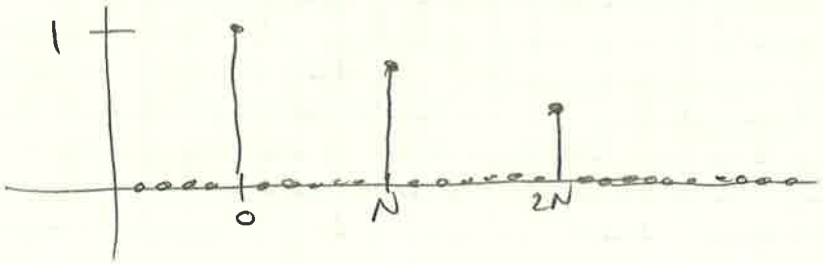
- Callback example

```
int Callback (const void *input, void *output, unsigned long samples)
{
  float * pIn  = (float*) input;
  float * pOut = (float*) output;
  for(int n=0; n<samples; n++)
     *pOut = Process (*pIn++);
}
```

Some processing function

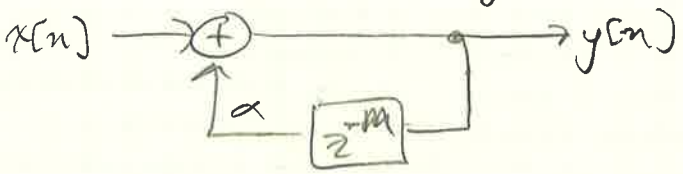
- Simple Echo

$$y[n] = \frac{ax[n] + bx[n-N] + cx[n-2N]}{a+b+c}$$



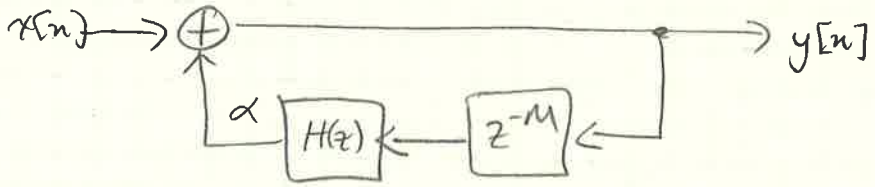
- A better echo

remember the KS algorithm? it's a sort of IIR echo



$$y[n] = \alpha y[n-M] + x[n]$$

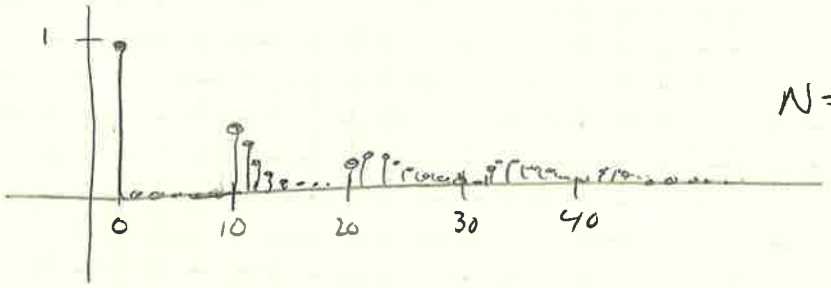
a natural echo has a lowpass characteristic



$$y[n] = \alpha (h * y)[n-M] + x[n]$$

Choose for instance $H(z)$ = Leaky integrator:

$$y[n] = x[n] - \lambda x[n-1] + \lambda y[n-1] + \alpha(1-\lambda) y[n-N]$$



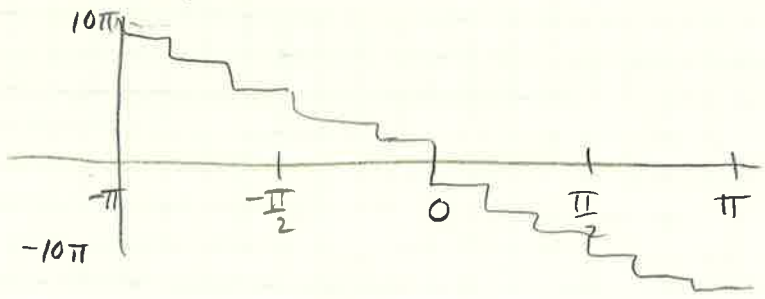
$N = 10, \lambda = 0.6, \alpha = 0.8$

- Reverb

- Reverb is given by the superposition of many many echos with different delays and magnitudes
- Many ways to simulate, always rather costly
- A cheap alternative is to use an allpass filter

$$H(z) = \frac{-\alpha + z^{-N}}{1 - \alpha z^{-N}}$$

phase response



S.1 The continuous-time world

S.1.a The continuous-time paradigm

-Digital processing of signals from/to the analog world

- Input is continuous-time: $x(t)$
- Output is continuous-time: $y(t)$
- Processing is on sequences: $x[n], y[n]$

analog world → [processing] → analog world

examples: MP3, digital photography

-Digital processing of signals to the analog world

- Input is discrete-time: $x[n]$
- Output is continuous-time: $y(t)$
- processing is on sequences: $x[n], y[n]$

digital world → [processing for synthesis] → analog world

examples: computer graphics, video games

-Digital processing of signals from the analog world

- Input is continuous-time: $x(t)$
- Output is discrete-time: $y[n]$
- Processing is on sequences: $x[n], y[n]$

analog world → [processing for analysis] → digital world

examples: control systems, monitoring

-Two views of the world

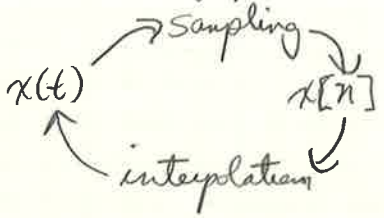
digital worldview:
- arithmetic
- combinatorics
- computer science
- DSP

- countable integer index n
- sequences $x[n] \in l_2(\mathbb{Z})$
- frequency $\omega \in [-\pi, \pi]$
- DTFT: $l_2(\mathbb{Z}) \to L_2([-\pi, \pi])$

analog worldview:
- calculus
- distributions
- system theory
- electronics

- real-valued time t (sec)
- functions $x(t) \in L_2(\mathbb{R})$
- frequency $\Omega \in \mathbb{R}$ (rad/sec)
- FT: $L_2(\mathbb{R}) \to L_2(\mathbb{R})$

- Bridging the gap



5.1.b Continuous-time signal processing

- About continuous time

- time: real variable t
- signal $x(t)$: complex functions of a real variable
- finite energy: $x(t) \in L_2(\mathbb{R})$
- inner product in $L_2(\mathbb{R})$

$$\langle x(t), y(t) \rangle = \int_{-\infty}^{\infty} x^*(t)\, y(t)\, dt$$

- energy: $\|x(t)\|^2 = \langle x(t), x(t) \rangle$

- Analog LTI filters

$$x(t) \longrightarrow \boxed{\mathcal{H}} \longrightarrow y(t)$$

$$y(t) = (x * h)(t) = \langle h^*(t-\tau), x(\tau) \rangle$$
$$= \int_{-\infty}^{\infty} x(\tau)\, h(t-\tau)\, d\tau$$

- Fourier analysis

- In discrete time max angular frequency is $\pm\pi$
- In continuous time no max frequency: $\Omega \in \mathbb{R}$
- Concept is the same:

$$X(j\Omega) = \int_{-\infty}^{\infty} x(t)\, e^{-j\Omega t}\, dt \quad \leftarrow \text{not periodic!}$$

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(j\Omega)\, e^{j\Omega t}\, d\Omega$$

- Real-world frequency

- Ω expressed in rad/s
- $F = \frac{\Omega}{2\pi}$, expressed in Hertz $\left(\frac{1}{s}\right)$
- period $T = \frac{1}{F} = \frac{2\pi}{\Omega}$

Example: $x(t) = \exp\left(\frac{-t^2}{2\sigma^2}\right)$

$$X(j\Omega) = \sigma\sqrt{2\pi}\exp\left(\frac{-\sigma^2}{2}\Omega^2\right)$$

- Convolution theorem

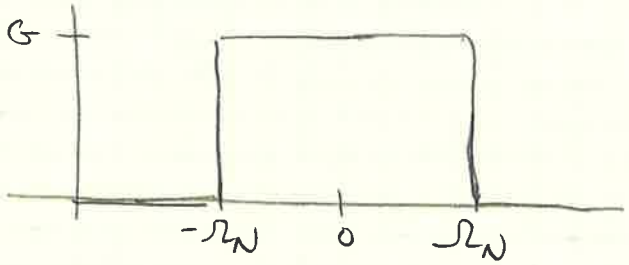
$x(t) \longrightarrow \boxed{H} \longrightarrow y(t)$

$$Y(j\Omega) = X(j\Omega)\,H(j\Omega)$$

- A new concept: bandlimited functions

Ω_N - bandlimitedness: $X(j\Omega) = 0$ for $|\Omega| > \Omega_N$

- Prototypical bandlimited function



$$\Phi(j\Omega) = G\,\mathrm{rect}\left(\frac{\Omega}{2\Omega_N}\right)$$

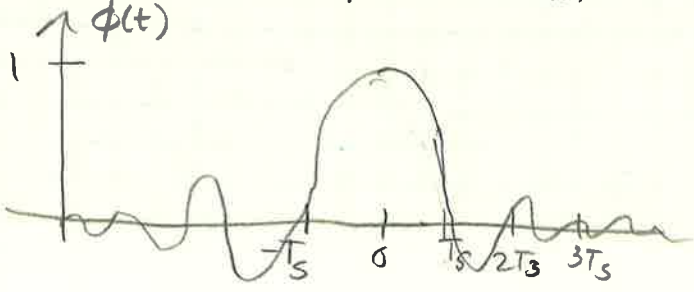
$$\phi(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\Phi(j\Omega)\,e^{j\Omega t}\,d\Omega$$

$$= \dots$$

$$= G\frac{\Omega_N}{\pi}\,\mathrm{sinc}\left(\frac{\Omega_N}{\pi}t\right)$$

- normalization: $G = \frac{\pi}{\Omega_N}$
- total bandwidth: $\Omega_B = 2\Omega_N$
- define $T_S = \frac{2\pi}{\Omega_B} = \frac{\pi}{\Omega_N}$

$$\Rightarrow \Phi(j\Omega) = \frac{\pi}{\Omega_N}\,\mathrm{rect}\left(\frac{\Omega}{2\Omega_N}\right) \quad \Rightarrow \phi(t) = \mathrm{sinc}\left(\frac{t}{T_S}\right)$$



zero-crossings at multiples of T_S

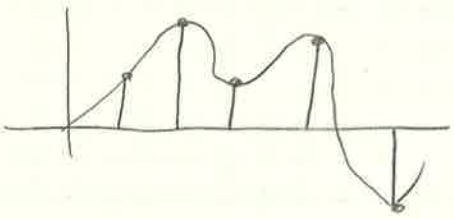
5.2 Interpolation

5.2.a Polynomial interpolation

- Interpolation

$x[n] \rightarrow x(t)$: "fill the gaps" between samples

- example



- requirements
  - decide on T_s
  - make sure $x(nT_s) = x[n]$
  - make sure $x(t)$ is smooth

- why smoothness?
  - jumps (1st order discontinuities) would require the signal to move "faster than light"...
  - 2nd order discontinuities would require infinite acceleration
  - ...
  - the interpolation should be infinitely differentiable
  - "natural" solution : polynomial interpolation

- Polynomial interpolation

- N points → polynomial of degree $(N-1)$
- $p(t) = a_0 + a_1 t + a_2 t^2 + \dots + a_{N-1} t^{N-1}$
- "naive" approach:

$$\begin{cases} p(0) = x[0] \\ p(T_s) = x[1] \\ p(2T_s) = x[2] \\ \vdots \\ p((N-1)T_s) = x[N-1] \end{cases}$$

- Without loss of generality:
  - consider a symmetric interval $I_N = [-N, \dots, N]$
  - set $T_s = 1$

$$\begin{cases} p(-N) = x[-N] \\ p(-N+1) = x[-N+1] \\ \vdots \\ p(0) = x[0] \\ p[N] = x[N] \end{cases}$$

- Lagrange interpolation

- P_N : space of degree-$2N$ polynomials over I_N

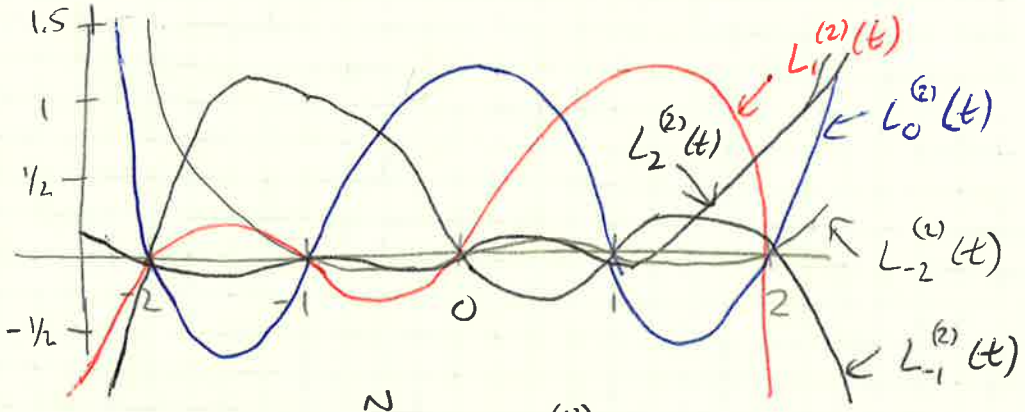
- a basis for P_N is the family of $2N+1$ Lagrange polynomials

$$L_n^{(N)}(t) = \prod_{\substack{k=-N \\ k\neq n}}^{N} \frac{t-k}{n-k} \quad , n = -N, \dots, N$$

- example : Pick $N=1 \Rightarrow L_{-1}^{(1)}, L_0^{(1)}, L_1^{(1)}$

$$L_0^{(1)}(t) = \prod_{\substack{k=-1 \\ k\neq 0}}^{1} \frac{t-k}{-k} = \frac{t+1}{1} \cdot \frac{t-1}{-1} = 1-t^2$$

$$L_1^{(1)}(t) = \frac{t^2+t}{2} \quad , \quad L_{-1}^{(1)}(t) = \frac{t^2-t}{2}$$



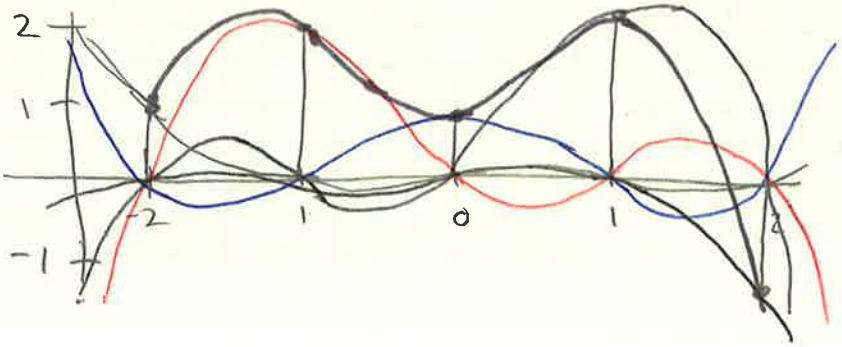
$$p(t) = \sum_{n=-N}^{N} x[n]\, L_n^{(N)}(t)$$

- The Lagrange interpolation is the sought-after polynomial interpolation:
  - polynomial of degree $2N$ through $2N+1$ points is unique
  - the Lagrangian interpolator satisfies

$$p(n) = x[n], \quad -N \le n \le N$$

since

$$L_n^{(N)}(m) = \begin{cases} 1, & n = m \\ 0, & n \neq m \end{cases}, \quad -N \le n, m \le N$$



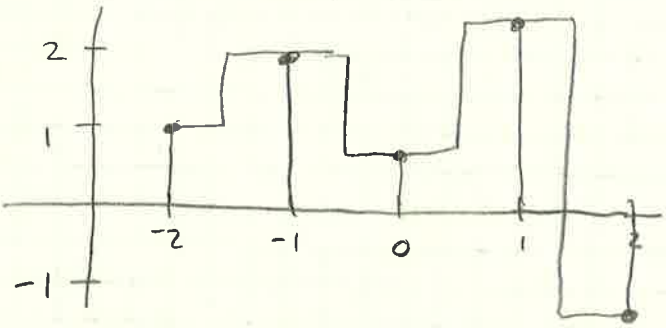
$-x[-2]L_{-2}^{(2)}(t)$
$-x[-1]L_{-1}^{(2)}(t)$
$-x[0]L_0^{(2)}(t)$
$-x[1]L_1^{(2)}(t)$
$-x[2]L_2^{(2)}(t)$
- total sum

- Polynomial interpolation
  - Key property:
    - maximally smooth (infinitely many continuous derivatives)
  - Drawback:
    - interpolation "bricks" depend on N

5.2.6 Local interpolation

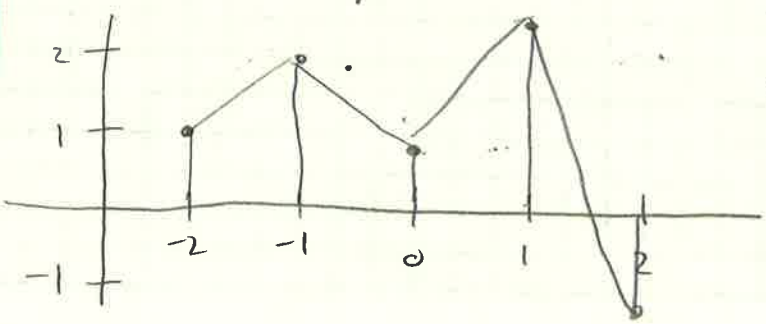
- Relaxing the interpolation requirements
  - decide on T_s
  - make sure $x(nT_s) = x[n]$
  - make sure $x(t)$ is smooth

- Zero-order interpolation



- $x(t) = x[\lfloor t+0.5 \rfloor]$, $-N \le t \le N$
- $x(t) = \sum_{n=-N}^{N} x[n]\, \mathrm{rect}(t-n)$
- interpolation kernel: $i_0(t) = \mathrm{rect}(t)$: "zero-order hold"
- interpolator's support is 1
- interpolation is not even continuous

- First-order interpolation

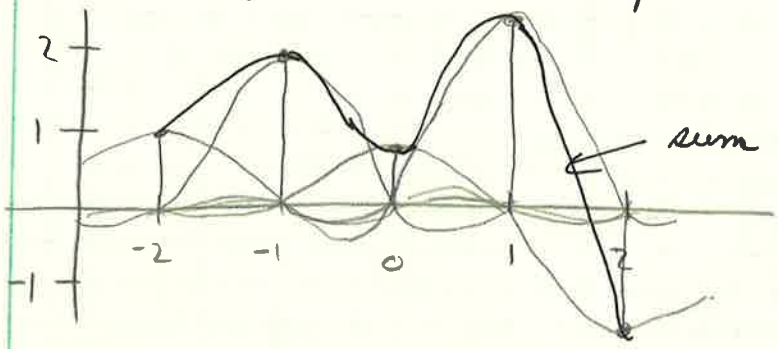


- "connect the dots" strategy
- $x(t) = \sum_{n=-N}^{N} x[n]\, i_1(t-n)$
- interpolation kernel: $\begin{cases} 1-|t| , & |t|<1 \\ 0 , & \text{otherwise} \end{cases}$
- interpolator's support is 2
- interpolator is continuous but derivative is not

- Third-order interpolation

$$x(t) = \sum_{n=-N}^{N} x[n]\, i_3(t-n)$$

- interpolation kernel obtained by splicing two cubic polynomials
- interpolator's support is 4
- interpolation is continuous up to 2nd derivative

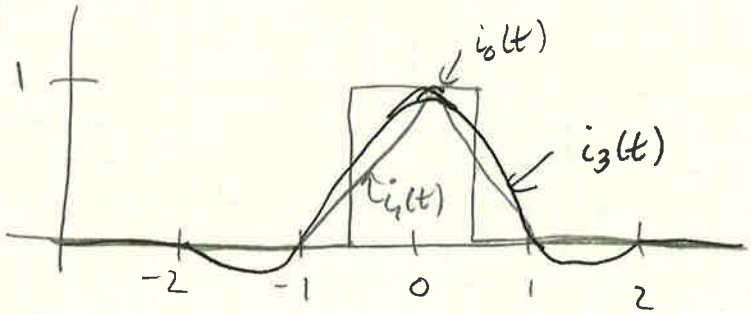


- Local interpolation schemes

$$x(t) = \sum_{n=-N}^{N} x[n]\, i_c(t-n)$$

Interpolator's requirements: $i_c(0) = 1$

$i_c(t) = 0$ for t a nonzero integer



Key property: same interpolating function independent of N

drawback: lack of smoothness

- A remarkable result:

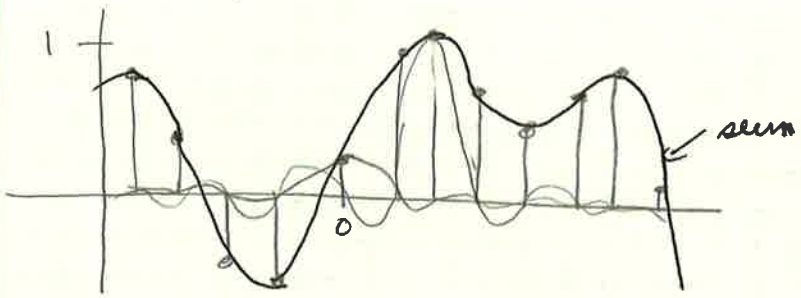
$$\lim_{N\to\infty} L_n^{(N)}(t) = \operatorname{sinc}(t-n)$$

- in the limit, local and global interpolation are the same!

- Sinc interpolation formula

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \operatorname{sinc}\left(\frac{t-nT_s}{T_s}\right)$$

- Sinc interpolation



- "Proof" that $L_n^{(N)}(t) \to \text{sinc}(t-n)$

- real proof is rather technical (see the book)
- intuition: $\text{sinc}(t-n)$ and $L_n^{(\infty)}(t)$ share an infinite number of zeros

$$\text{sinc}(m-n) = \delta[m-n] \quad , m,n \in \mathbb{Z}$$

$$L_n^{(N)}(m) = \delta[m-n] \quad , m,n \in \mathbb{Z}, \; -N \le n,m \le N$$

5.2.c Signal of the Day

- Band-limited interpolation

- $f_c(t)$ continuous-time, T-periodic, and band limited

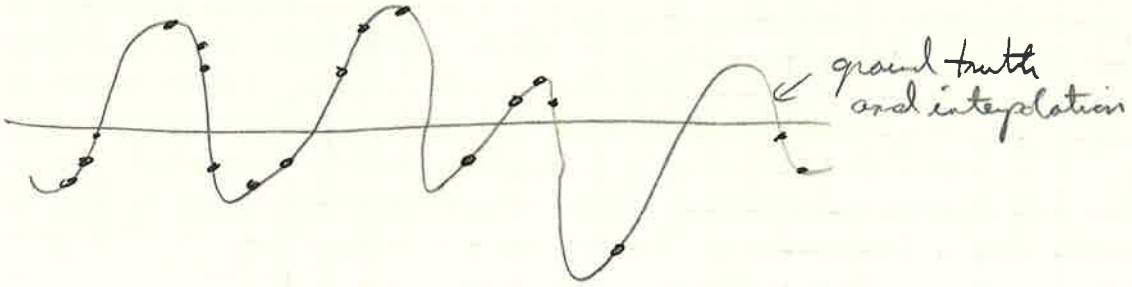
$$f_c(t) = \sum_{k=-M}^{M} c_k e^{j2\pi \frac{kt}{T}} \quad , \underline{c} = \{c_{-M}, \ldots, c_M\}$$

- known: $\{(t_i, z_i)\}_{i=1}^{N}$, $t_i \in \mathbb{R}$ and $z_i = f(t_i)$
- Recover:

$$\underline{\tilde{c}} = \underset{\underline{c}}{\text{argmin}} \sum_{i=1}^{N} |z_i - f_c(t_i)|^2$$

- Example 1

- Example: $f(t) = 2\sin(2\pi t) - 4\cos(8\pi t)$
- $M = 6, N = 20$



5.3 Sampling of bandlimited functions

5.3.a The spectrum of interpolated signals

- Sinc interpolation

The ingredients:
- discrete-time signal $x[n]$, $n \in \mathbb{Z}$ (with DTFT $X(e^{j\omega})$)
- interpolation interval T_s
- the sinc function

The result:
- a **smooth**, continuous-time signal $x(t)$, $t \in \mathbb{R}$

What does the spectrum of $x(t)$ look like?

- Key facts about the sync

$$\phi(t) = \operatorname{sinc}\left(\frac{t}{T_s}\right) \longleftrightarrow \Phi(j\Omega) = \frac{\pi}{\Omega_N} \operatorname{rect}\left(\frac{\Omega}{2\Omega_N}\right)$$

$$T_s = \frac{\pi}{\Omega_N} \qquad \Omega_N = \frac{\pi}{T_s}$$

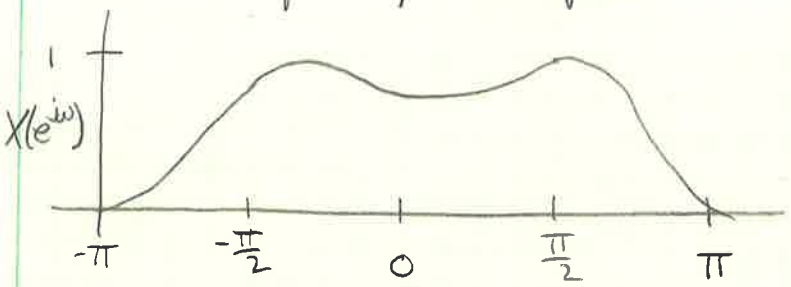
- Sinc interpolation

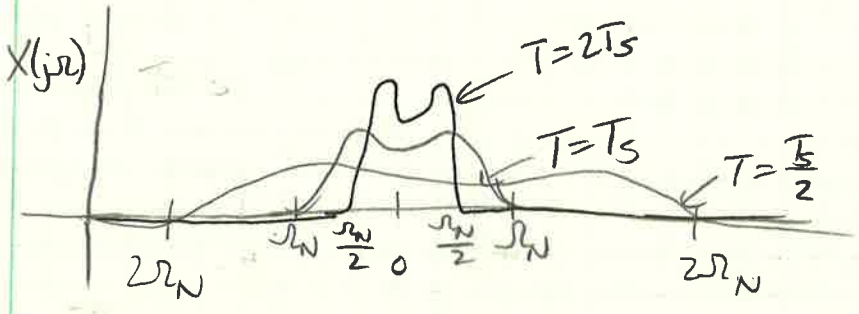
$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right)$$

- Spectral representation (1)

$$\begin{aligned}
X(j\Omega) &= \int_{-\infty}^{\infty} x(t) e^{-j\Omega t} dt \\
&= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[n] \operatorname{sinc}\left(\frac{t-nT_s}{T_s}\right) e^{-j\Omega t} dt \\
&= \sum_{n=-\infty}^{\infty} x[n] \int_{-\infty}^{\infty} \operatorname{sinc}\left(\frac{t-nT_s}{T_s}\right) e^{-j\Omega t} dt \\
&= \sum_{n=-\infty}^{\infty} x[n] \frac{\pi}{\Omega_N} \operatorname{rect}\left(\frac{\Omega}{2\Omega_N}\right) e^{-jnT_s\Omega} \\
&= \frac{\pi}{\Omega_N} \operatorname{rect}\left(\frac{\Omega}{2\Omega_N}\right) \sum_{n=-\infty}^{\infty} x[n] e^{-j\left(\frac{\pi}{\Omega_N}\right)\Omega n} \\
&= \begin{cases} \frac{\pi}{\Omega_N} X\left(e^{j\pi(\Omega/\Omega_N)}\right), & |\Omega| \le \Omega_N \\ 0, & \text{otherwise} \end{cases}
\end{aligned}$$

- Spectrum of interpolated signal





- Pick interpolation period T_S:
  - $X(j\Omega)$ is Ω_N-bandlimited, with $\Omega_N = \pi/T_S$
  - fast interpolation (T_S small) → wider spectrum
  - slow interpolation (T_S large) → narrower spectrum
  - (for those who remember) it's like changing the speed of a record player

- Space of bandlimited functions

$$x[n] \in \ell_2(\mathbb{Z}) \xrightarrow{T_S} x(t) \in L_2(\mathbb{R})$$
$$\xleftarrow{?} \quad \Omega_N\text{-BL}$$
↑ yes, by the sampling theorem

5.3.6 The space of bandlimited functions

Let's lighten the notation: for a while we will proceed with
$T_S = 1$, $\Omega_N = \pi$

(derivations in the general case are in the book)

- The road to the sampling theorem

Claims:
- the space of π-bandlimited functions is a Hilbert space
- The functions $\phi^{(n)}(t) = \text{sinc}(t-n)$, with $n \in \mathbb{Z}$, form a basis for the space
- If $x(t)$ is π-BL, the sequence $x[n] = x(n)$, $n \in \mathbb{Z}$, is a sufficient representation (i.e., we can reconstruct $x(t)$ from $x[n]$)

- The space π-BL
  - clearly a vector space because π-BL $\subset L_2(\mathbb{R})$ (and linear combinations of π-BL functions are π-BL functions
  - inner product is standard inner product in $L_2(\mathbb{R})$
  - completeness... that's more delicate

- The space of π-BL functions

recap:
- inner product: $\langle x(t), y(t)\rangle = \int_{-\infty}^{\infty} x^*(t) y(t)\, dt$
- convolution: $(x * y)(t) = \langle x^*(\tau), y(t-\tau)\rangle$

- A basis for the π-BL space

$$\phi^{(n)}(t) = \text{sinc}(t-n), \quad n \in \mathbb{Z}$$

$$\langle \phi^{(n)}(t), \phi^{(m)}(t)\rangle = \langle \phi^{(0)}(t-n), \phi^{(0)}(t-m)\rangle$$
$$= \langle \phi^{(0)}(t-n), \phi^{(0)}(m-t)\rangle$$
$$= \int_{-\infty}^{\infty} \text{sinc}(t-n)\,\text{sinc}(m-t)\, dt$$
$$= \int_{-\infty}^{\infty} \text{sinc}(\tau)\,\text{sinc}\big((m-n)-\tau\big)\, d\tau$$
$$= (\text{sinc} * \text{sinc})(m-n)$$

now use the convolution theorem knowing that:

$$FT\{\text{sinc}(t)\} = \text{rect}\left(\frac{\Omega}{2\pi}\right)$$

$$(\text{sinc} * \text{sinc})(m-n) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \left[\text{rect}\left(\frac{\Omega}{2\pi}\right)\right]^2 e^{j\Omega(m-n)}\, d\Omega$$
$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\Omega(m-n)}\, d\Omega = \begin{cases} 1, & m=n \\ 0, & \text{otherwise} \end{cases}$$

5.3.c The sampling theorem

- Sampling as a basis expansion

for any $x(t) \in \pi$-BL,

$$\langle \phi^{(n)}(t), x(t)\rangle = \langle \text{sinc}(t-n), x(t)\rangle = \langle \text{sinc}(n-t), x(t)\rangle$$
$$= (\text{sinc} * x)(n)$$
$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} \text{rect}\left(\frac{\Omega}{2\pi}\right) X(j\Omega)\, e^{j\Omega n}\, d\Omega$$
$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\Omega)\, e^{j\Omega n}\, d\Omega$$
$$= x(n)$$

- Sampling as a basis expansion, π-BL

  - Analysis formula: $x[n] = \langle \text{sinc}(t-n), x(t) \rangle$

  - Synthesis formula: $x(t) = \sum_{n=-\infty}^{\infty} x[n] \, \text{sinc}(t-n)$

- Sampling as a basis expansion, Ω_N-BL

  - Analysis formula: $x[n] = \langle \text{sinc}\left(\frac{t-nT_s}{T_s}\right), x(t) \rangle = T_s \, x(nT_s)$

  - Synthesis formula: $x(t) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} x[n] \, \text{sinc}\left(\frac{t-nT_s}{T_s}\right)$

- The Sampling Theorem

  - The space of Ω_N-bandlimited functions is a Hilbert space
  - set $T_s = \pi/\Omega_N$
  - the functions $\phi^{(n)}(t) = \text{sinc}\left(\frac{t-nT_s}{T_s}\right)$ form a basis for the space
  - for any $x(t) \in \Omega_N$-BL, the coefficients in the sinc basis are the (scaled) samples $T_s \, x(nT_s)$

    $\forall x(t) \in \Omega_N$-BL, a sufficient representation is the sequence
    $x[n] = x(nT_s)$

- corollary

  - Ω_N-BL $\subseteq \Omega$-BL, $\forall \Omega \geq \Omega_N$

  $\forall x(t) \in \Omega_N$-BL, a sufficient representation is the sequence
  $x[n] = x(nT_s)$, $\forall T_s \leq \frac{\pi}{\Omega_N}$

- The sampling theorem, in hertz

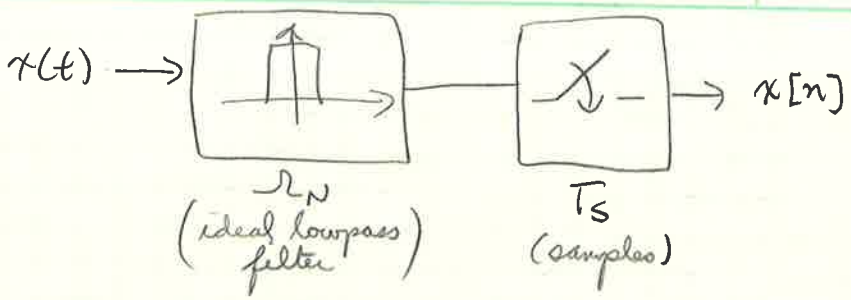
  Any signal $x(t)$ bandlimited to F_N Hz can be sampled with no loss of information using a sampling frequency $F_s \geq 2F_N$ (i.e. a sampling period $T_s \leq \frac{1}{2F_N}$)

5.4 Sampling of nonbandlimited functions

5.4.a Raw sampling

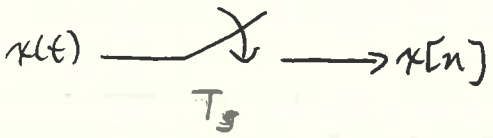
- Sinc Sampling

$$x[n] = \langle \text{sinc}\left(\frac{t-nT_s}{T_s}\right), x(t) \rangle = (\text{sinc}_{T_s} * x)(nT_s)$$



- "Raw" Sampling

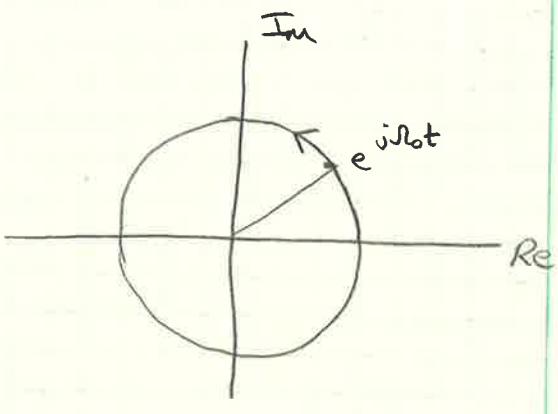
$$x[n] = x(nT_S)$$



- The continuous-time complex exponential

$$x(t) = e^{j\Omega_0 t}$$

- always periodic, period $T = \frac{2\pi}{\Omega_0}$
- all angular speeds are allowed
- FT $\{e^{j\Omega_0 t}\} = 2\pi\delta(\Omega - \Omega_0)$
- bandlimited to Ω_0

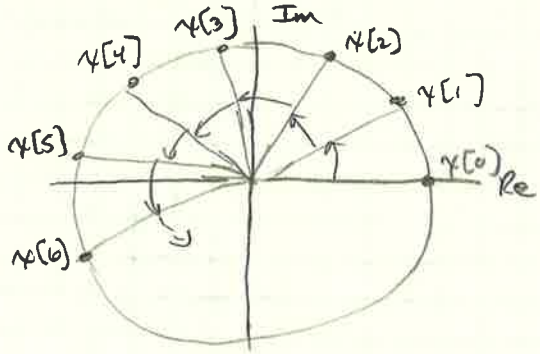


- Raw samples of the continuous-time complex exponential

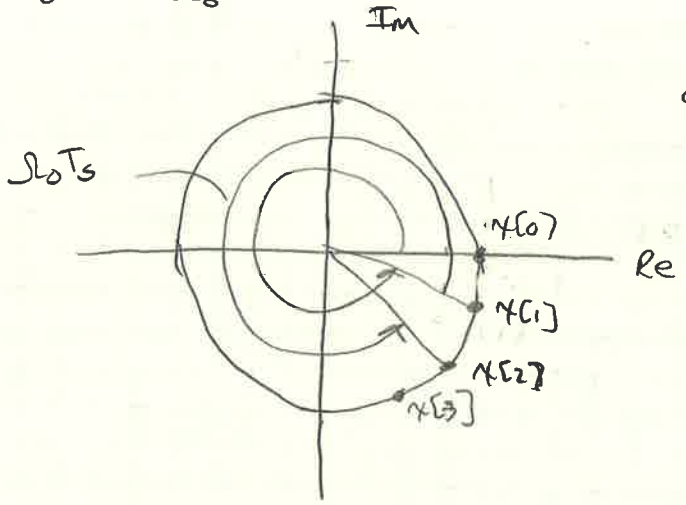
$$x[n] = e^{j\Omega_0 T_S n}$$

- raw samples are snapshots at regular intervals of the rotating point
- resulting digital frequency is $\omega_0 = \Omega_0 T_S$

- When $T_S < \frac{\pi}{\Omega_0}$, $\omega_0 < \pi$...

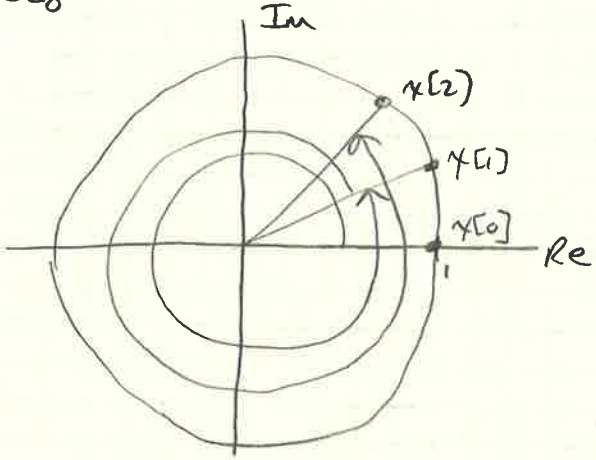


- When $\frac{\pi}{\Omega_0} < T_s < \frac{2\pi}{\Omega_0}$, $\pi < \omega_0 < 2\pi$...



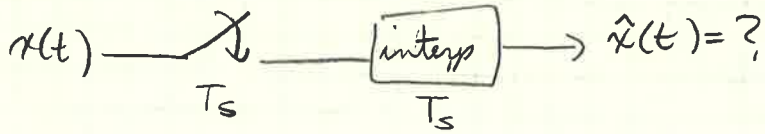
appears to be small, backward steps

- When $T_s > \frac{2\pi}{\Omega_0}$, $\omega_0 > 2\pi$...



Looks like a small frequency

- Aliasing



Example: $x(t) = e^{j\Omega_0 t}$

sampling period	digital frequency	$\hat{x}(t)$
$T_s < \frac{\pi}{\Omega_0}$	$0 < \omega_0 < \pi$	$e^{j\Omega_0 t}$
$\frac{\pi}{\Omega_0} < T_s < \frac{2\pi}{\Omega_0}$	$\pi < \omega_0 < 2\pi$	$e^{j\Omega_1 t}, \Omega_1 = \Omega_0 - \frac{2\pi}{T_s}$
$T_s > \frac{2\pi}{\Omega_0}$	$\omega_0 > 2\pi$	$e^{j\Omega_2 t}, \Omega_2 = \Omega_0 \bmod \left(\frac{2\pi}{T_s}\right)$

5.4.b Sinusoidal Aliasing

Again, with a simple sinusoid and using hertz:

$x(t) = \cos(2\pi F_0 t)$, $F_s = \frac{1}{T_s}$

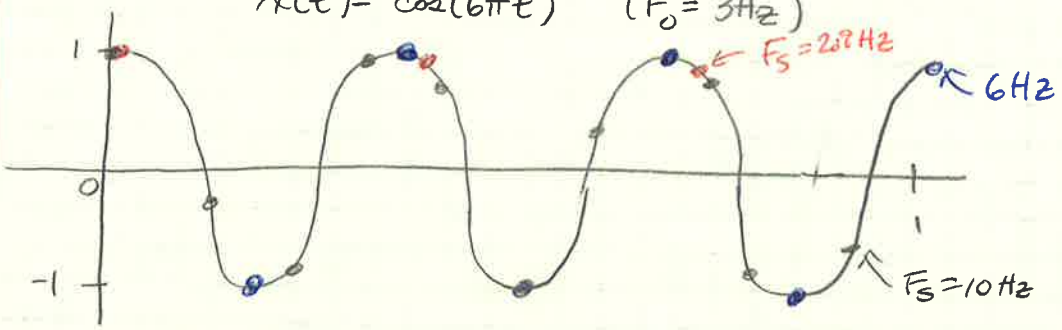
$x[n] = x(nT_s) = \cos(\omega_0 n)$, $\omega_0 = 2\pi\left(\frac{F_0}{F_s}\right)$

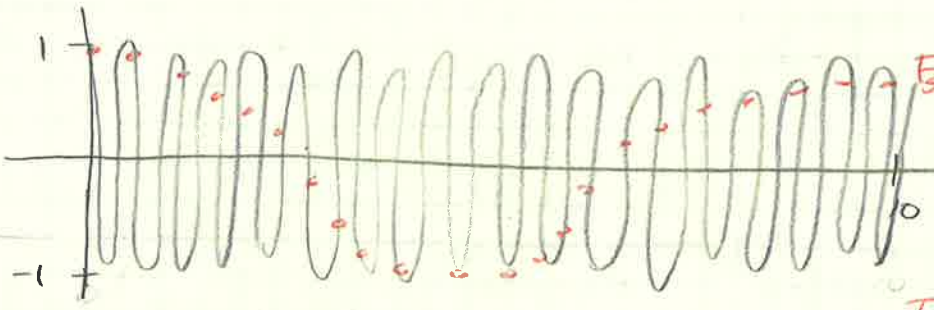
- Sampling a sinusoid

sampling frequency	digital frequency	result
$F_s > 2F_0$	$0 < \omega_0 < \pi$	OK
$F_s = 2F_0$	$\omega_0 = \pi$	max digital frequency: $x[n] = (-1)^n$
$F_0 < F_s < 2F_0$	$\pi < \omega_0 < 2\pi$	negative frequency: $\omega_0 - 2\pi$
$F_s < F_0$	$\omega_0 > 2\pi$	full aliasing: $\omega_0 \bmod 2\pi$

- Aliasing: Sampling a Sinusoid

$x(t) = \cos(6\pi t)$ $(F_0 = 3\text{Hz})$





5.4.c Aliasing for arbitrary spectra

- Raw-sampling an arbitrary signal

$x_c(t) \xrightarrow{T_s} x[n] = x_c(nT_s)$

$X_c(j\Omega) \xrightarrow{T_s} X(e^{j\omega}) = ?$

- Key idea:
  - pick T_s (and set $\Omega_N = \frac{\pi}{T_s}$)
  - pick $\Omega_0 < \Omega_N$

$e^{j\Omega_0 t} \xrightarrow{T_s} e^{j\Omega_0 T_s n}$

add $2\Omega_N t$: $e^{j(\Omega_0 + 2\Omega_N)t} \xrightarrow{T_s} e^{j(\Omega_0 + 2\Omega_N)T_s n}$

$$e^{j(\Omega_0+2\Omega_N)T_s n} = e^{j(\Omega_0 T_s n + 2\Omega_N T_s n)} = e^{j(\Omega_0 T_s n + 2\pi n)}$$

$$= e^{j\Omega_0 T_s n}$$

$$Ae^{j\Omega_0 t} + Be^{j(\Omega_0+2\Omega_N)t} \xrightarrow{T_s} (A+B)e^{j\Omega_0 T_s n}$$

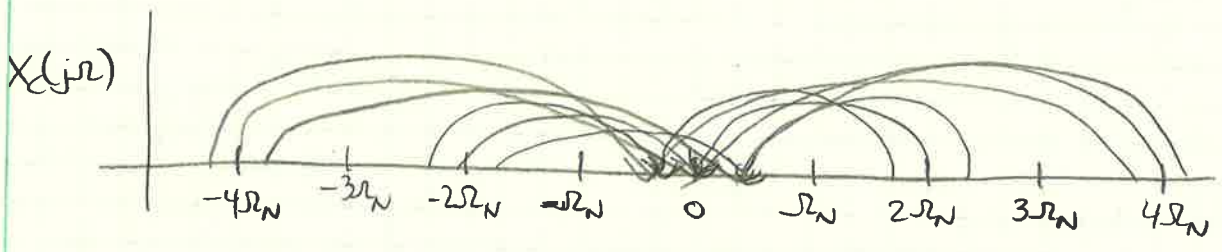
- Spectrum of raw-sampled signals

Start with the inverse Fourier Transform

$$x[n] = x_c(nT_s) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_c(j\Omega)e^{j\Omega n T_s}\,d\Omega$$

Frequencies $2\Omega_N$ apart will be aliased, so split the integration interval

$$x[n] = \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\int_{(2k-1)\Omega_N}^{(2k+1)\Omega_N} X_c(j\Omega)e^{j\Omega n T_s}\,d\Omega$$



With a change of variable and using $e^{j(\Omega+2k\Omega_N)T_s n} = e^{j\Omega T_s n}$:

$$x[n] = \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\int_{-\Omega_N}^{\Omega_N} X_c(j(\Omega-2k\Omega_N))e^{j\Omega n T_s}\,d\Omega$$

$$= \frac{1}{2\pi}\int_{-\Omega_N}^{\Omega_N}\left[\sum_{k=-\infty}^{\infty} X_c(j(\Omega-2k\Omega_N))\right]e^{j\Omega n T_s}\,d\Omega$$

Periodization of the spectrum; define: $\tilde{X}_c(j\Omega) = \sum_{k=-\infty}^{\infty} X_c(j(\Omega-2k\Omega_N))$

so that $x[n] = \frac{1}{2\pi}\int_{-\Omega_N}^{\Omega_N}\tilde{X}_c(j\Omega)e^{j\Omega n T_s}\,d\Omega$

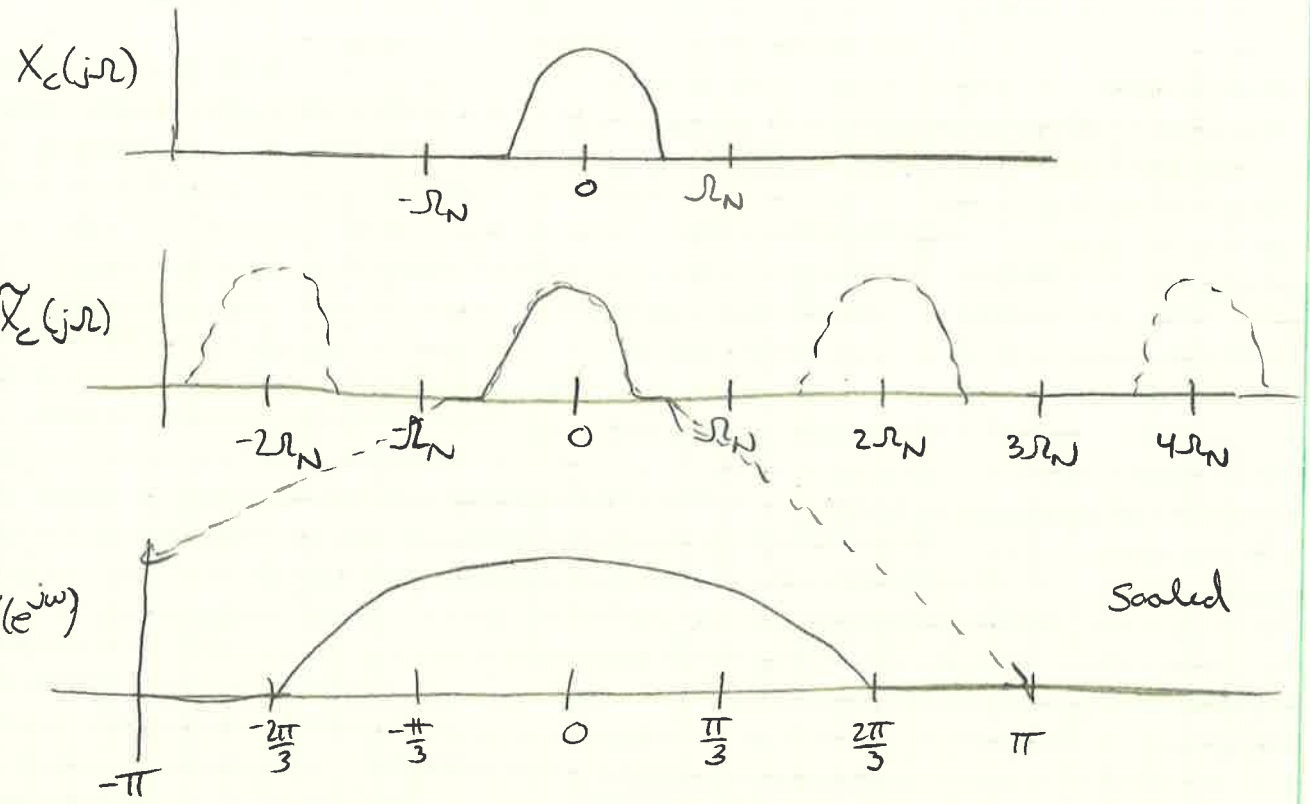
Set $\omega = \Omega T_s$:

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{1}{T_s}\tilde{X}_c\left(j\frac{\omega}{T_s}\right)e^{j\omega n}\,d\omega$$

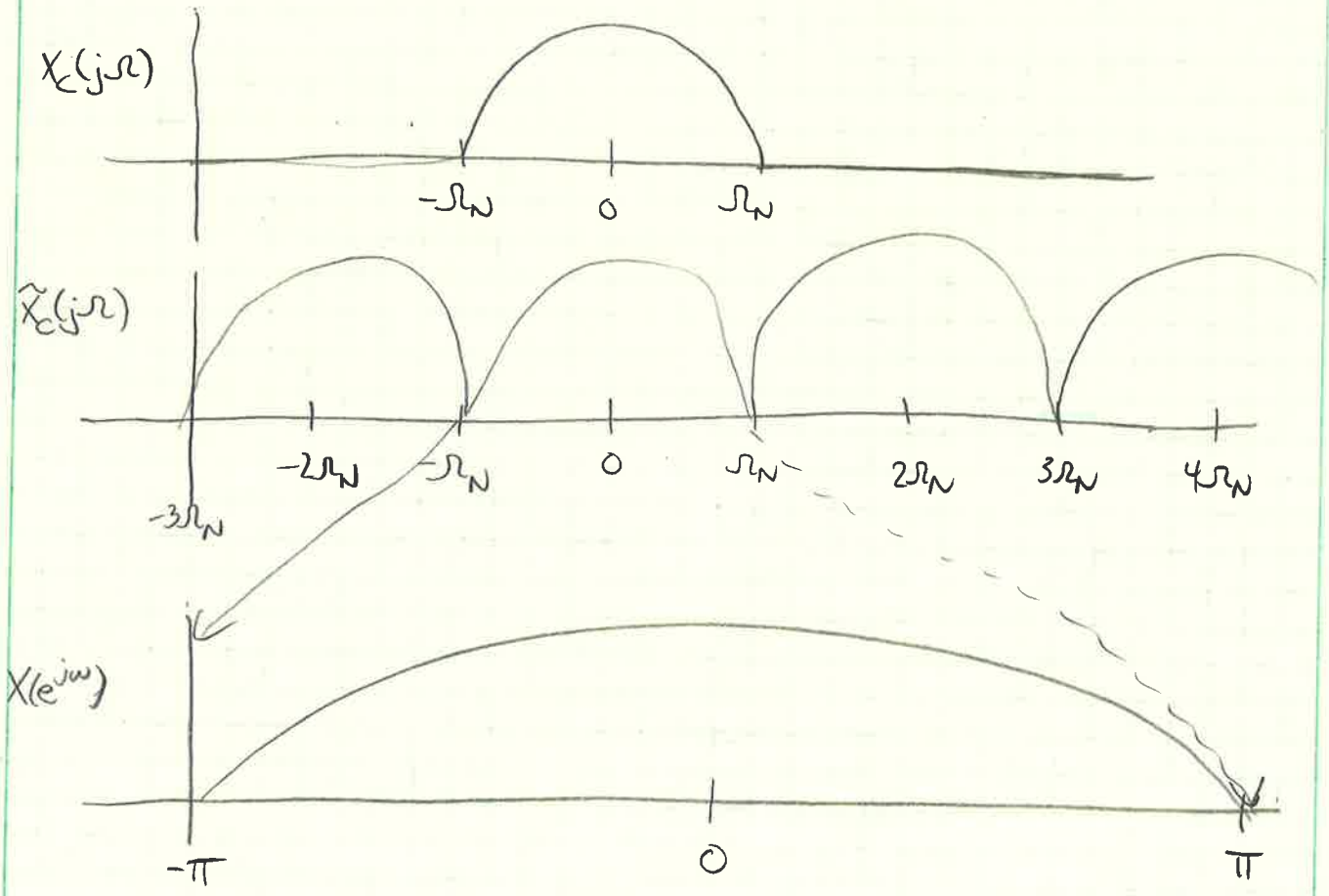
$$= \text{IDTFT}\left\{\frac{1}{T_s}\tilde{X}\left(j\frac{\omega}{T_s}\right)\right\}$$

$$X(e^{j\omega}) = \frac{1}{T_s}\sum_{k=-\infty}^{\infty} X_c\left(j\frac{\omega}{T_s} - j\frac{2\pi k}{T_s}\right)$$

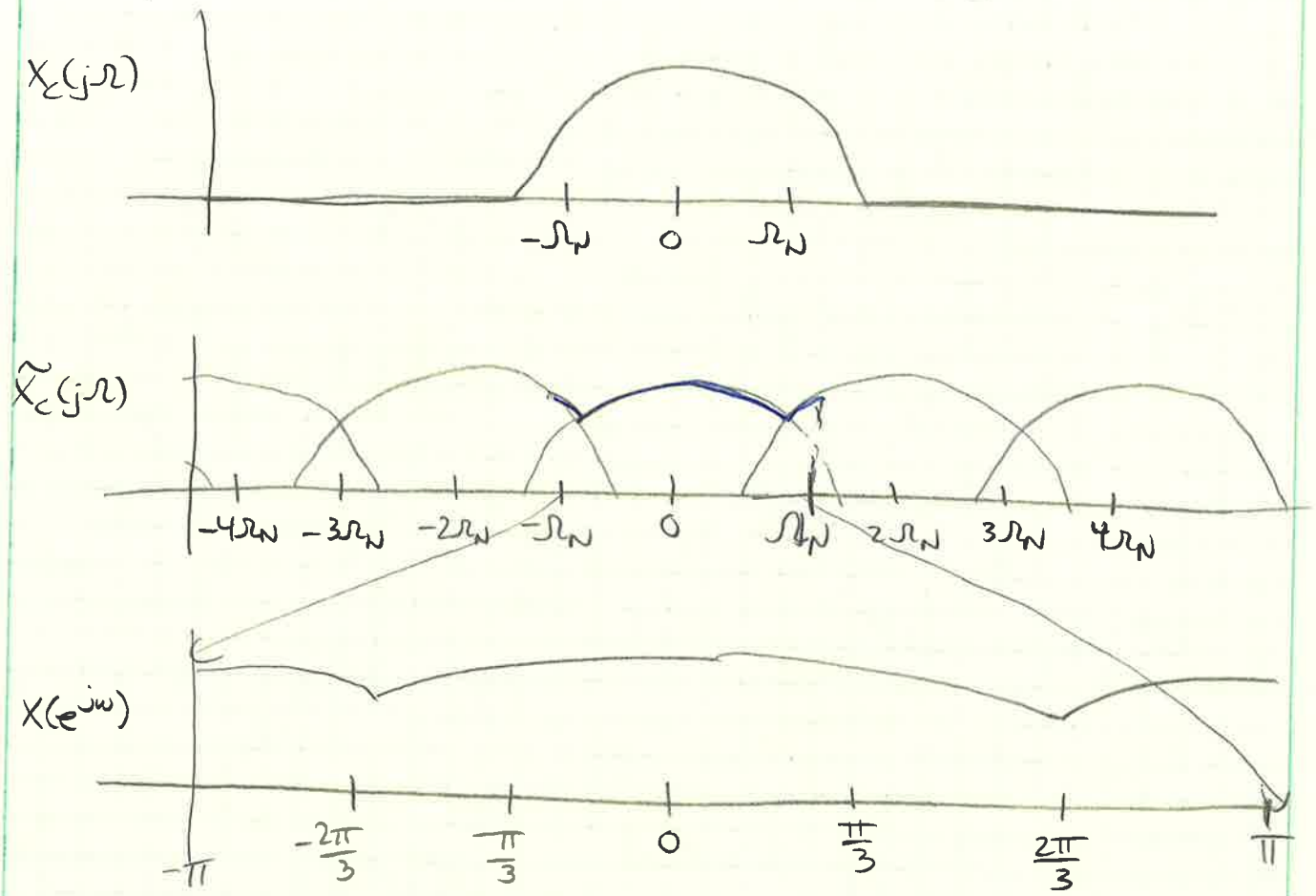
- Example : signal bandlimited to Ω_0 and $\Omega_N > \Omega_0$



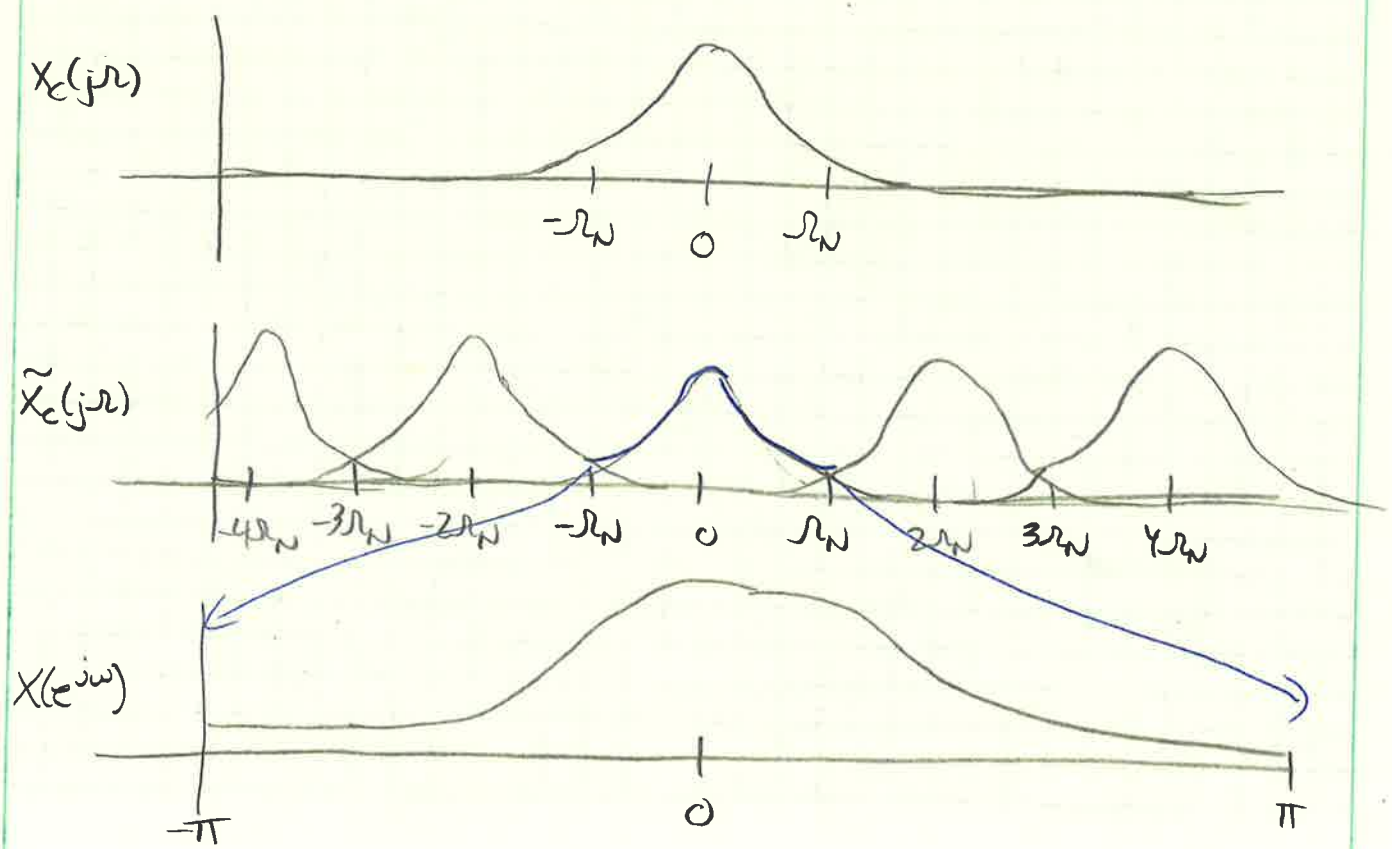
- Example : signal bandlimited to Ω_0 and $\Omega_N = \Omega_0$



- Example: signal bandlimited to Ω_0 and $\Omega_N < \Omega_0$



- Example: non-bandlimited signal



5.4.d Sampling strategies

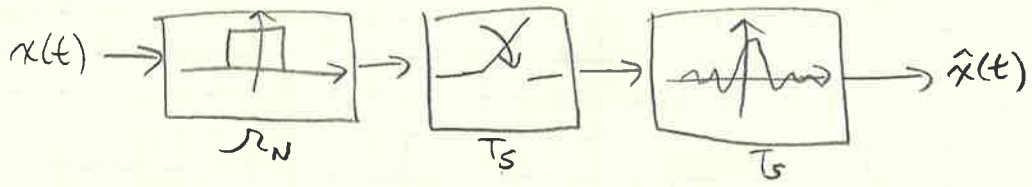
Given a sampling period T_s

- if the signal is bandlimited to π/T_s or less, raw sampling is fine (i.e. equivalent to sinc sampling up to a scaling factor T_s)
- if the signal is not bandlimited, two choices:
  - bandlimit via a lowpass filter in the continuous-time domain before sampling (i.e. sinc sampling)
  - or, raw sample the signal and incur aliasing
- aliasing sounds horrible, so usually we choose to bandlimit in continuous time

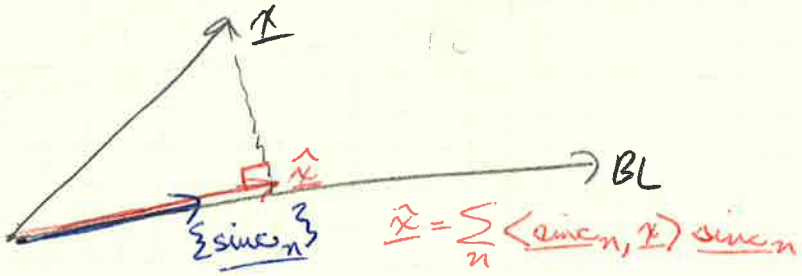
- Sinc Sampling and Interpolation

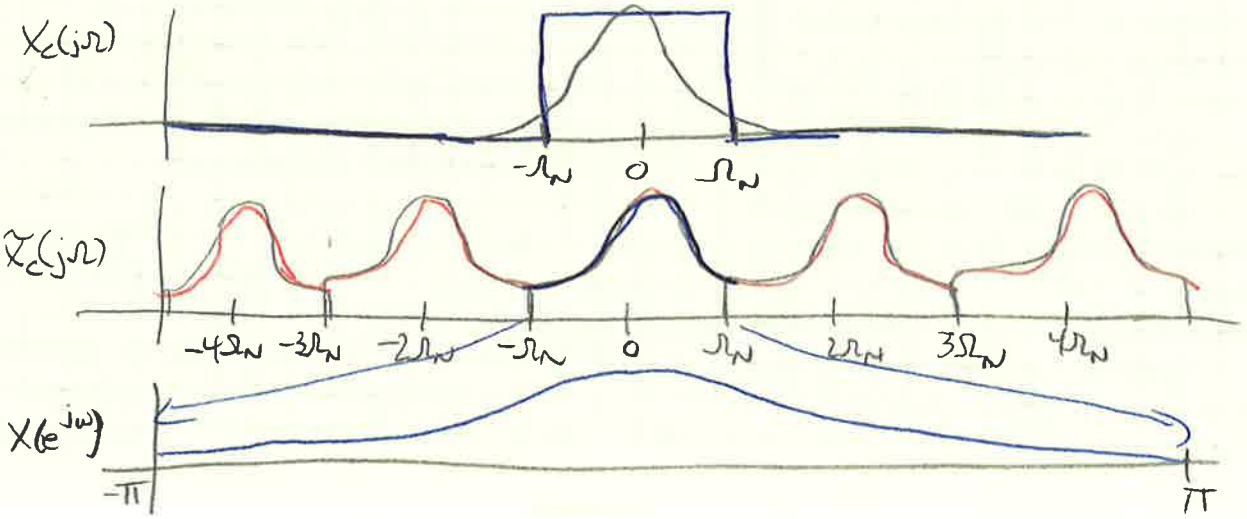
$$\hat{x}[n] = \left\langle \text{sinc}\left(\frac{t - nT_s}{T_s}\right), x(t) \right\rangle = (\text{sinc}_{T_s} * x)(nT_s)$$

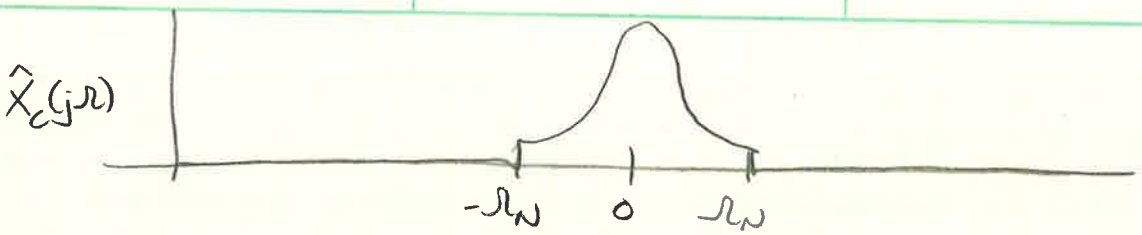
$$\hat{x}(t) = \sum_n x[n] \,\text{sinc}\left(\frac{t - nT_s}{T_s}\right)$$



- Least squares approximation with sinc sampling and interpolation







5.5 Quantization

5.5.a Stochastic Signal Processing

- Deterministic vs. Stochastic

- deterministic signals are known in advance: $x[n] = \sin(0.2n)$
- interesting signals are not known in advance: $s[n] = ?$
- we usually know something, though: $s[n]$ is a speech signal
- stochastic signals can be described probabilistically
- can we do signal processing with random signals? Yes!
- will not develop stochastic signal processing rigorously but give enough intuition to deal with things such as "noise".

- A simple discrete-time random signal generator

For each new sample, toss a fair coin: $x[n] = \begin{cases} 1, & \text{heads} \\ -1, & \text{tails} \end{cases}$

- each sample is independent from all others
- each sample value has 50% probability
- every time we turn on the generator we obtain a different realization of the signal
- we know the "mechanism" behind each instance
- but how can we analyze a random signal?

- Spectral properties?

- let's try with the DFT of a finite set of random samples
- every time it's different; maybe with more data?
- no clear pattern... we need a new strategy

- Averaging

- when faced with random data an intuitive response is to take "averages"
- in probability theory, the average is across realizations and is called expectation (not along the time index)
- for the coin toss signal: $E[x[n]] = -1 \cdot P[n^{th} \text{ toss is tails}] + 1 \cdot P[n^{th} \text{ toss is heads}] = 0$
- so the average value for each sample is zero...

- Averaging the DFT

- ... as a consequence, averaging the DFT will not work
- $E[X[k]] = 0$
- however the signal "moves", so its energy or power must be nonzero

- Energy and power

- the coin-toss signal has infinite energy (see Module 2.1):

$$E_x = \lim_{N\to\infty} \sum_{n=-N}^{N} |x[n]|^2 = \lim_{N\to\infty} (2N+1) = \infty$$

- however it has finite power over any interval:

$$P_x = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2 = 1$$

- Averaging

Let's try to average the DFT's square magnitude, normalized:

- pick an interval of length N
- pick a number of iterations M
- run the signal generator M times and obtain M N-point realizations
- compute the DFT of each realization
- average their square magnitude divided by N

- Power spectral density (PSD)

$$P[k] = \mathbb{E}\left[|X_N[k]|^2/N\right]$$

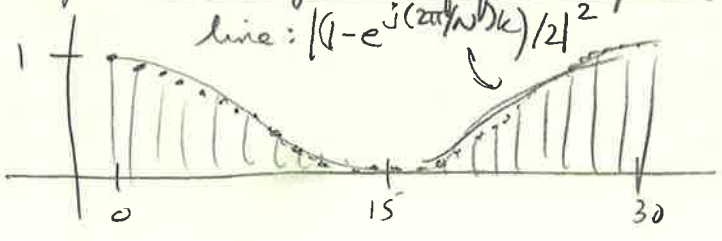
- it looks very much as if $P[k]=1$
- if $|X_N[k]|^2$ tends to the energy distribution in frequency ...
- ... $|X_N[k]|^2/N$ tends to the power distribution (aka density) in frequency
- the frequency-domain representation for stochastic processes is the PSD.

- intuition
  - $P[k]=1$ means that the power is equally distributed over all frequencies
  - I.e., we cannot predict if the signal moves "slowly" or "super-fast"
  - this is because each sample is independent of each other: we could have a realization of all ones or a realization in which the sign changes every other sample or anything in between

- Filtering a random process

- let's filter the random process with a 2-point MA filter
- $y[n] = (x[n]+x[n-1])/2$
- what is the PSD?

- Averaged DFT magnitude of filtered process



$M=5000$

$$H(e^{j\omega}) = \frac{(1-e^{j\omega})}{2}$$

- Filtering a random process

- it looks like $P_y[k] = P_x[k]\,|H[k]|^2$, where $H[k] = \text{DFT}\{h[n]\}$
- can we generalize these results beyond a finite set of samples?

- Stochastic signal processing

- A stochastic process is characterized by its PSD
- it can be shown (see the textbook) that the PSD is

$$P_x(e^{j\omega}) = \text{DTFT}\{r_x[n]\}$$

where $r_x[n] = \mathcal{E}[x[k]\,x[n+k]]$ is the autocorrelation of the process

- for a filtered stochastic process $y[n] = \mathcal{H}\{x[n]\}$, it is

$$P_y(e^{j\omega}) = |H(e^{j\omega})|^2\, P_x(e^{j\omega})$$

- key points:
  - filters designed for deterministic signals still work (in magnitude) in the stochastic case
  - we lose the concept of phase since we don't know the shape of a realization in advance

- Noise

- Noise is everywhere:
  - thermal noise
  - sum of extraneous interferences
  - quantization and numerical errors
  - ...
- We can model noise as a stochastic signal
- The most important noise is white noise

- White noise

- "white" indicates uncorrelated samples
- $r_w[n] = \sigma^2 \delta[n]$
- $P_w(e^{j\omega}) = \sigma^2$
- the PSD is independent of the probability distribution of the single samples (depends only on the variance)
- distribution is important to estimate bounds for the signal
- very often a Gaussian distribution models experimental data the best
- AWGN: additive white Gaussian noise

5.5.b Quantization

- digital devices can only deal with integers (b bits per sample)
- we need to map the range of a signal onto a finite set of values
- irreversible loss of information → quantization noise

- Quantization scheme

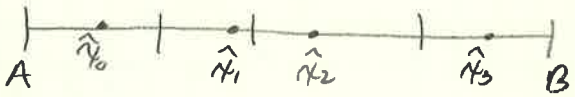
$$x[n] \in \mathbb{C} \longrightarrow \boxed{Q\{\cdot\}} \longrightarrow \hat{x}[n] \in \mathbb{Z}$$

- Several factors at play:
  - storage budget (bits per sample)
  - storage scheme (fixed point, floating point)
  - properties of the input
    - range
    - probability distribution

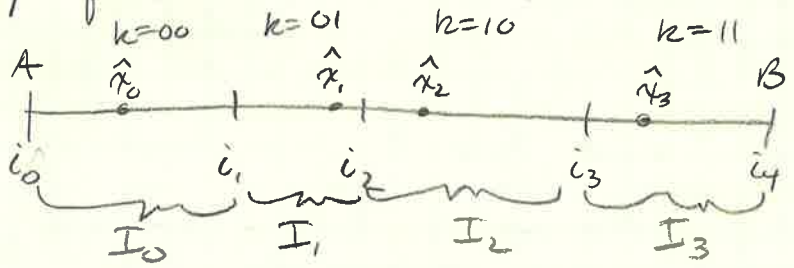
- Scalar quantization

- The simplest quantizer:
  - each sample is encoded individually (hence scalar)
  - each sample is quantized independently (memoryless quantization)
  - each sample is encoded using R bits

- Assume input signal is bounded: $A \le x[n] < B$ for all n:
  - each sample quantized over 2^R possible values $\Rightarrow 2^R$ intervals (R bps)
  - each interval associated to a quantization value



Example for $R=2$:



- what are the optimal interval boundaries i_k?
- what are the optimal quantization values $\hat{x}_k$?

- Quantization Error

$$e[n] = Q\{x[n]\} - x[n] = \hat{x}[n] - x[n]$$

- model $x[n]$ as a stochastic process
- model error as a white noise sequence:
  - error samples are uncorrelated
  - all error samples have the same distribution
- we need statistics of the input to study the error

- Uniform quantization

- simple but very general case
- range is split into 2^R equal intervals of width $\Delta = (B-A)2^{-R}$

$R = 2$ gives four equally sized intervals

- Mean Square Error is the variance of the error signal:

$$\sigma_e^2 = \mathbb{E}\left[\left|Q\{x[n]\} - x[n]\right|^2\right]$$

$$= \int_A^B f_x(\tau)\left(Q\{\tau\} - \tau\right)^2 d\tau$$

$$= \sum_{k=0}^{2^R-1} \int_{I_k} f_x(\tau)(\hat{x}_k - \tau)^2 d\tau$$

- error depends on the probability distribution of the input

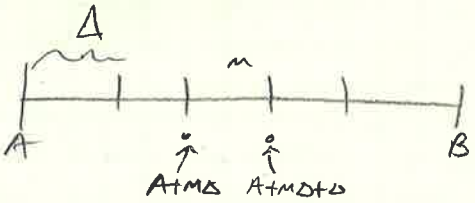
- Uniform quantization of uniform input

$$f_x(\tau) = \frac{1}{B-A} \Rightarrow \sigma_e^2 = \sum_{k=0}^{2^R-1} \int_{I_k} \frac{(\hat{x}_k - \tau)^2}{B-A} d\tau$$

Let's find the optimal quantization point by minimizing the error

$$\frac{\partial \sigma_e^2}{\partial \hat{x}_m} = \frac{\partial}{\partial \hat{x}_m} \sum_{k=0}^{2^R-1} \int_{I_k} \frac{(\hat{x}_k - \tau)^2}{B-A} d\tau = \int_{I_m} \frac{2(\hat{x}_m - \tau)}{B-A} d\tau$$

$$= \frac{(\hat{x}_m - \tau)^2}{B-A} \Bigg|_{A+m\Delta}^{A+m\Delta+\Delta}$$

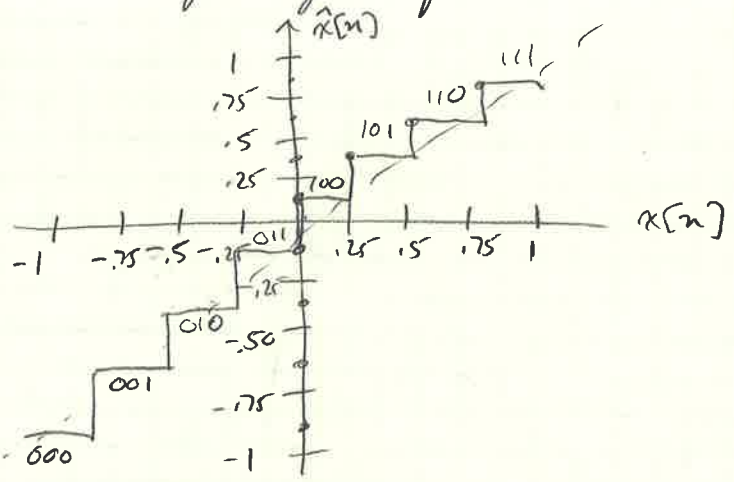


Minimize the error:

$$\frac{\partial \sigma_e^2}{\partial \hat{x}_m} = 0 \quad \text{for} \quad \hat{x}_m = A + m\Delta + \frac{\Delta}{2}, \text{ i.e. optimal quantization point is the interval's midpoint, for all intervals}$$

- Uniform 3-Bit quantization function

$R=3$



- Uniform quantization of uniform input

Quantizer's mean square error (MSE):

$$\sigma_e^2 = \sum_{k=0}^{2^R-1} \int_{A+k\Delta}^{A+k\Delta+\Delta} \frac{\left(A+k\Delta+\frac{\Delta}{2}-x\right)^2}{B-A}\,dx$$

$$= 2^R \int_0^{\Delta} \frac{\left(\frac{\Delta}{2}-x\right)^2}{B-A}\,dx = \frac{\Delta^2}{12}, \quad \Delta = \frac{B-A}{2^R}$$

- Error analysis

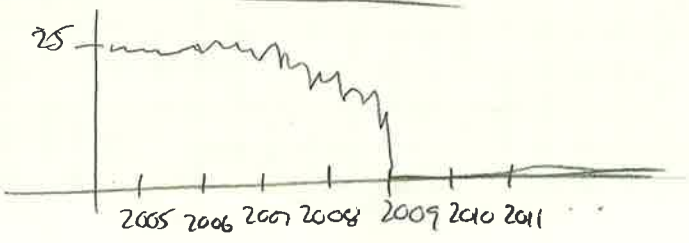
- error energy: $\sigma_e^2 = \frac{\Delta^2}{12}$, $\Delta = \frac{B-A}{2^R}$
- signal energy: $\sigma_x^2 = \frac{(B-A)^2}{12}$
- signal to noise ratio (SNR): $SNR = 2^{2R} \left(= \frac{\sigma_x^2}{\sigma_e^2}\right)$
  - in dB: $SNR_{dB} = 10 \log_{10} 2^{2R} \approx 6R$ dB

- The "6 dB/bit" rule of thumb

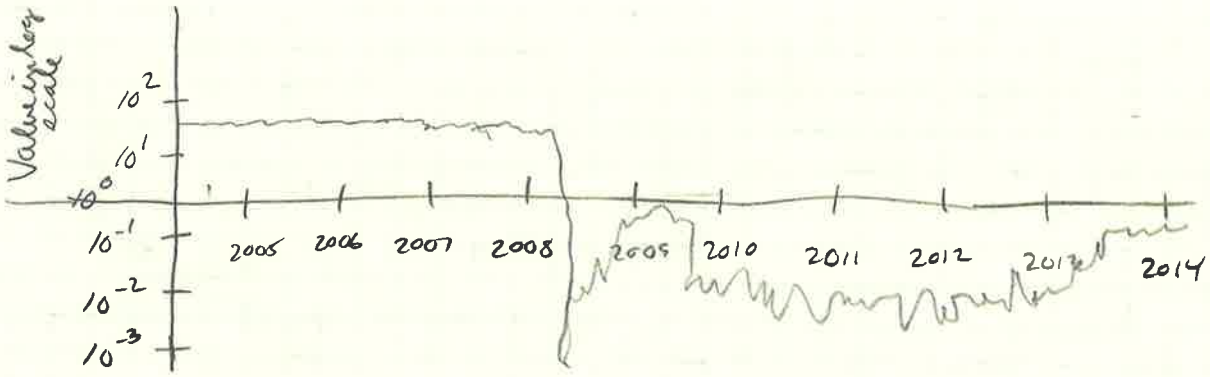
- a compact disk has 16 bits/sample: max SNR = 96 dB
- a DVD has 24 bits/sample: max SNR = 144 dB

Signal of the Day: Lehman Brothers

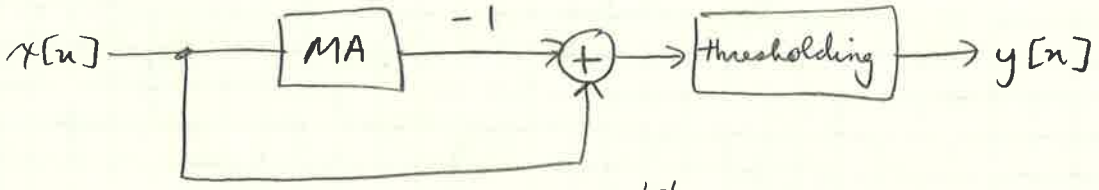
Lehman Brothers Stock Price



- Distressed Securities

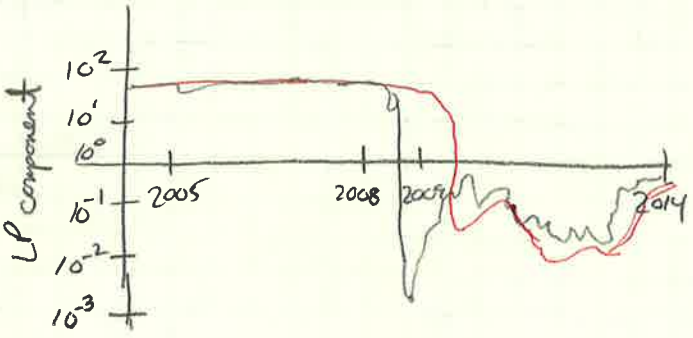


- Spike Detection

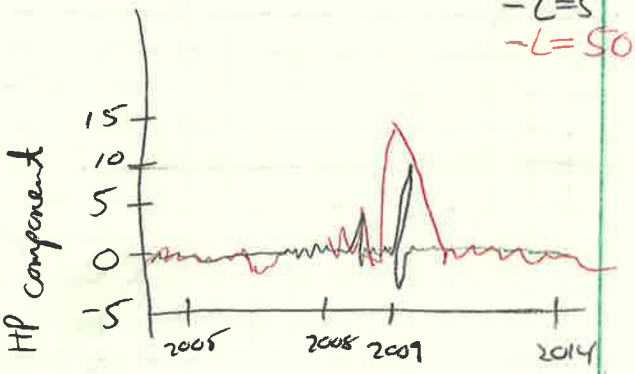


- Low pass component: $w[n] = \frac{1}{L} \sum_{m=0}^{L-1} x(n-m)$
- High pass component: $x[n] - w[n]$

- Spike Detector



- L=5
- L=50



- Stochastic Signal Processing

- Limitations from this approach
  - trade-off between sensitivity to noise and reactiveness
  - backward looking

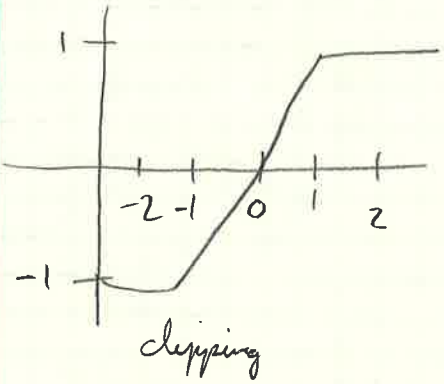
- Statistical signal processing
  - introduce randomness in modelling
  - Certain SP tools we have studied
  - statistical signal processing, new tools to handle randomness

5.5c* Clipping, saturation and companding

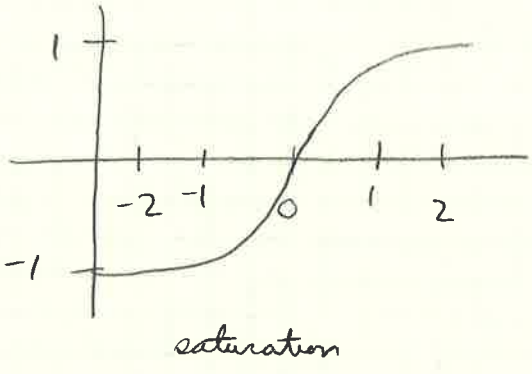
- Other quantization errors

If input is not bounded to $[A,B]$:
- clip samples to $[A,B]$: linear distortion (can be put to good use in guitar effects!)
- smoothly saturate input: this simulates the saturation curves of analog electronics

- Clipping vs saturation



clipping



saturation

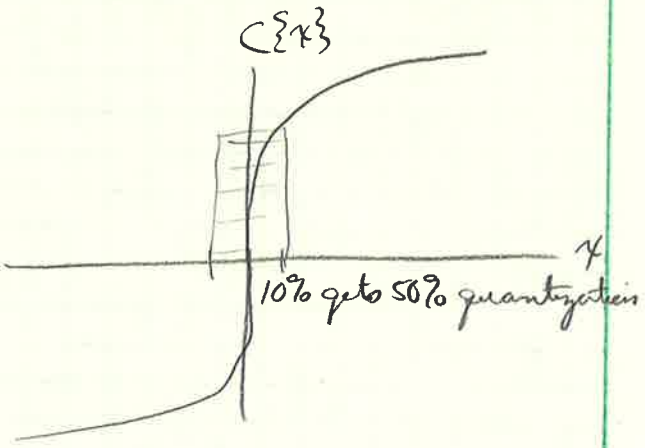
If input is not uniform:
- use uniform quantizer and accept increased error. For instance, if input is Gaussian:

$$\sigma_e^2 = \frac{\sqrt{3}}{2}\pi\sigma^2\Delta^2 > \frac{\Delta^2}{12}$$

- design optimal quantizer for input distribution, if known (Lloyd-Max)
- use "companders"

- μ-law compander

$$C\{x[n]\} = \operatorname{sgn}(x[n])\frac{\ln(1+\mu|x[n]|)}{\ln(1+\mu)}$$



5.6* Practical sampling and interpolation

- From Continuous to Discrete Time

$$x(t) \to x[n]$$

ideally

$$x[n] = \langle x(t), \operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right) \rangle$$

$$\Omega_N = \frac{\pi}{T_s}$$

$$X(e^{j\omega}) = X\left(j \frac{\Omega_N}{\pi} (\omega \bmod 2\pi)\right)$$

in practice

$$x[n] = x(nT_s)$$

$$X(e^{j\omega}) = \frac{\pi}{\Omega_N} \sum_{k=-\infty}^{\infty} X_c\left(j \frac{\pi}{\Omega_N} \omega - 2jk\Omega_N\right)$$

$$x[n] \to x(t)$$

ideally

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \operatorname{sinc}\left(\frac{t - nT_s}{T_s}\right)$$

$$X(j\Omega) = \frac{\pi}{\Omega_N} X(e^{j\pi\Omega/\Omega_N}) \operatorname{rect}\left(\frac{\Omega}{2\Omega_N}\right)$$

in practice

$$x(t) = \sum_{n=-\infty}^{\infty} x[n]\, i\left(\frac{t - nT_s}{T_s}\right)$$

(i: interpolation fcn)

$$X(j\Omega) = ?$$

- Practical interpolation

$$x(t) = \sum_{n=-\infty}^{\infty} x[n]\, i\left(\frac{t - nT_s}{T_s}\right)$$

- Spectral representation (1)

$$X(j\Omega) = \int_{-\infty}^{\infty} x(t) e^{-j\Omega t} dt$$

$$= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[n]\, i\left(\frac{t - nT_s}{T_s}\right) e^{-j\Omega t} dt$$

$$= \sum_{n=-\infty}^{\infty} x[n] \int_{-\infty}^{\infty} i\left(\frac{t - nT_s}{T_s}\right) e^{-j\Omega t} dt \qquad t \to \frac{t - nT_s}{T_s}$$

$$= T_s \sum_{n=-\infty}^{\infty} x[n]\, I(jT_s\Omega)\, e^{-jnT_s\Omega}$$

$$= \frac{\pi}{\Omega_N} I\left(j\pi \frac{\Omega}{\Omega_N}\right) \sum_{n=-\infty}^{\infty} x[n] e^{-j\pi \frac{\Omega}{\Omega_N} n} \qquad [-\Omega_N, \Omega_N] \leftrightarrow [-\pi, \pi]$$

$$= \frac{\pi}{\Omega_N} I\left(j\pi \frac{\Omega}{\Omega_N}\right) X(e^{j\pi \frac{\Omega}{\Omega_N}})$$

- Sinc interpolation

$$i(t) = \text{sinc}(t)$$

$$I(j\Omega) = \text{rect}\left(\frac{\Omega}{2\pi}\right) \qquad (T_S = 1)$$

$$X(j\Omega) = \frac{\pi}{\Omega_N} I\left(\frac{j\pi\Omega}{\Omega_N}\right) X\left(e^{j\pi\Omega/\Omega_N}\right)$$

$$= \frac{\pi}{\Omega_N} \text{rect}\left(\frac{\Omega}{2\Omega_N}\right) X\left(e^{j\pi\Omega/\Omega_N}\right)$$

- Zero-order hold

$$i(t) = \text{rect}(t) \longleftrightarrow I(j\Omega) = \text{sinc}\left(\frac{\Omega}{2\pi}\right)$$

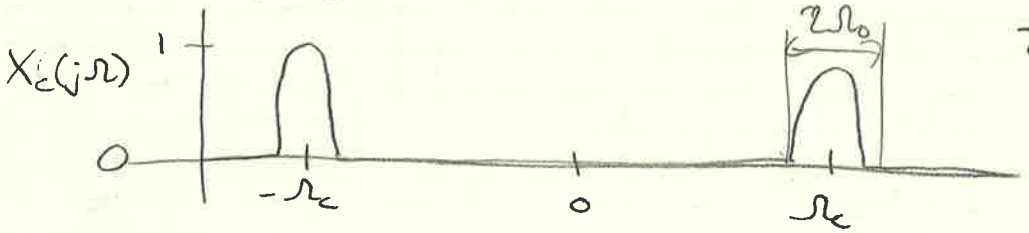
$$X(j\Omega) = \frac{\pi}{\Omega_N} I\left(j\pi\Omega/\Omega_N\right) X\left(e^{j\pi\Omega/\Omega_N}\right)$$

$$= \frac{\pi}{\Omega_N} \text{sinc}\left(\frac{\Omega}{2\Omega_N}\right) X\left(e^{j\pi\Omega/\Omega_N}\right)$$

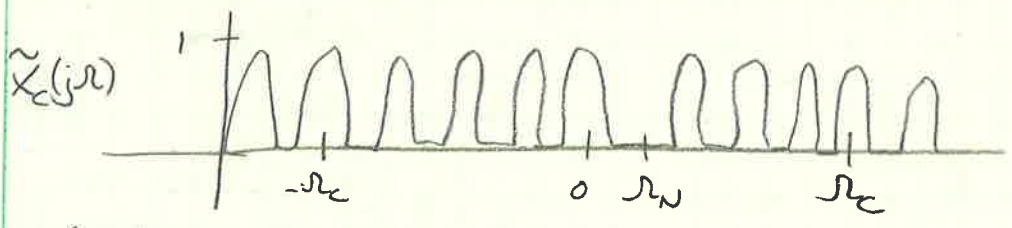
5.7* Bandpass sampling

- Sampling theorem gives a sufficient condition
  - in theory, $\Omega_N > \Omega_{max}$
  - what if signal is bandpass?

- Bandpass sampling



Typically a result of a modulation

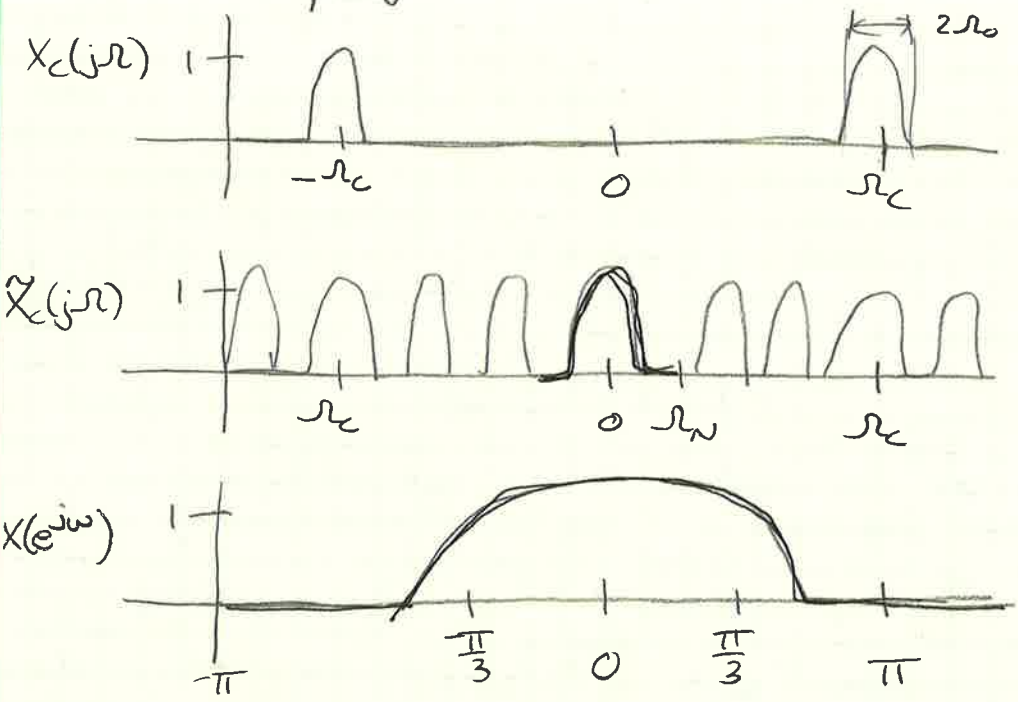


If sampling freq is chosen correctly, the information can be preserved

- Bandpass sampling conditions

- Bandpass signal: $X(j\Omega) = 0$, $|\Omega - \Omega_c| > \Omega_0$
- no alias requires at least $\Omega_N \geq \Omega_0$
- Baseband condition: $\Omega_N = \Omega_c / k$ for some $k \in \mathbb{N}$
- for $T_S = \frac{\pi}{\Omega_N}$, $x[n] = x(nT_S)$ is an alias-free, baseband DT version of $x(t)$

- Baseband sampling



$\Omega_N > \Omega_0$

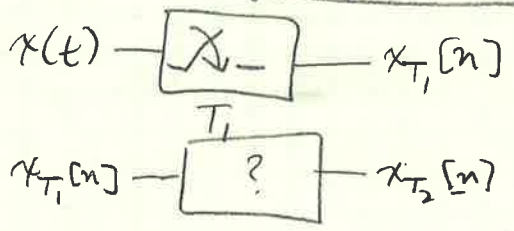
$\Omega_N = \frac{\Omega_c}{k}$

$k = 4$

- Example: AM Channel

- AM radio band: 500kHz to 1.6MHz
- channel width is 9kHz, i.e. $f_0 = 4.5$ kHz
- take a channel at $f_c = 1.5$ MHz
- in theory: $F_S \geq 2 \cdot 1{,}504{,}500$ Hz, $T_S < 10^{-6}$ seconds!
  - antialias: $F_S \geq 2f_0 \Rightarrow F_S \geq 9$kHz
- baseband: $kF_S/2 = 1500$kHz
- pick $k = 300$
  - $F_S = 10$kHz, $T_S = 10^{-4}$ seconds

5.8* Multirate signal processing



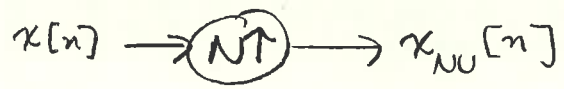
1) $T_1 = NT_2$

2) $T_2 = MT_1$

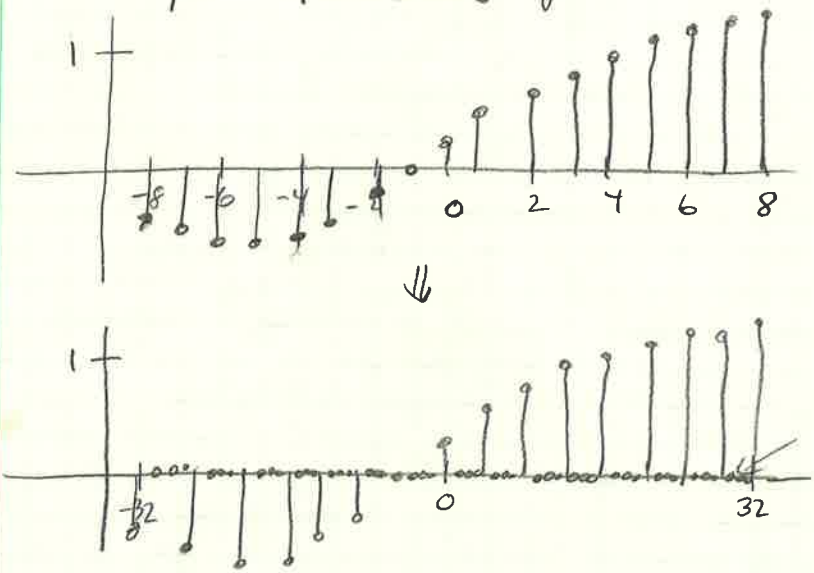
$M, N \in \mathbb{Z}$

- Upsampling

$$x_{NU}[n] = \begin{cases} x[k], & n = kN, \ k \in \mathbb{Z} \\ 0, & \text{otherwise} \end{cases}$$



- Example: upsampling by 4



3 zeros in between

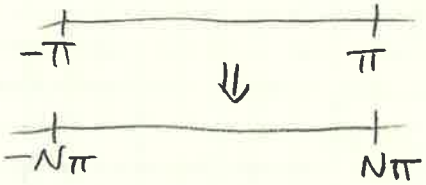
- Spectral representation

$$X_{NU}(z) = \sum_{k=-\infty}^{\infty} x_{NU}[k] z^{-k}$$

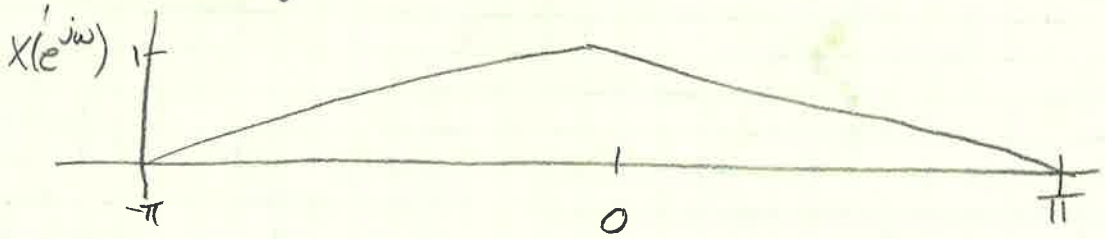
$$= \sum_{k=-\infty}^{\infty} x[k] z^{-Nk}$$

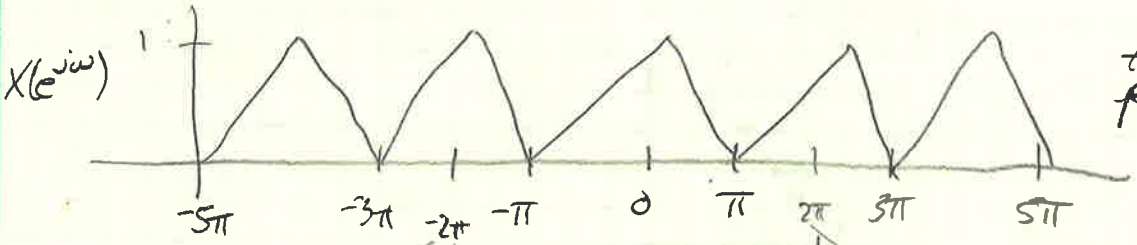
$$= X(z^N)$$

$$X_{NU}(e^{j\omega}) = X(e^{j\omega N})$$

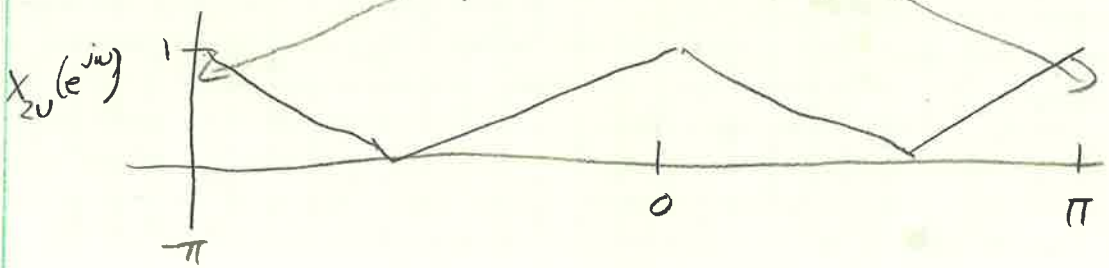


- Upsampling by 2

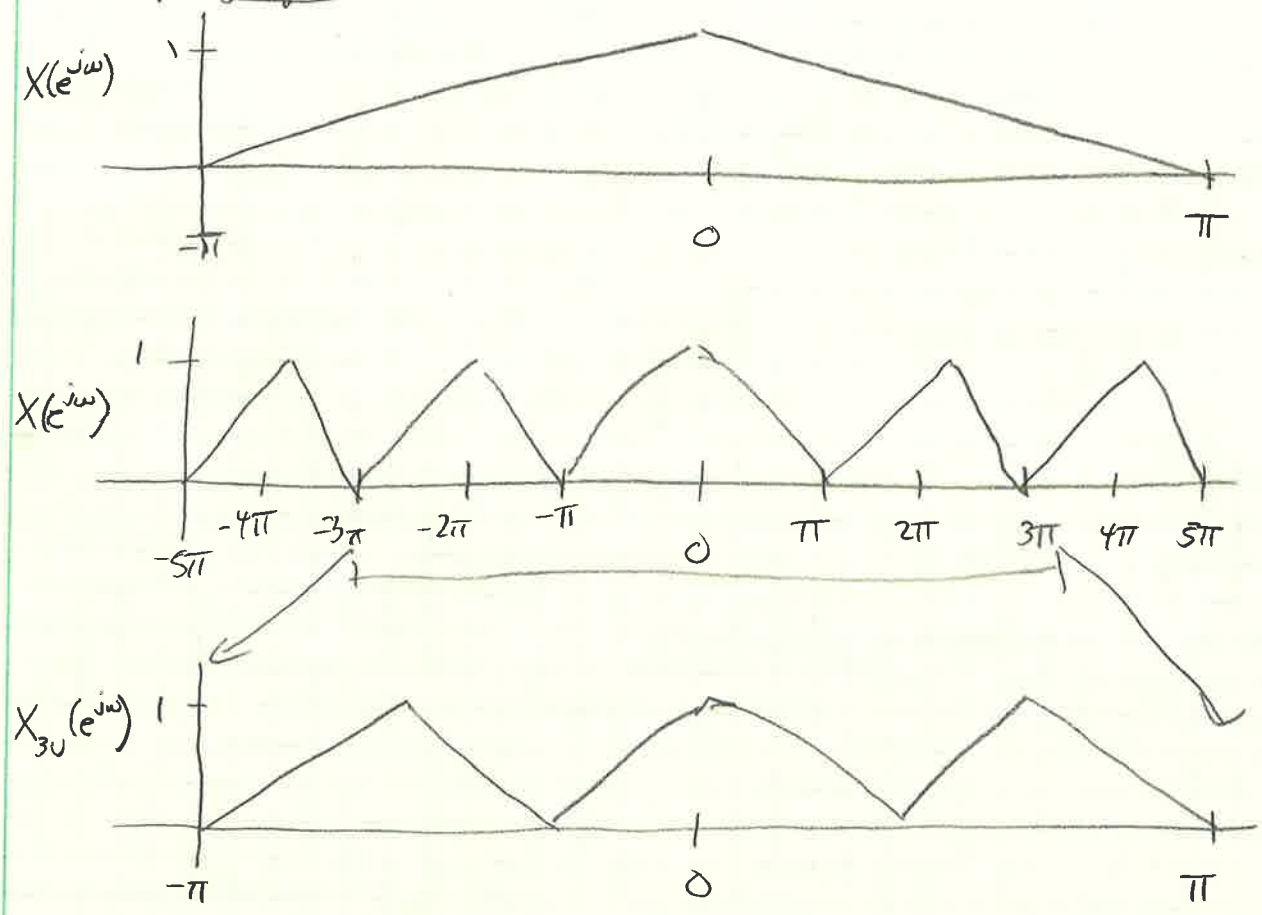




to show periodicity



- Upsampling by 3

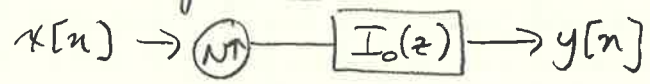


- Upsampling: what we don't like

  - in the time domain: zeros between nonzero samples are not "natural"
  - in the frequency domain: extra replicas of the spectrum; can we get rid of them?

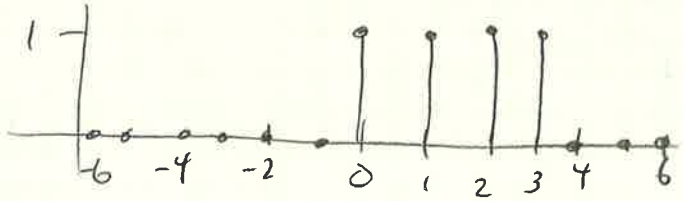
the two problems are the same!

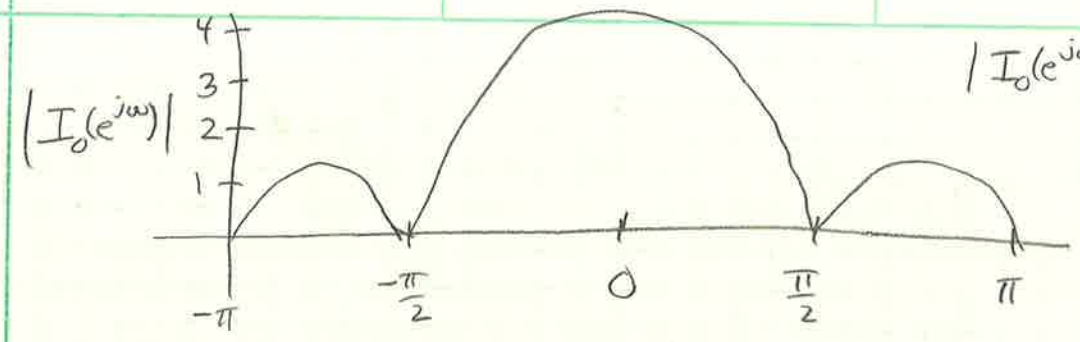
- Zero-order interpolator



Zero-order interpolator for 4-upsampling

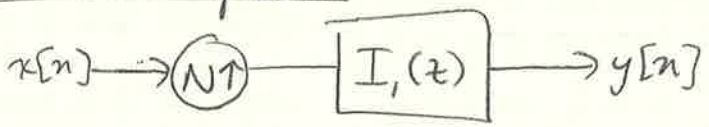
$$i_0[n] = u[n] - u[n-4]$$





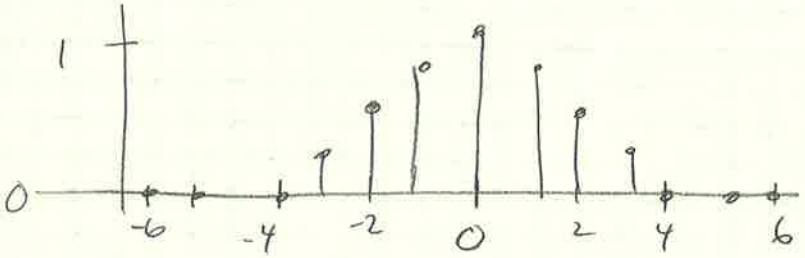
$$|I_0(e^{j\omega})| = \left|\frac{\sin(\frac{\omega}{2}N)}{\sin(\frac{\omega}{2})}\right|, N=4$$

- First-order interpolator

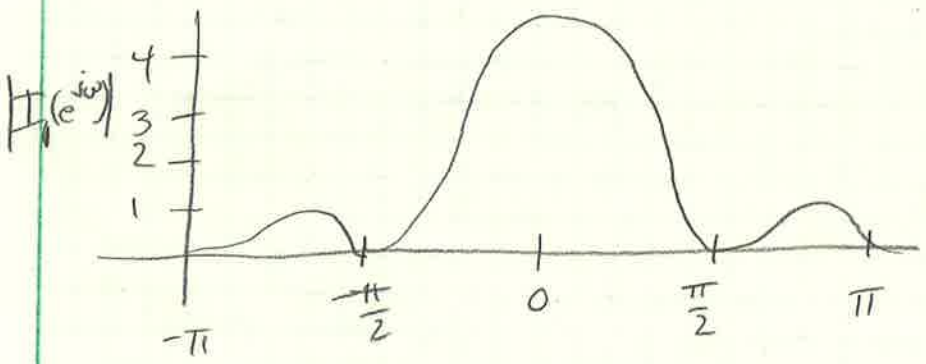


- first-order interpolator for 4-upsampling

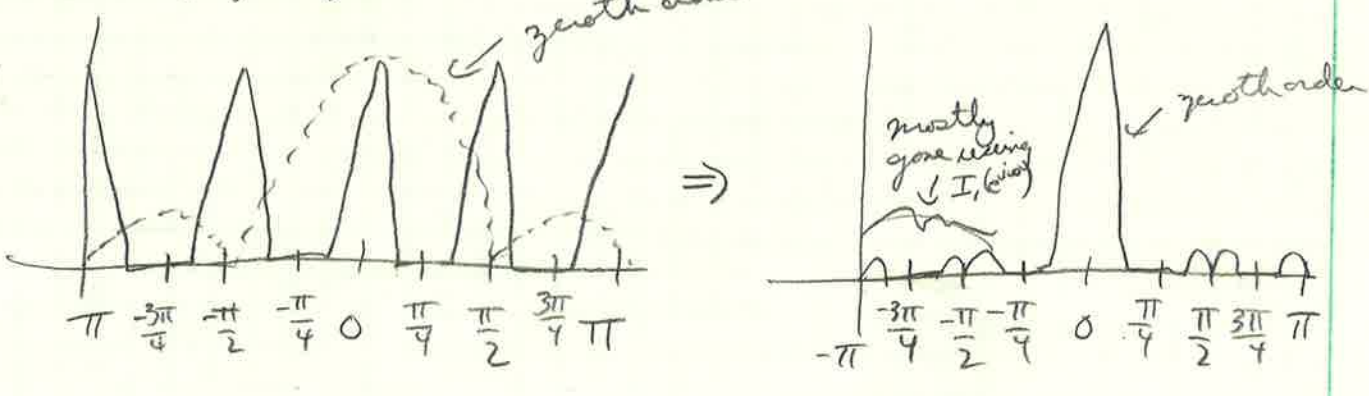
$$i_1[n] = (i_0[n] * i_0[n])/N$$



$$|I_1(e^{j\omega})| = |I_0(e^{j\omega})|^2/N \quad , N=4$$



in the frequency domain

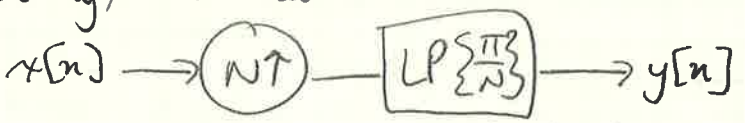


- ideal digital interpolator

$$H(e^{j\omega}) = \text{rect}\left(\frac{\omega N}{2\pi}\right)$$

$$h[n] = \frac{1}{N}\,\text{sinc}\left(\frac{n}{N}\right)$$

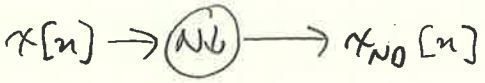
in reality, we can use



↑ discrete-time lowpass filter

- Downsampling

$$x_{ND}[n] = x[nN]$$



retain 1 sample out of every N

- spectral representation

$$X_{ND}(z) = \sum_{k=-\infty}^{\infty} x[kN]\,z^{-k}$$

$$A(z) = \sum_{k=-\infty}^{\infty} x[kN]\,z^{-kN}$$

$$X_{ND}(z) = A(z^{1/N})$$

$$A(z) = \sum_{k=-\infty}^{\infty} \xi[k]\,x[k]\,z^{-k}, \quad \xi[n] = \begin{cases} 1, & n = kN \\ 0, & \text{otherwise} \end{cases}$$

- Remember the roots of unity!

$$\xi[n] = \frac{1}{N}\sum_{m=0}^{N-1} e^{j\frac{2\pi}{N}mn}$$

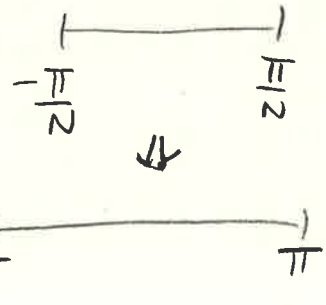
$$A[z] = \sum_{k=-\infty}^{\infty} \xi[k]\,x[k]\,z^{-k}$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\sum_{k=-\infty}^{\infty} x[k]\,e^{j\frac{2\pi}{N}mk}\,z^{-k}$$

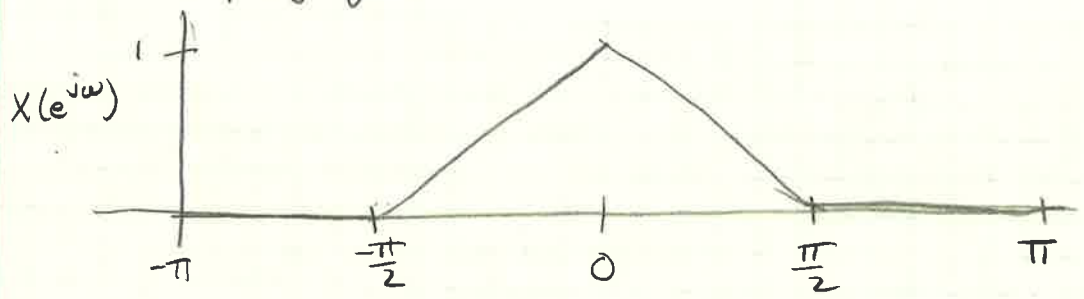
$$= \frac{1}{N}\sum_{m=0}^{N-1} X\left(e^{-j\frac{2\pi}{N}m} z\right)$$

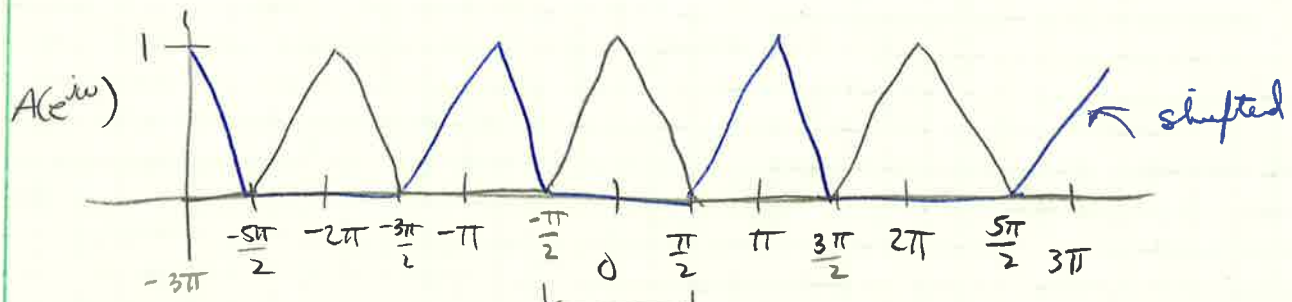
$$X_{ND}(z) = A(z^{1/N}) = \frac{1}{N}\sum_{m=0}^{N-1} X\left(e^{-j\frac{2\pi}{N}m} z^{1/N}\right)$$

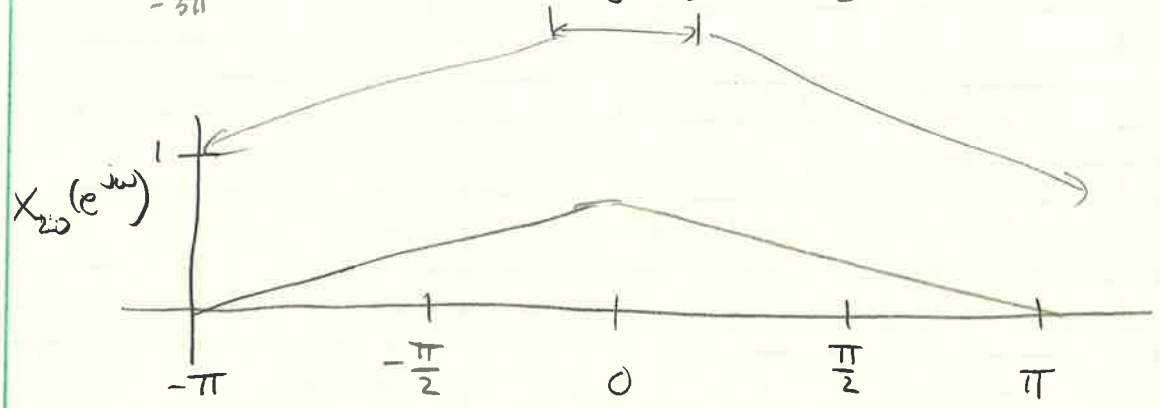
$$X_{ND}(e^{j\omega}) = \frac{1}{N}\sum_{m=0}^{N-1} X\left(e^{j\left(\frac{\omega - 2\pi m}{N}\right)}\right)$$



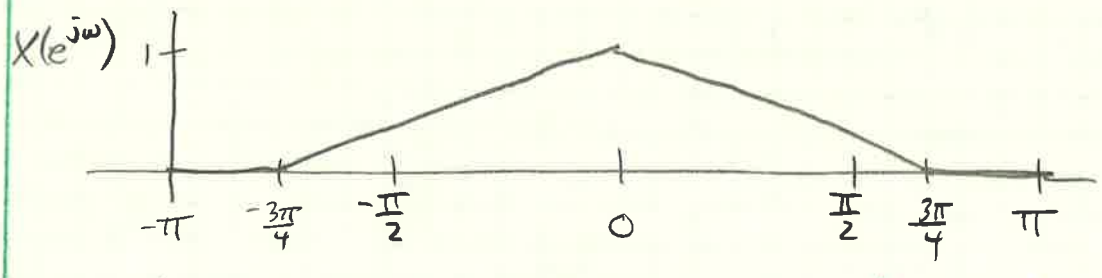
- Downsampling by 2

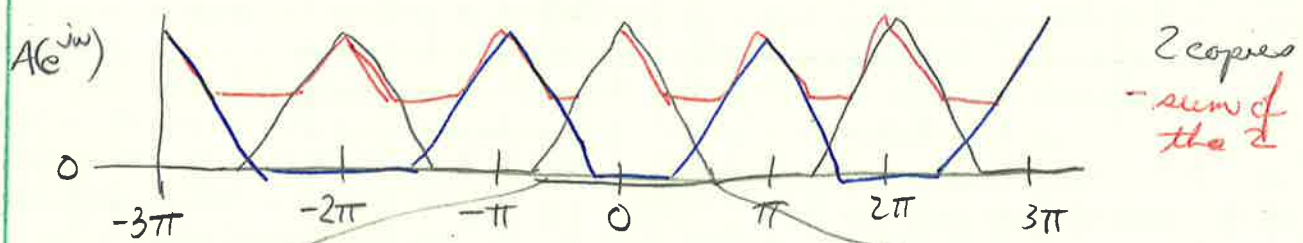


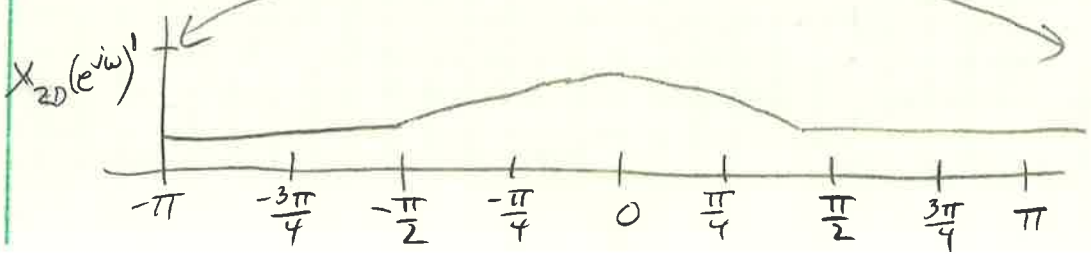




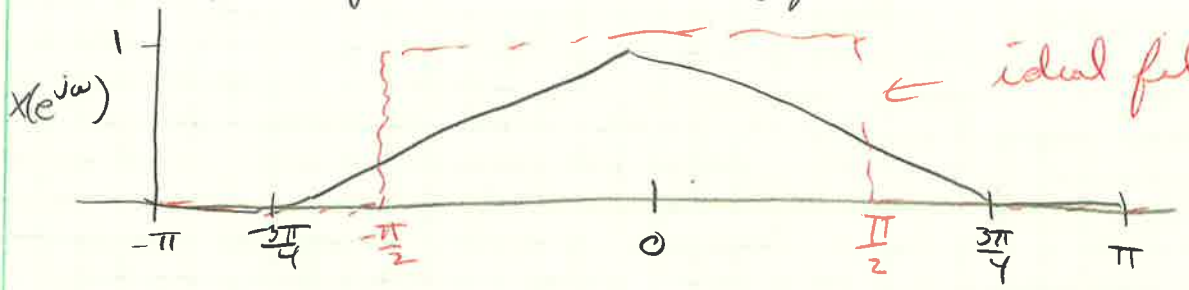
- Downsampling by 2 with aliasing

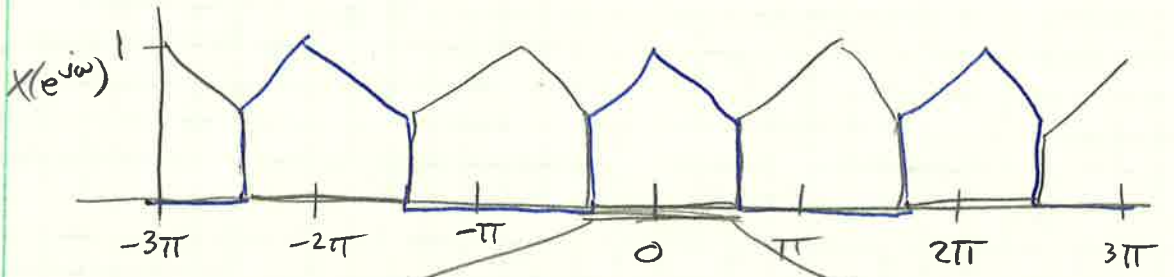




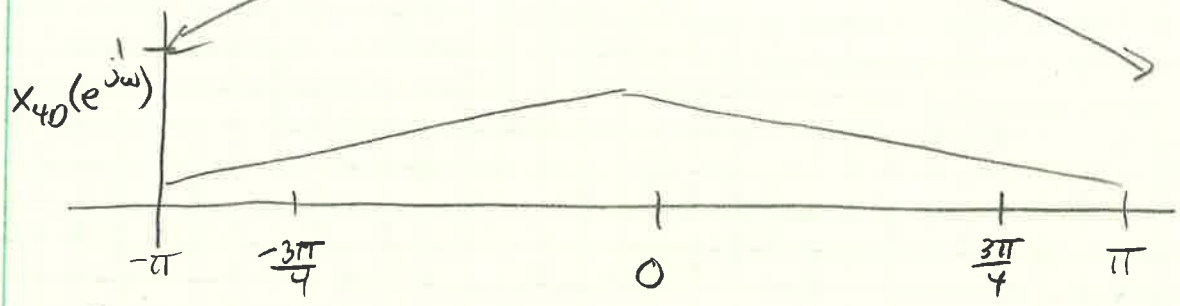


- Downsampling by 2 with antialiasing filter

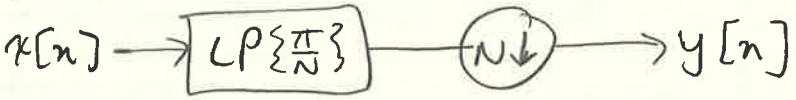




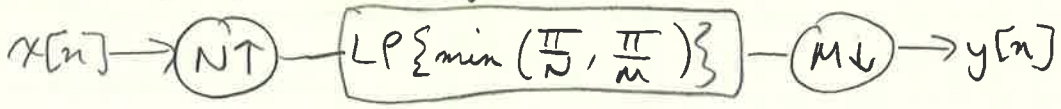
2 copies



- Downsampling



- Rational Sampling Rate Change

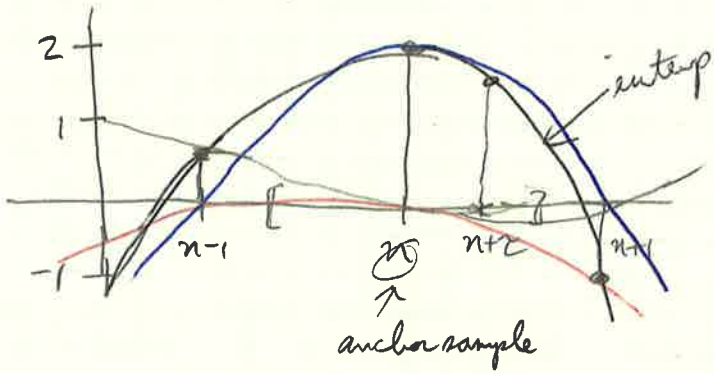


Example CD to DVD:

- CD: $F_s = 44100$ Hz
- DVD: $F_s = 48000$ Hz
- $\frac{N}{M} = \frac{160}{147}$
- in practice, we use time-varying local interpolation

5.9* FIR-based sampling rate conversion

- Subsample Interpolation



$|\tau| < \frac{1}{2}$

- we want to compute $x(n+\tau)$, with $|\tau| < \frac{1}{2}$
- local Lagrange approximation around n

$$x_L(n;t) = \sum_{k=-N}^{N} x[n-k]\, L_k^{(N)}(t)$$

$$L_k^{(N)}(t) = \prod_{\substack{i=-N \\ i\neq N}}^{N} \frac{t-i}{k-i}, \quad k = -N, \dots, N$$

- $x(n+\tau) \approx x_L(n;\tau)$

- Lagrange interpolation as an FIR

- $x(n+\tau) \approx x_L(n;\tau)$
- $x_L(n;\tau) = \sum_{k=-N}^{N} x[n-k]\, L_k^{(N)}(\tau)$
- define $d_\tau[k] = L_k^{(N)}(\tau)$, $k = -N, \dots, N$
- $x_L(n;\tau) = \sum_{k=-N}^{N} x[n-k]\, d_\tau[k]$
- $x_L(n;\tau) = (x * d_\tau)[n]$
- $d_\tau[k]$ is a $(2N+1)$-tap FIR (dependent on τ)

- Example ($N=1$, 2nd-order approximation)

$$L_{-1}^{(1)}(t) = t\,\frac{t-1}{2}$$

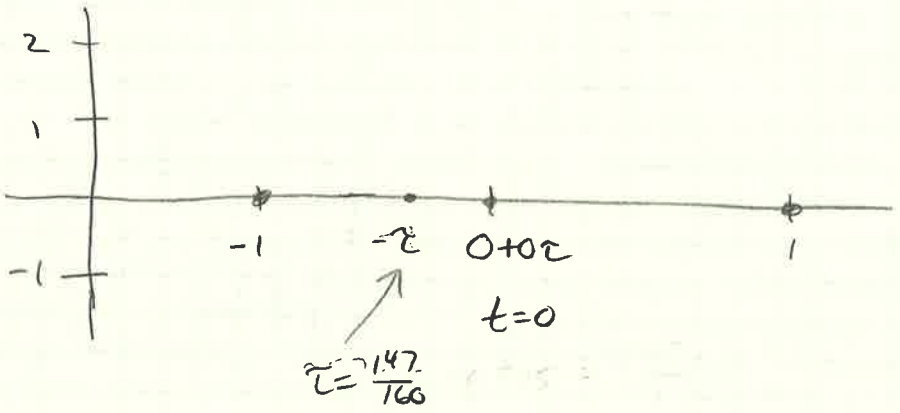
$$L_0^{(1)}(t) = (1-t)(1+t)$$

$$L_1^{(1)}(t) = t\,\frac{t+1}{2}$$

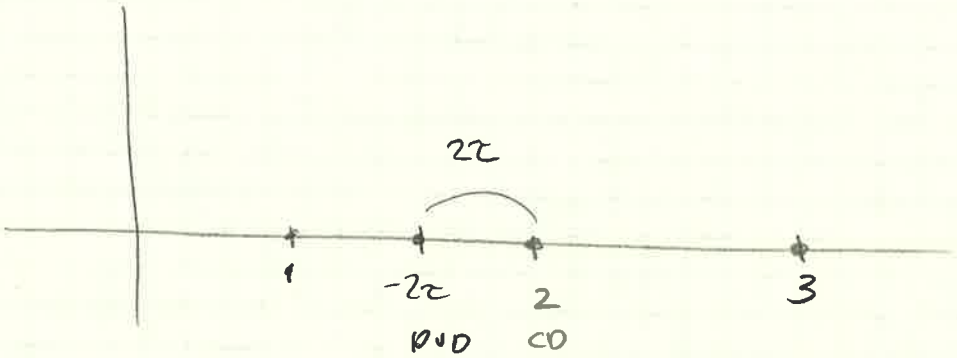
$$d_{0.2}[n] = \begin{cases} -0.08, & n=-1 \\ 0.96, & n=0 \\ 0.12, & n=1 \\ 0, & \text{otherwise} \end{cases} \qquad (\tau = 0.2)$$

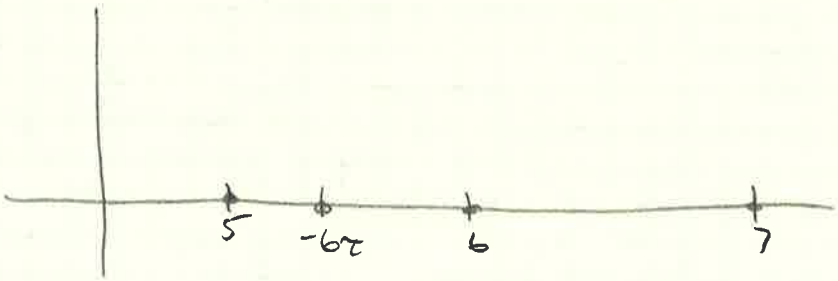
- CD to DVD, revisited

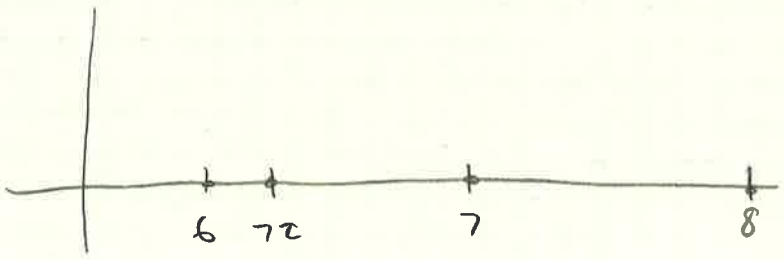
for every 147 CD samples, generate 160 DVD samples



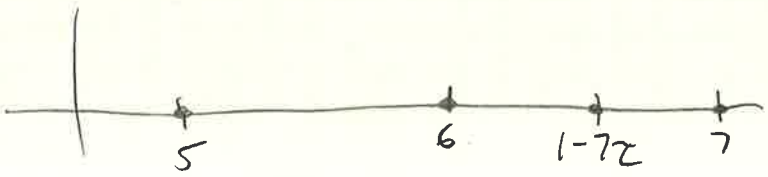
$\tau[1] = 0.06875$



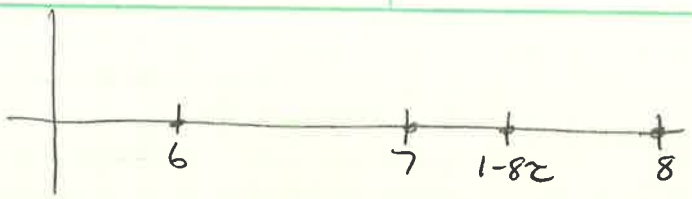




but $-7\tau < -0.5$
so reuse fractional point



repeat a sample



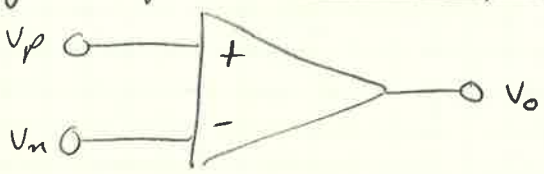
efficient local interpolation with 160 3-tap filters, used in sequence

5.10* Analog-to-digital and digital-to-analog converters

- From analog to digital

- sampling discretizes time
- quantization discretizes amplitude
- how is it done in practice?

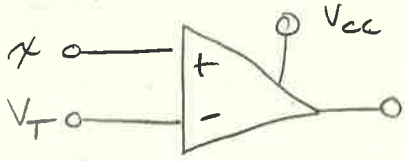
- A tiny bit of electronics: the op-amp



$$v_o = G(v_p - v_n)$$

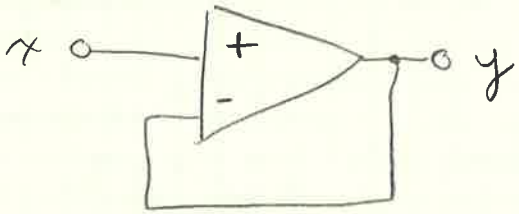
- the two key properties
  - infinite input gain ($G \approx \infty$)
  - zero input current

- The op-amp in open loop: comparator



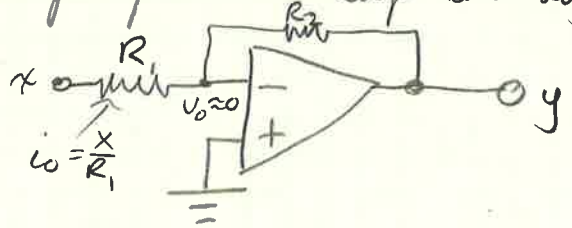
$$y = \begin{cases} +V_{cc}, & x > V_T \\ -V_{cc}, & x < V_T \end{cases}$$

- The op-amp in closed loop: buffer



$$y = x$$

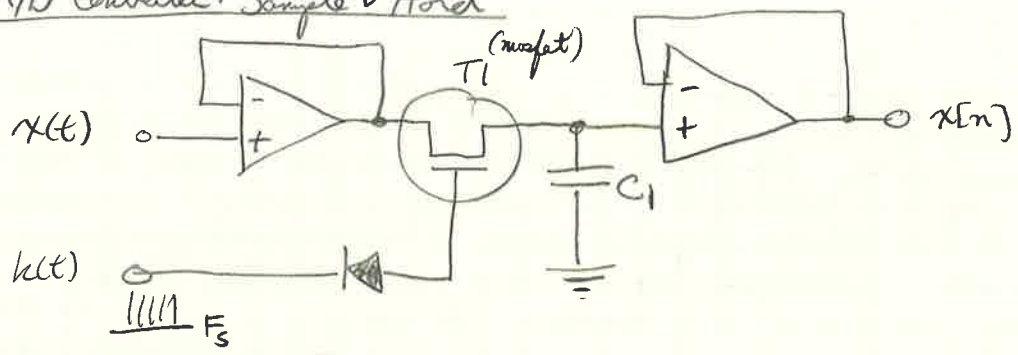
- The op-amp in closed loop: inverting amplifier



$$y = -R_2 i_o$$

$$y = -\left(\frac{R_2}{R_1}\right) x$$

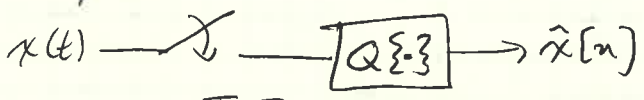
- A/D Converter: Sample & Hold



5.11* Oversampling

- oversampled A/D
  - → reduced quantization error
- oversampled D/A
  - use cheaper hardware for interpolation

- Oversampled A/D



$\hat{x}[n] = x[n] + e[n]$ ← noise samples created by quantizer

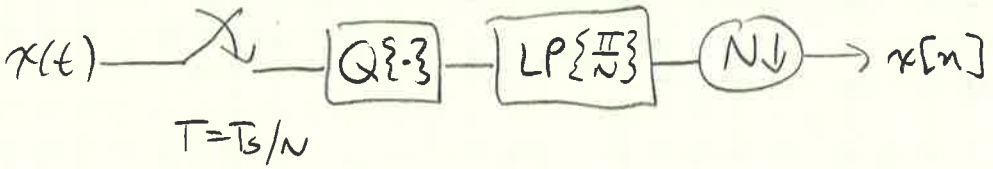
- Key assumptions:

  $e[n]$ i.i.d. process, independent of $x[n]$

  $P_e(e^{j\omega}) = \frac{\Delta^2}{12}$ over $[-\pi, \pi]$ (no aliasing)

- Key observation:

$$X(e^{j\omega}) = \frac{1}{T_s} X\left(j\frac{\omega}{T_s}\right)$$



- $SNR_0 \approx N \cdot SNR$
- 3dB per octave (doubling of F_s)
- but key assumption (independence) breaks down fast...

- Oversampled D/A

Sinc interpolation: $X_c(j\Omega) = \frac{\pi}{\Omega_N} X(e^{j\pi\Omega/\Omega_N}) \operatorname{rect}\left(\frac{\Omega}{2\Omega_N}\right)$

In general: $X_c(j\Omega) = \frac{\pi}{\Omega_N} X(e^{j\pi\Omega/\Omega_N})\, I\left(j\pi\frac{\Omega}{\Omega_N}\right)$

The cheapest (hence most common) interpolator:

$$i(t) = \text{rect}(t) \longleftrightarrow I(j\Omega) = \text{sinc}\left(\frac{\Omega}{2\pi}\right)$$

Consider a K-upsampled and interpolated version of $x[n]$:

$$X_K(e^{j\omega}) = X(e^{j\omega K})\,\text{rect}\left(\frac{\omega K}{2\pi}\right), \quad 2\pi\text{-periodic}$$

sinc-interpolate $x_K[n]$ with $T_s' = T_s/K$:

$$\Omega_N' = K\Omega_N$$

$$X_{c,K}(j\Omega) = \frac{\pi}{\Omega_N'} X_K(e^{j\omega})\Big|_{\omega=\frac{\pi\Omega}{\Omega_N'}} \text{rect}\left(\frac{\Omega}{2\Omega_N'}\right)$$

$$= \frac{\pi}{K\Omega_N} X(e^{j\pi\Omega/\Omega_N})\,\text{rect}\left(\frac{\Omega}{2\Omega_N}\right)\text{rect}\left(\frac{\Omega}{2K\Omega_N}\right)$$

$$= \frac{1}{K} X_c(j\Omega), \quad |\Omega| < \Omega_N$$

6.1 Introduction to digital communication systems

6.1.a The success factors for digital communications

-Digital data throughputs

· Transatlantic cable:

- 1866: 8 word per minute (≈5 bps)
- 1956: AT&T, coax, 48 voice channels (≈3Mbps)
- 2005: Alcatel TeraLO, fiber, 8.4 Tb/s (8.4×10^{12} bps)
- 2012: fiber, 60 Tbps

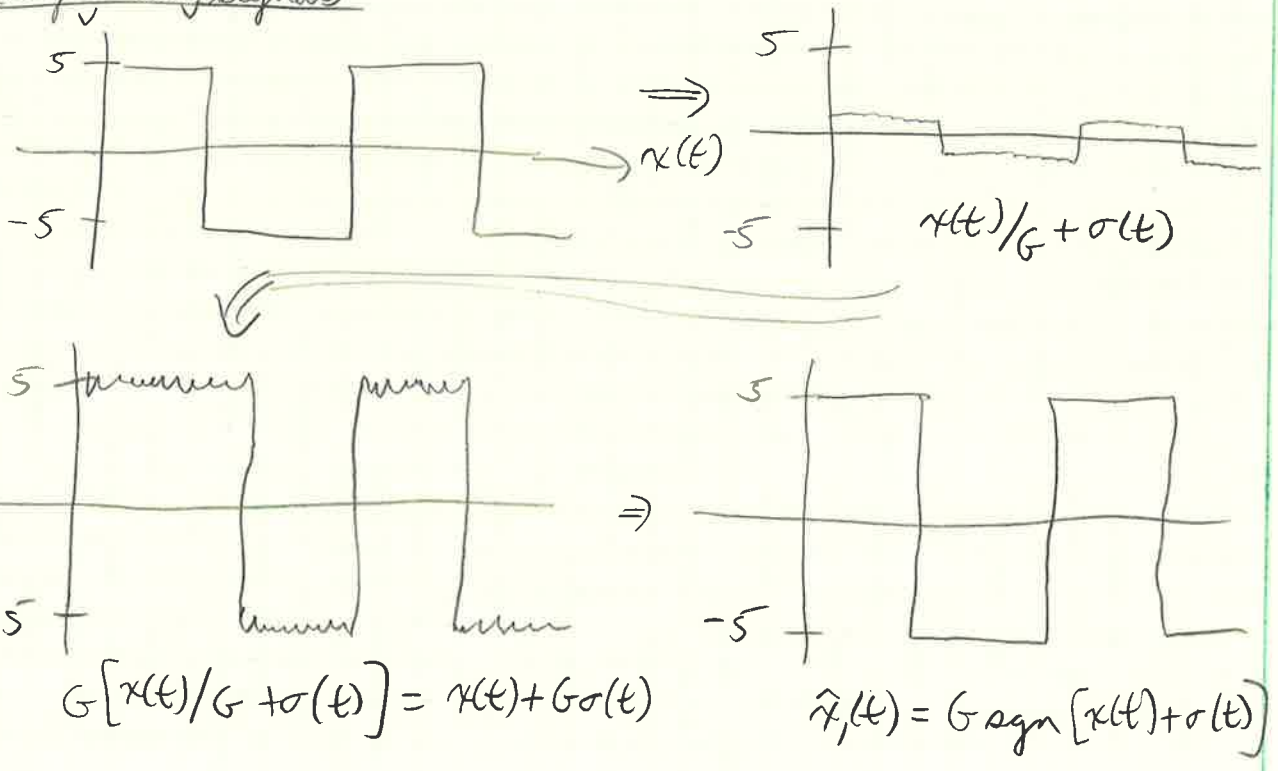
· Voiceband modems

- 1950s: Bell 202, 1200 bps
- 1990s: V90, 56 Kbps
- 2008: ADSL+, 24Mbps

-Success factors for digital communications

1) power of the DSP paradigm:
   - integers are "easy" to regenerate
   - good phase control
   - adaptive algorithms

-Regenerating signals



$$G\left[x(t)/G + \sigma(t)\right] = x(t) + G\sigma(t)$$

$$\hat{x}_1(t) = G\,\mathrm{sgn}\left[x(t) + \sigma(t)\right]$$

2) algorithmic nature of DSP is a perfect match with information theory:
   - JPEG's entropy coding
   - CD's and DVD's error correction
   - trellis-coded modulation and Viterbi decoding

3) hardware advancement
   - miniaturization
   - general-purpose platforms
   - power efficiency

6.1.6 The analog channel constraints

- The analog channel

unescapable "limits" of physical channels:
- bandwidth constraint
- power constraint

both constraints will affect the final capacity of the channel

- The analog channel's capacity

maximum amount of information that can be reliably delivered over a channel (bits per second)

- Bandwidth vs. capacity

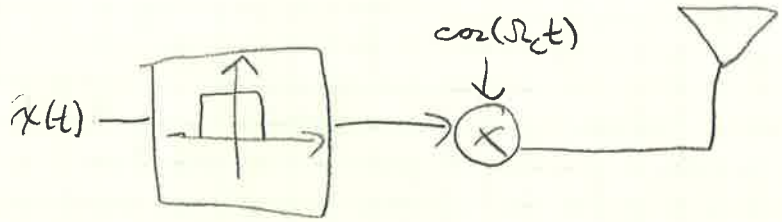
Simple thought experiment
- we want to transmit information encoded as a sequence of digital samples over a continuous-time channel
- we interpolate the sequence of samples with a period T_s
- if we make T_s small we can send more info per unit of time, but the bandwidth of the signal will grow as $1/T_s$

$$X(j\Omega) = 0 \quad , \quad |\Omega| > \Omega_N = \frac{\pi}{T_s}$$

- Power and capacity

Another thought experiment:
- all channels introduce noise; at the receiver we have to "guess" what was transmitted.
- suppose noise variance is 1
- suppose we are transmitting integers between 1 and 10: lots of guessing errors
- transmit only odd numbers: fewer errors but less information

- Example: the AM radio channel



- from 530 kHz to 1.7 MHz
- each channel is 8 kHz
- power limited by law:
  - daytime / nighttime
  - interference
  - health hazards

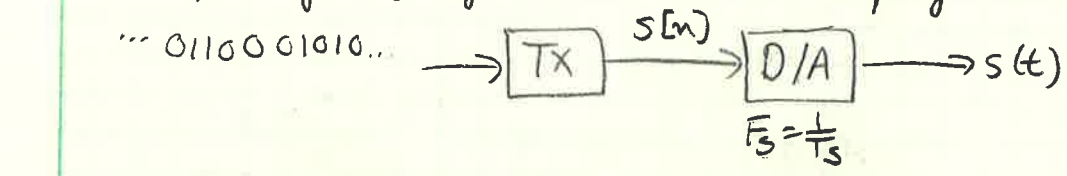
- Example: the telephone channel

- one channel from around 300 Hz to around 3000 Hz
- power limited by law to 0.2 - 0.7 V rms
- noise is rather low: SNR usually 30 dB or more

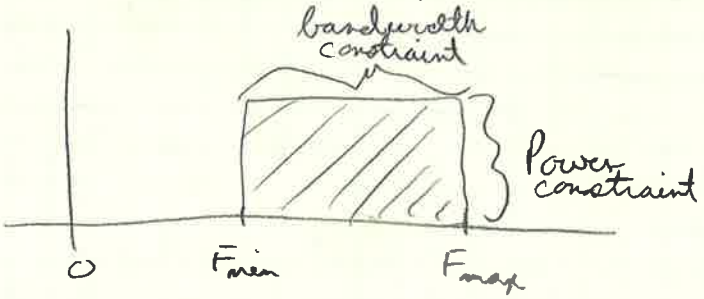
6.1.c the design problem

- The all-digital paradigm

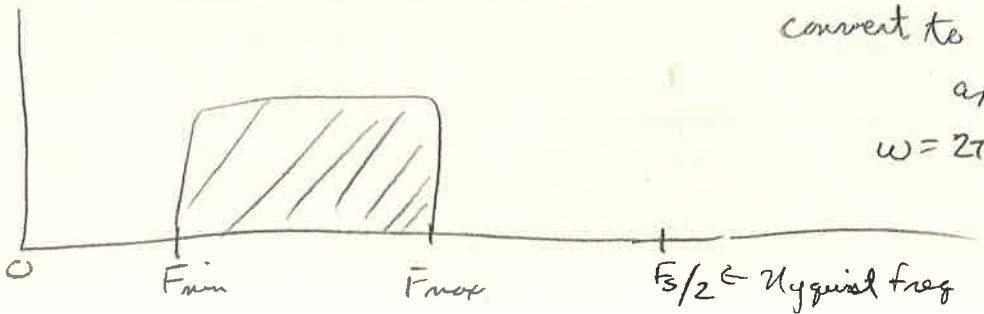
keep everything digital until we hit the physical channel



Let's look at the channel constraints



Converting the specs to a digital design



convert to ω_{min}, ω_{max} and π

$$\omega = 2\pi \frac{F}{F_S}$$

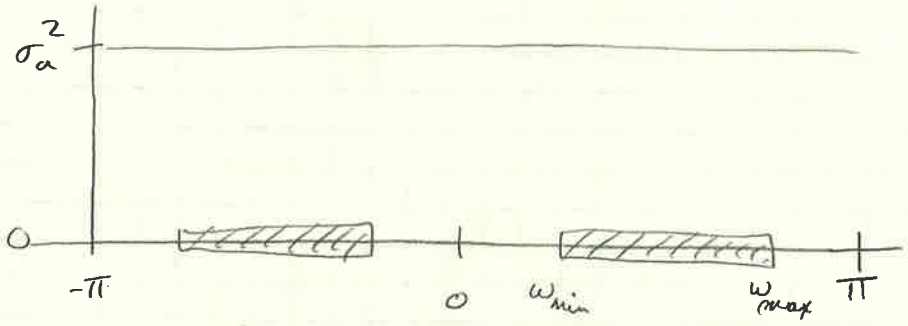
- Transmitter design

Some working hypotheses:
- convert the bitstream into a sequence of symbols $a[n]$ via a mapper
- model $a[n]$ as a white random sequence (add a scrambler on the bitstream to make sure)

- now we need to convert $a[n]$ into a continuous time signal within the constraints

- First problem: the bandwidth constraint

$$P_a(e^{j\omega}) = \sigma_a^2$$



- Signal of the Day: Moiré Patterns

- Moiré Patterns in 1D: Beatings

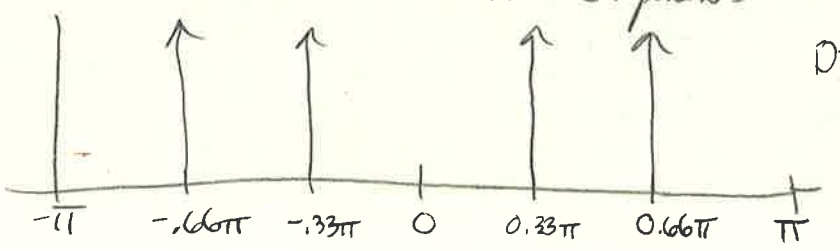
$$x[n] = \cos(\omega_1 n) + \cos(\omega_2 n)$$
$$= 2\cos\left(\frac{\omega_1 - \omega_2}{2} n\right) \cos\left(\frac{\omega_1 + \omega_2}{2} n\right)$$
$$\approx 2\cos(\Delta_\omega n)\cos(\omega_1 n) \qquad (\omega_1 \approx \omega_2)$$

- ω_1 is the nominal "high" frequency
- Δ_ω is a low frequency beating

- Moiré and Aliasing; 1D example

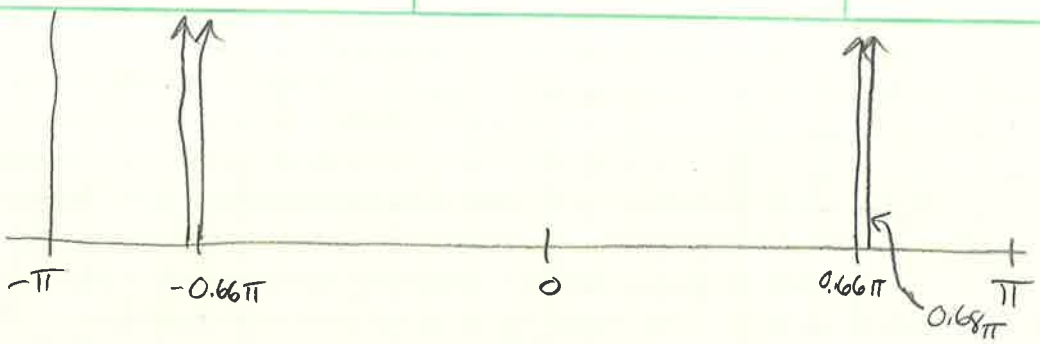
$$x(t) = \cos(2\pi f_0 t) + \cos(4\pi f_0 t), \quad f_0 = 165\,Hz$$

tone with 2 harmonic components



DTFT

Sampling rate $F_s = 1000\,Hz$



Use a lowpass filter to avoid Moiré patterns!

6.2 Controlling the bandwidth

6.2.a Upsampling

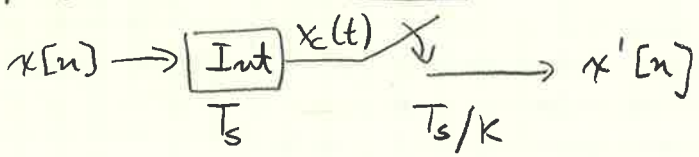
- Shaping the bandwidth

Our problem:
- bandwidth constraint requires us to control the spectral support of a signal
- we need to be able to "shrink" the support of a full-band signal
- the answer is multirate techniques

- Multirate signal processing

In a nutshell:
- increase or decrease the number of samples in a discrete-time signal
- equivalent to going to continuous time and resampling
- staying in the digital world is "cleaner"

- Upsampling via continuous time

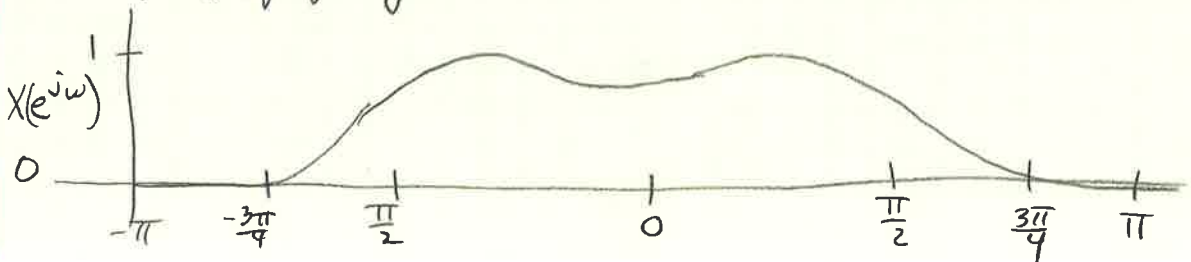


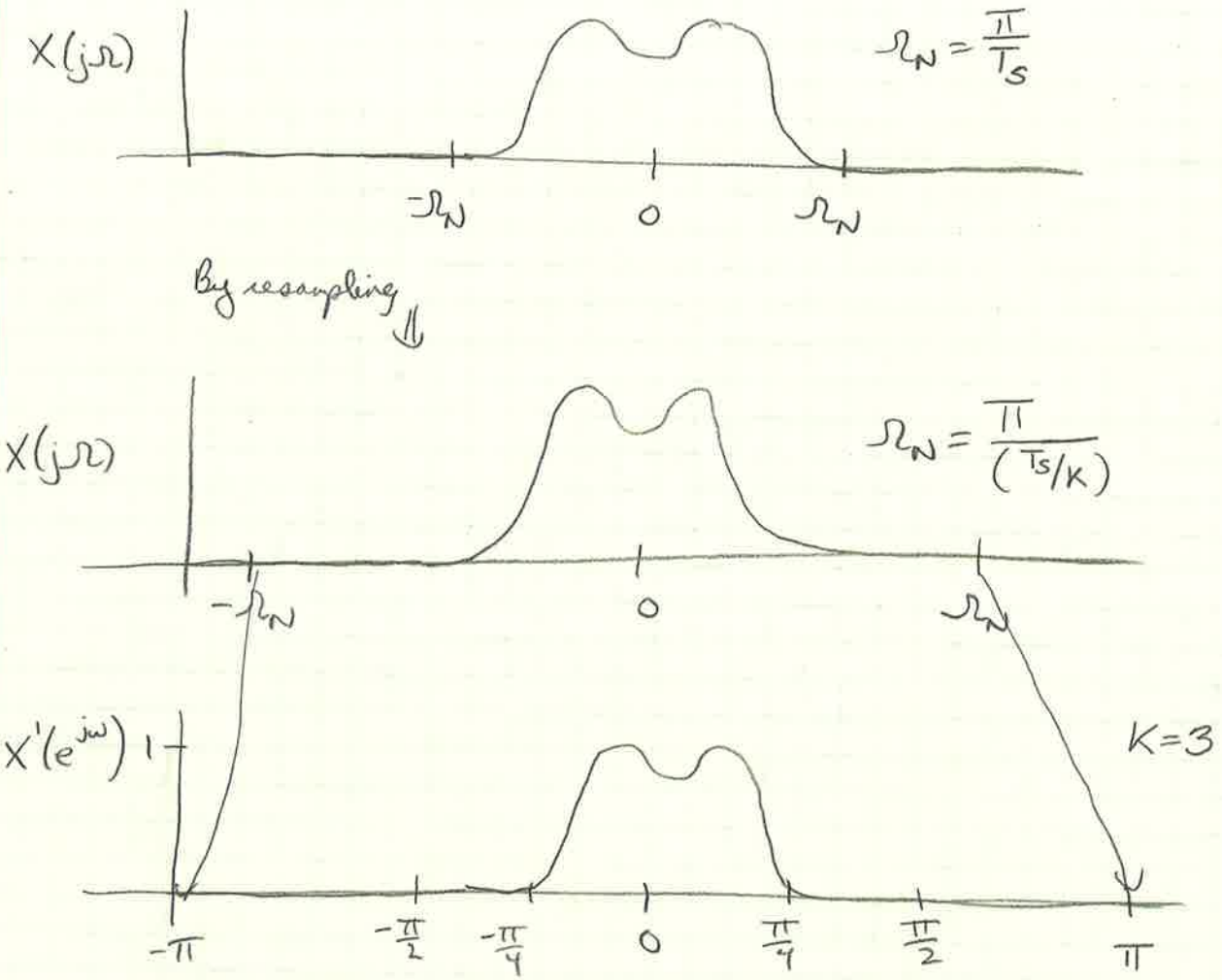
As per usual, we can choose $T_s = 1$...

$$x_c(t) = \sum_{m=-\infty}^{\infty} x[m] \operatorname{sinc}(t-m)$$

$$x'[n] = x_c(n/K) = \sum_{m=-\infty}^{\infty} x[m] \operatorname{sinc}\left(\frac{n}{K} - m\right)$$

- Upsampling (frequency domain)





- Upsampling in the digital domain

What can we do purely digitally?

- we need to "increase" the number of samples by K
- obviously $x_u[m] = x[n]$ when m is a multiple of K
- for lack of a better strategy, put zeros elsewhere
- example for $K = 3$:

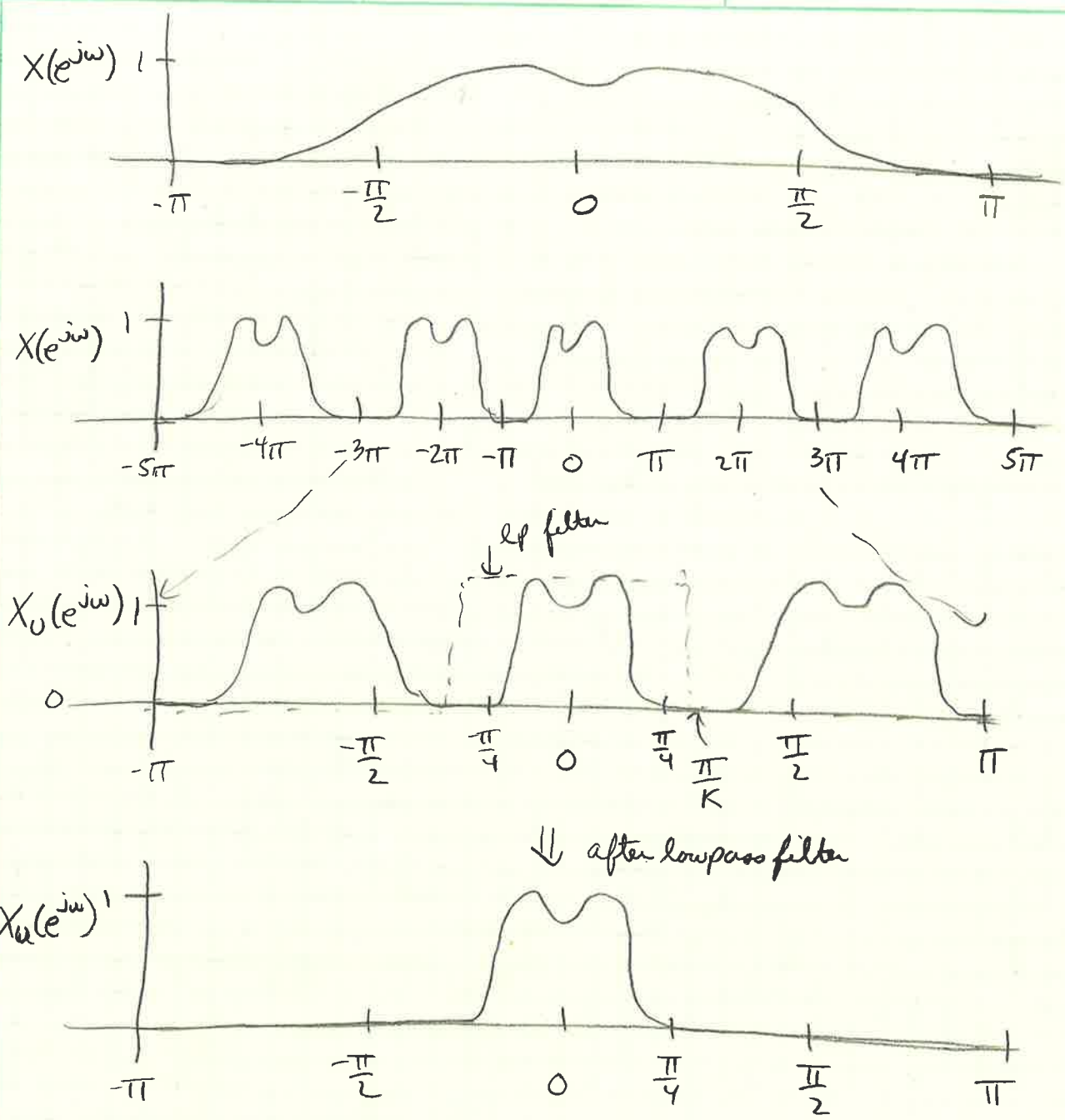
$$x_u[m] = \ldots \underset{\uparrow\, m=0}{x[0]}, 0, 0, \underset{\uparrow\, m=1}{x[1]}, 0, 0, x[2], 0, 0, \ldots$$

in the frequency domain

$$X_u(e^{j\omega}) = \sum_{m=-\infty}^{\infty} x_u[m]\, e^{-j\omega m}$$

$$= \sum_{n=-\infty}^{\infty} x[n]\, e^{-j\omega nK} \qquad x_u[m] = 0,\ m \neq nK$$

$$= X(e^{j\omega K})$$



back in time domain ...

- insert $k-1$ zeros after every sample
- ideal lowpass filtering with $\omega_c = \pi/K$

$$x'[n] = x_u[n] * \text{sinc}(n/K)$$

$$= \sum_{i=-\infty}^{\infty} x_u[i] \, \text{sinc}\left(\frac{n-i}{K}\right)$$

$$= \sum_{m=-\infty}^{\infty} x[m] \, \text{sinc}\left(\frac{n}{K} - m\right) \qquad i = mk$$

- Downsampling

- given an upsampled signal we can always recover the original:

$$x[n] = x_u[nK]$$

- downsampling of generic signals is more complicated (aliasing)

6.2.5 Fitting the transmitter spectrum

- Remember the bandwidth constraint? Here's a neat trick:

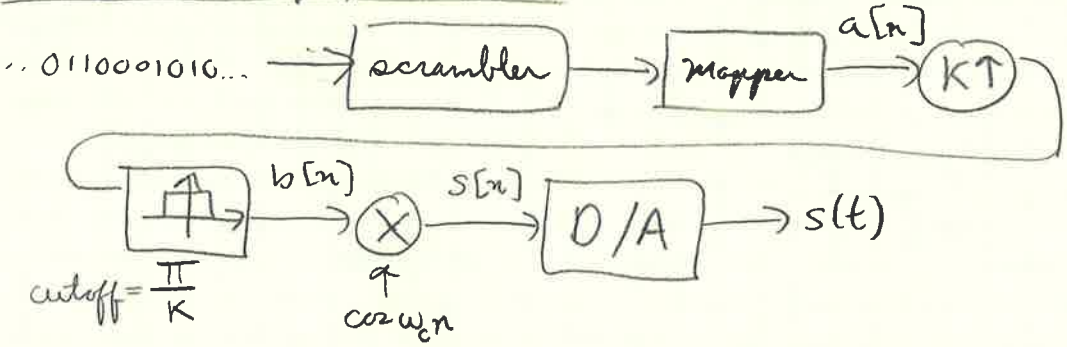
let $W = F_{max} - F_{min}$; pick F_s so that:

- $F_s > 2F_{max}$ (obviously)
- $F_s = KW$, $K \in \mathbb{N}$
- $\omega_{max} - \omega_{min} = 2\pi \frac{W}{F_s} = \frac{2\pi}{K}$ $\left(\omega = 2\pi \frac{F}{F_s}\right)$
- we can simply upsample by K

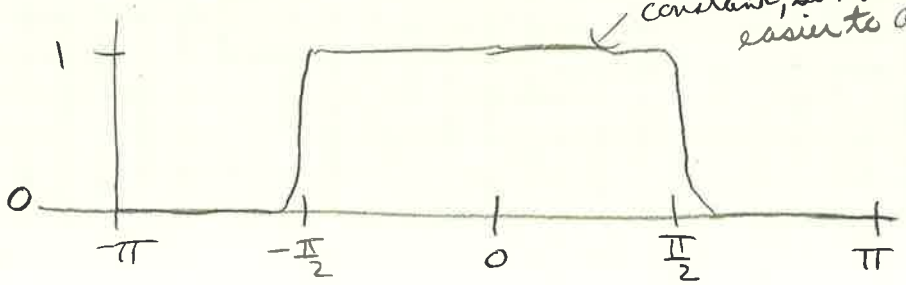
- Data rates

- upsampling does not change the data rate
- we produce (and transmit) W symbols per second
- W is sometimes called the Baud rate of the system and is equal to the available bandwidth

- Transmitter design, continued



- Raised Cosine



constant, so still ideal but easier to approximate than sinc

Impulse response decays as $\frac{1}{n^3}$

6.3 Controlling the power

6.3.a Noise and probability of error

- Transmission reliability

- transmitter sends a sequence of symbols $a[n]$
- receiver obtains a sequence $\hat{a}[n]$

- even if no distortion, we can't avoid noise: $\hat{a}[n] = a[n] + \eta[n]$
- when noise is large, we make an error

- Probability of error

depends on:
- power of the noise w.r.t. power of the signal
- decoding strategy
- alphabet of transmission symbols

- Signaling alphabets

- we have a (randomized) bitstream coming in
- we want to send some unsampled and interpolated samples over the channel
- how do we go from bitstream to samples?

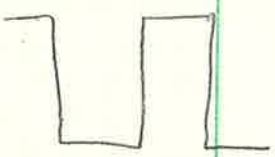
- Mappers and slicers

mapper:
- split incoming bitstream into chunks
- assign a symbol $a[n]$ from a finite alphabet $\mathcal{A}$ to each chunk

slicer:
- receive a value $\hat{a}[n]$
- decide which symbol from $\mathcal{A}$ is "closest" to $\hat{a}[n]$
- piece back together the corresponding bitstream

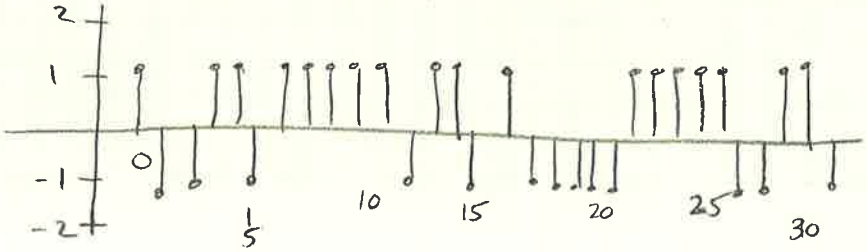
- Example: two-level signaling

mapper:
- split incoming bitstream into single bits
- $a[n] = G$ if the bit is 1, $a[n] = -G$ if the bit is 0

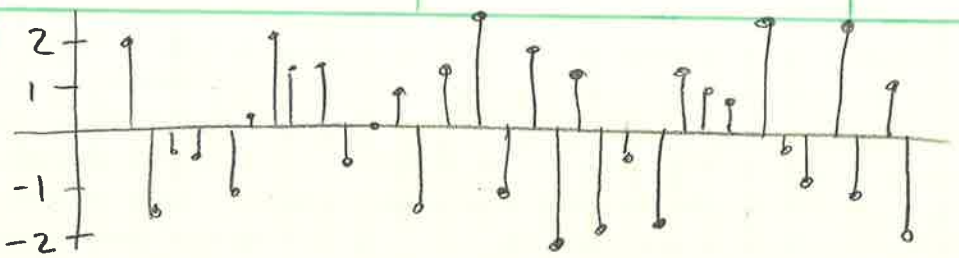


slicer:

$$n^{th}\ \text{bit} = \begin{cases} 1, & \hat{a}[n] > 0 \\ 0, & \text{otherwise} \end{cases}$$



$G = 1$



after apply a random noise

Let's look at the probability of error after making some hypotheses:

- $\hat{a}[n] = a[n] + \eta[n]$
- bits in bitstream are equiprobable
- noise and signal are independent
- noise is additive Gaussian white noise with zero mean and variance σ_0^2

$$P_{err} = P[\eta[n] < -G \mid n^{th} \text{ bit is } 1] + P[\eta[n] > G \mid n^{th} \text{ bit is } 0]$$

$$= \left(P[\eta[n] < -G] + P[\eta[n] > G]\right)/2$$

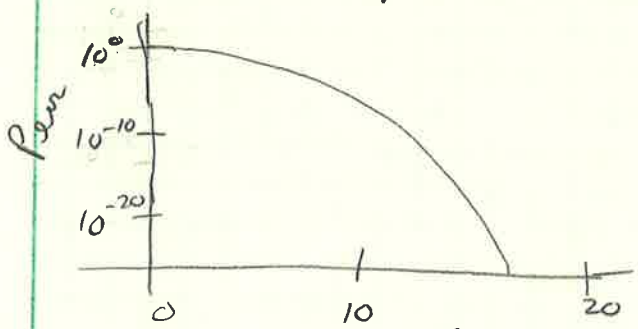
$$= P[\eta[n] > G]$$

$$= \int_G^\infty \frac{1}{\sqrt{2\pi\sigma_0^2}} e^{-\frac{\tau^2}{2\sigma_0^2}} d\tau = \operatorname{erfc}\left(\frac{G}{\sigma_0}\right)$$

transmitted power

$$\sigma_s^2 = G^2 P[n^{th} \text{ bit is } 1] + G^2 P[n^{th} \text{ bit is } 0] = G^2$$

$$P_{err} = \operatorname{erfc}(\sigma_s/\sigma_0) = \operatorname{erfc}\left(\sqrt{SNR}\right)$$



$$dB = 10 \log_{10} \frac{\sigma_s^2}{\sigma_0^2}$$

Lesson learned:

- to reduce the probability of error, increase G
- increasing G increases the power
- we can't go above the channel's power constraint!

- Multilevel signaling

- binary signaling is not very efficient (one bit at a time)
- to increase the throughput we can use multilevel signaling
- many ways to do so; we will just scratch the surface

6.3.6 PAM and QAM

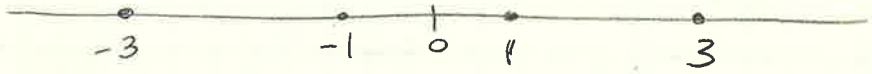
- PAM (Pulse amplitude modulation)

mapper:
- split incoming bitstream into chunks of M bits
- chunks define a sequence of integers $k[n] \in \{0, 1, \dots, 2^M - 1\}$
- $a[n] = G\left((-2^M + 1) + 2k[n]\right)$ (odd integers around zero)

If $M = 2$
$k[n] \in \{0, 1, 2, 3\}$
$a[n] \in \{-3, -1, 1, 3\}$
$G = 1$

slicer:
- $a'[n] = \underset{a \in \mathcal{A}}{\operatorname{argmin}} \left[|\hat{a}[n] - a| \right]$

- PAM, $M = 2$, $G = 1$



- distance between points is $2G$
- using odd integers creates a zero-mean sequence

- From PAM to QAM (Quadrature amplitude modulation)

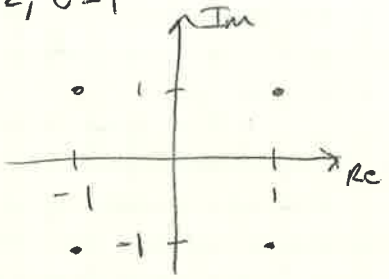
- error analysis for PAM along the lines of binary signaling
- can we increase the throughput even further?
- here's a wild idea, let's use complex numbers

- QAM

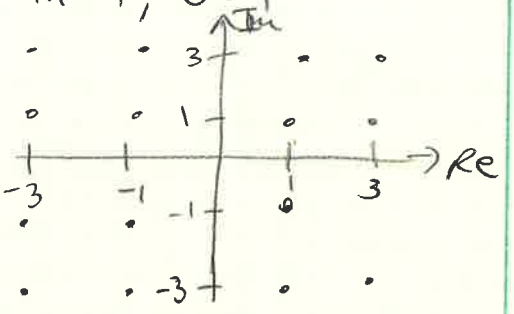
mapper:
- split incoming bitstream into chunks of M bits, M even
- use $M/2$ bits to define a PAM sequence $a_r[n]$
- use the remaining $\frac{M}{2}$ bits to define an independent PAM sequence $a_i[n]$
- $a[n] = G\left(a_r[n] + j\, a_i[n]\right)$

slicer:
- $a'[n] = \underset{a \in \mathcal{A}}{\operatorname{argmin}} \left[|\hat{a}[n] - a| \right]$

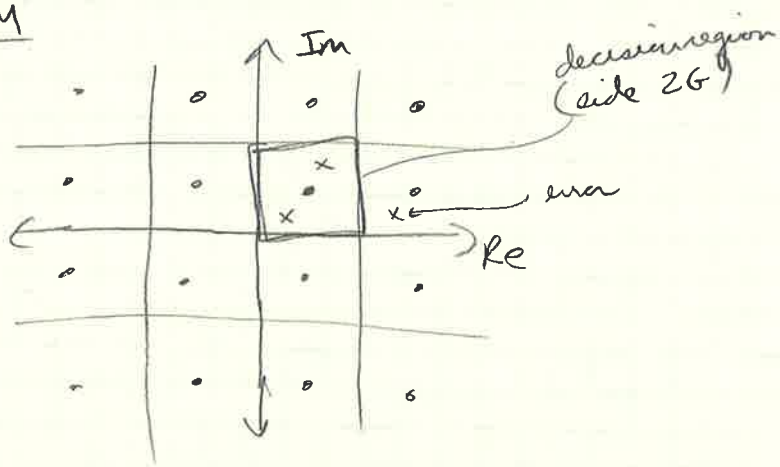
- QAM, $M=2$, $G=1$



$M=4$, $G=1$



- QAM



- QAM, probability of error

$$P_{err} = P\left[\left|Re(\eta[n])\right| > G\right] + P\left[\left|Im(\eta[n])\right| > G\right]$$

$$= 1 - P\left[\left|Re(\eta[n])\right| < G \underset{\text{and}}{\wedge} \left|Im(\eta[n])\right| < G\right]$$

$$= 1 - \int_{D_R \text{ decision region}} f_\eta(z)\,dz$$

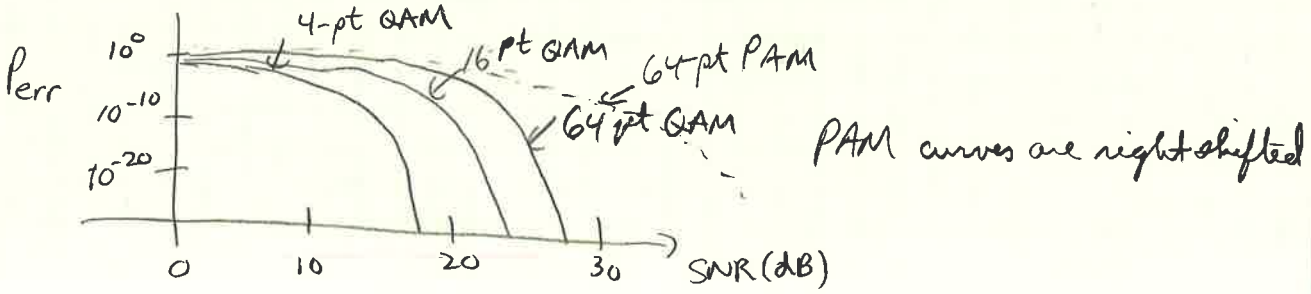
- approximate D by inscribed circle (of radius G)

$$\Rightarrow P_{err} \approx e^{-G^2/\sigma_0^2}$$

transmitted power (all symbols are equiprobable and independent)

$$\sigma_s^2 = G^2 \frac{1}{2^M} \sum_{a \in A} |a|^2 = G^2 \frac{2}{3}(2^M - 1)$$

$$\Rightarrow P_{err} \approx e^{-G^2/\sigma_0^2} \approx e^{-3 \cdot 2^{-(M+1)} \cdot SNR}$$



PAM curves are right-shifted

- QAM, the recipe

  - pick a probability of error you can live with (e.g. 10^{-6})
  - find out the SNR imposed by the channel's power constraint
  - $M = \log_2\left(1 - \frac{3}{2}\frac{SNR}{\ln(p_e)}\right)$ (p_e = P_{err})
  - final throughput will be MW

- QAM, where we stand

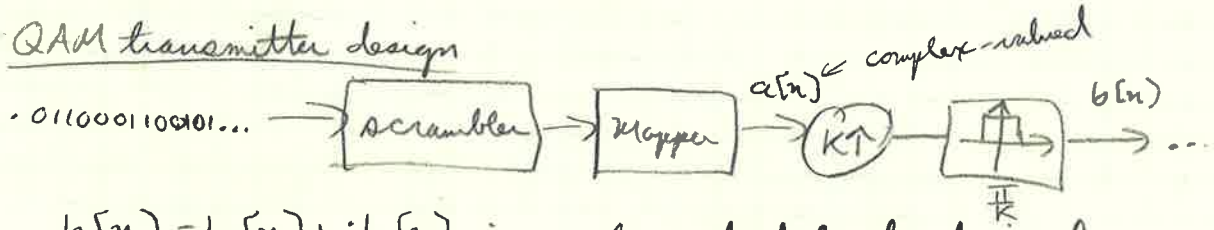
  - we know how to fit the bandwidth constraint
  - with QAM, we know how many bits per symbol we can use given the power constraint
  - we know the theoretical throughput of the transmitter

  but how do we transmit complex symbols over a real channel?

6.4 Modulation and demodulation

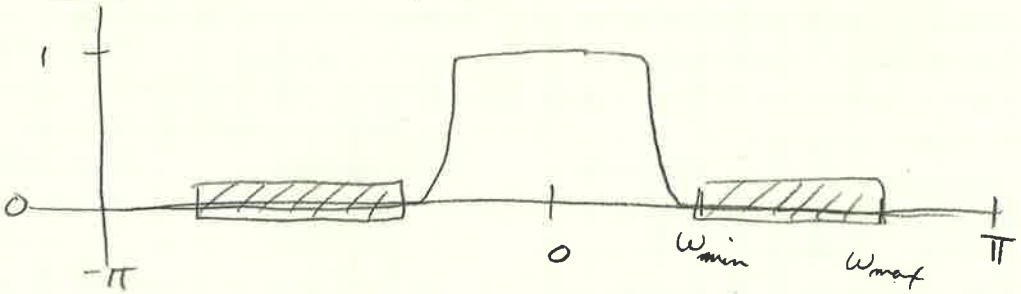
6.4.a Modulation and demodulation

- QAM transmitter design



$b[n] = b_r[n] + j\,b_i[n]$ is a complex-valued baseband signal

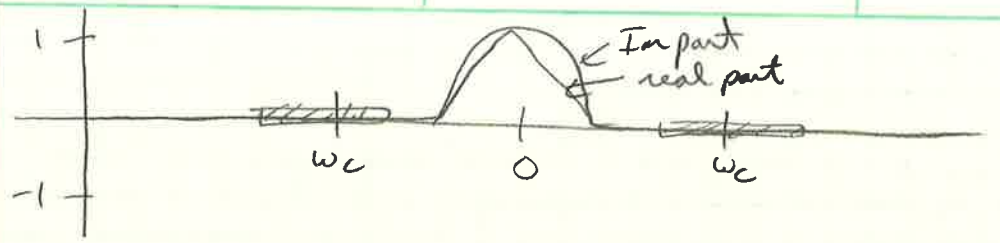
- Complex baseband signal



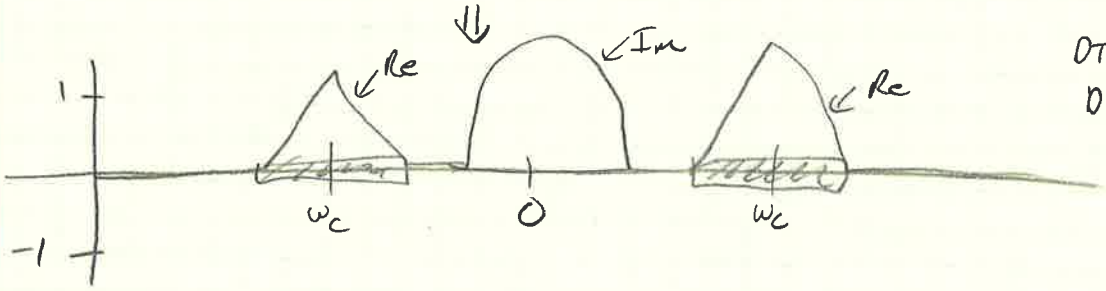
- The passband signal

$$s[n] = Re\left\{ b[n]\, e^{j\omega_c n} \right\}$$
$$= Re\left\{ (b_r[n] + j\,b_i[n])(\cos\omega_c n + j\sin\omega_c n) \right\}$$
$$= b_r[n]\cos\omega_c n - b_i[n]\sin\omega_c n \in \mathbb{R}$$

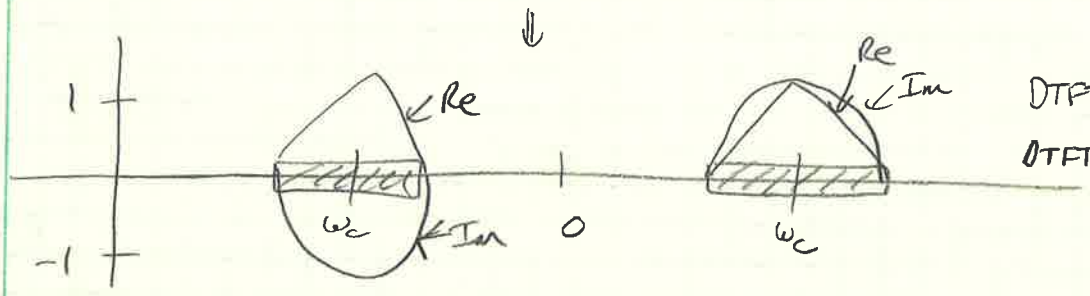
($b_r[n]\cos\omega_c n$: in phase part; $b_i[n]\sin\omega_c n$: quadrature part)



DTFT $\{b[n]\}$



DTFT $\{b_r[n] \cos \omega_c n\}$
DTFT $\{b_i[n]\}$



DTFT $\{b_r[n] \cos \omega_c n\}$
DTFT $\{-b_i[n] \sin \omega_c n\}$

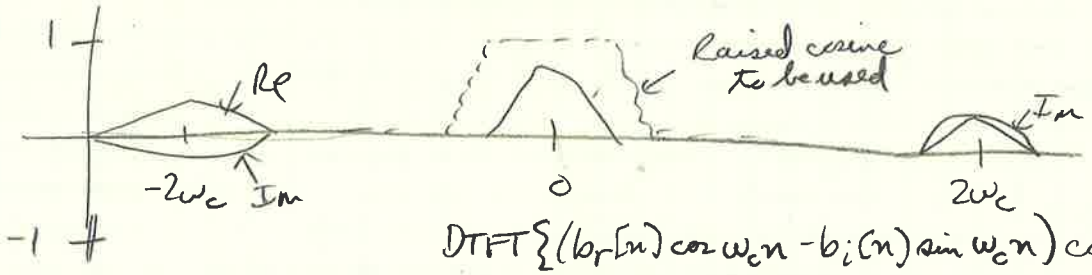
- Recovering the baseband signal

Let's try the usual method (multiplying by the carrier, see ncd 5.5):

$$s[n] \cos \omega_c n = b_r[n] \cos^2 \omega_c n - b_i[n] \sin \omega_c n \cos \omega_c n$$

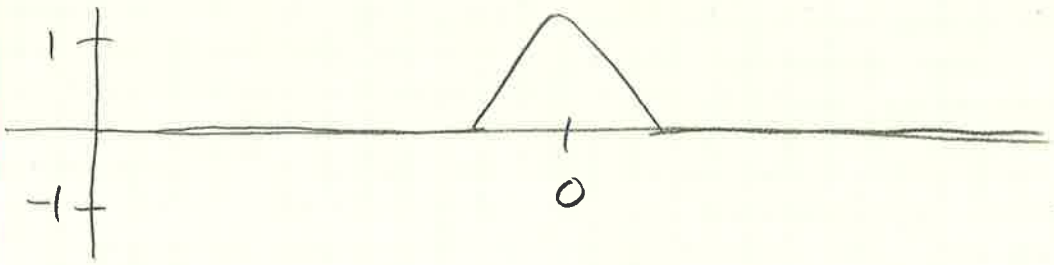
$$= b_r[n] \frac{1 + \cos 2\omega_c n}{2} - b_i[n] \frac{\sin 2\omega_c n}{2}$$

$$= \frac{1}{2} b_r[n] + \frac{1}{2} (b_r[n] \cos 2\omega_c n - b_i[n] \sin 2\omega_c n)$$



DTFT $\{(b_r[n] \cos \omega_c n - b_i[n] \sin \omega_c n) \cos \omega_c n\}$

⇓ after filtering

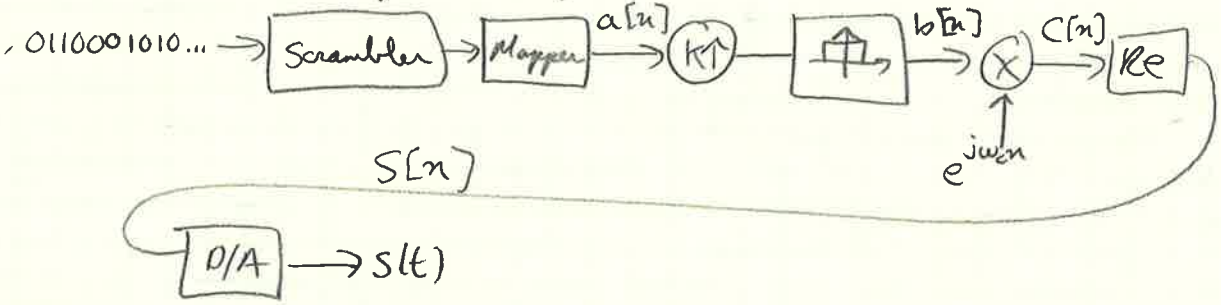


Similarly:

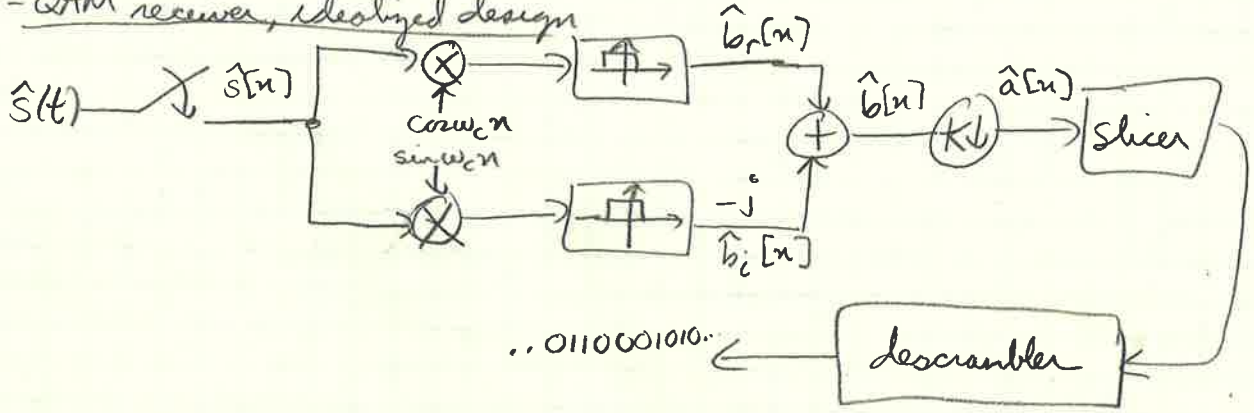
$$s[n]\sin\omega_c n = b_r[n]\cos\omega_c n \sin\omega_c n - b_i[n]\sin^2\omega_c n$$
$$= -\frac{1}{2}b_i[n] + \frac{1}{2}\left(b_r[n]\sin 2\omega_c n - b_i[n]\cos 2\omega_c n\right)$$

6.4.6 Design example

- QAM transmitter, final design



- QAM receiver, idealized design



- Example: the V.32 voiceband modem

- analog telephone channel: $F_{min} = 450$ Hz, $F_{max} = 2850$ Hz
- useable bandwidth: $W = 2400$ Hz, center frequency $F_c = 1650$ Hz
- pick $F_S = 3 \cdot 2400 = 7200$ Hz, so that $K = 3$
- $\omega_c = 0.458\pi$
- maximum SNR: 22dB
- pick $P_{err} = 10^{-6}$
- using QAM, we find $M = \log_2\left(1 - \frac{3}{2}\frac{10^{22/10}}{\ln(10^{-6})}\right) \approx 4.1865$
  so we pick $M = 4$ and use a 16-pt constellation
- final data rate is $WM = 9600$ bits per second

- Theoretical channel capacity

· we used very specific design choices to derive the throughput
· what is the best one can do?
· Shannon's capacity formula is the upper bound: $C = W \log_2(1+SNR)$
· for instance, for the previous example $C \approx 17500$ bps
· the gap can be narrowed by more advanced coding techniques

6.5 Receiver design

6.5.a Receiver design

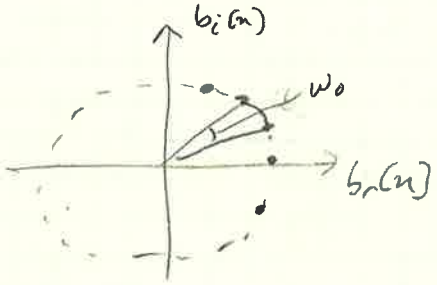
- Pilot tones

if $\hat{s}[n] = \cos((\omega_c+\omega_0)n)$:

$$\hat{b}[n] = \mathcal{H}\{\cos((\omega_c+\omega_0)n)\cos\omega_c n - j\cos((\omega_c+\omega_0)n)\sin\omega_c n\}$$

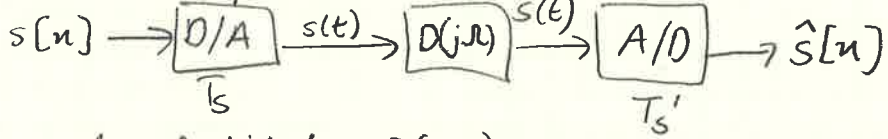
$$= \mathcal{H}\{\cos(\omega_0 n) + \cos((2\omega_c+\omega_0)n) - j\sin((2\omega_c+\omega_0)n) + j\sin\omega_0 n\}$$

$$= \cos\omega_0 n + j\sin\omega_0 n = e^{j\omega_0 n}$$



- It's a dirty job, but the receiver has to do it:
  · interference → handshake and line probing
  · propagation delay → delay estimation
  · linear distortion → adaptive equalization
  · clock drifts → timing recovery

- The two main problems



· channel distortion $D(j\Omega)$
· (time-varying) discrepancies in clocks $T_s' = T_s$

6.5.6 Delay compensation ($T'_S = T_S$)

Assume the channel is a simple delay: $\hat{s}(t) = s(t-d) \Rightarrow D(j\Omega) = e^{-j\Omega d}$

- channel introduces a delay of d seconds
- we can write $d = (b+\tau)T_S$ with $b \in \mathbb{N}$ and $|\tau| < 1/2$
- b is called the bulk delay
- τ is the fractional delay

- Estimating the fractional delay

- transmit $b[n] = e^{j\omega_0 n}$ (i.e. $s[n] = \cos((\omega_c+\omega_0)n)$)
- receive $\hat{s}[n] = \cos((\omega_c+\omega_0)(n-b-\tau))$
- after demodulation and bulk delay offset: $\hat{b}[n] = e^{j\omega_0(n-\tau)}$
- multiply by known frequency: $\hat{b}[n]e^{-j\omega_0 n} = e^{-j\omega_0\tau}$

- Compensating for the fractional delay

- $\hat{s}[n] = s(n-\tau)T_S$ (after offsetting bulk delay)
- we need to compute subsample values
- in theory, compensate with a sinc fractional delay $h[n] = \mathrm{sinc}(n+\tau)$
- in practice, use local Lagrange approximation

- Lagrange approximation (see Module 6.2)

as per usual, choose $T_S = 1$

- we want to compute $x(n+\tau)$, with $|\tau| < \frac{1}{2}$
- local Lagrange approximation around n

$$x_L(n;t) = \sum_{k=-N}^{N} x[n-k]\, L_k^{(N)}(t)$$

$$L_k^{(N)}(t) = \prod_{\substack{i=-N \\ i\neq n}}^{N} \frac{t-i}{k-i}, \qquad k = -N, \ldots, N$$

- $x(n+\tau) \approx x_L(n;\tau)$

- Lagrange interpolation as an FIR

  - $x(n+\tau) \approx x_L(n;\tau)$
  - define $d_\tau[k] = L_k^{(N)}(\tau)$, $k = -N, \dots, N$
  - $d_\tau[k]$ form a $(2N+1)$-tap FIR
  - $x_L(n;\tau) = (x * d)[n]$

- Example ($N=1$, 2nd order approximation)

$$L_{-1}^{(1)}(t) = \frac{t(t-1)}{2}, \quad L_0^{(1)}(t) = (1-t)(1+t), \quad L_1^{(1)}(t) = \frac{t(t+1)}{2}$$

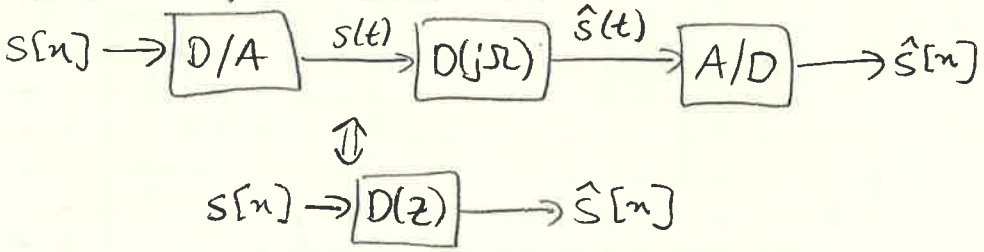
$$d_{0.2}[n] = \begin{cases} -0.08, & n=-1 \\ 0.96, & n=0 \\ 0.12, & n=1 \\ 0, & \text{otherwise} \end{cases} \qquad (\tau = 0.2)$$

- Delay compensation algorithm

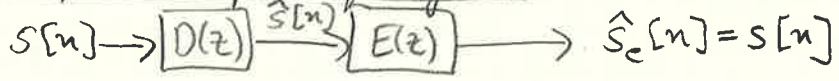
  - estimate the delay τ
  - compute the $2N+1$ Lagrangian coefficients
  - filter with the resulting FIR

6.5.c Adaptive equalization

- Compensating for the distortion

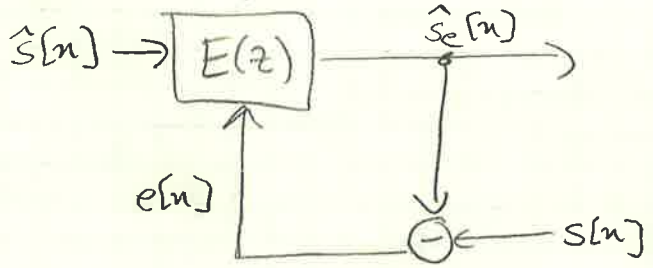


- Example: adaptive equalization

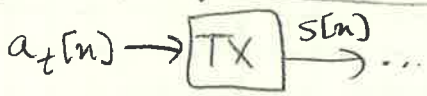


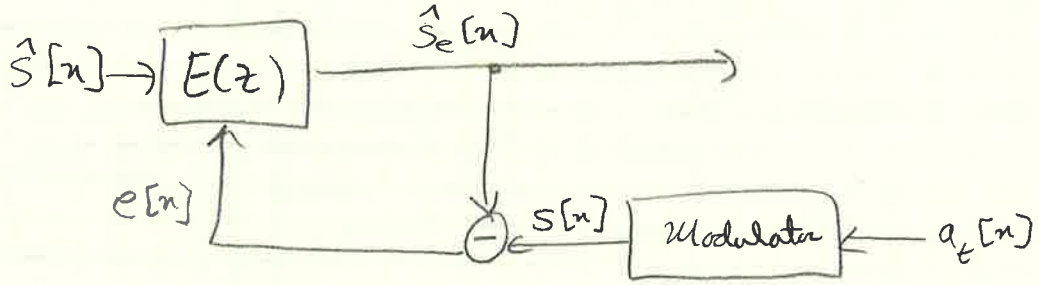
  - in theory, $E(z) = \frac{1}{D(z)}$
  - but we don't know $D(z)$ in advance
  - $D(z)$ may change over time

- Adaptive equalization

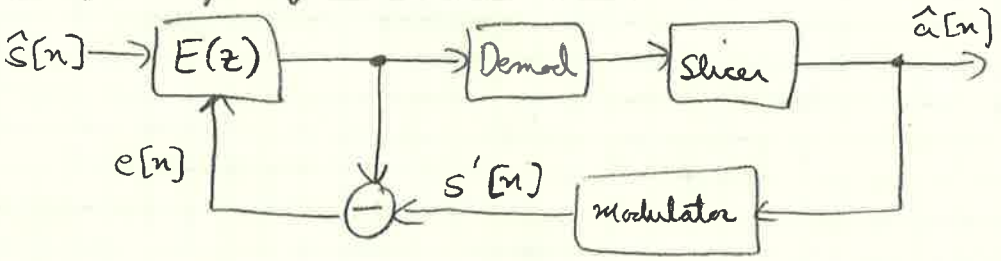


- Adaptive equalization: bootstrapping via a training sequence





- Adaptive equalization: online mode



- So much more to do...

  - how do we perform the adaptation of the coefficients?
  - how do we compensate for differences in clocks?
  - how do we recover from interference?
  - how do we improve resilience to noise?

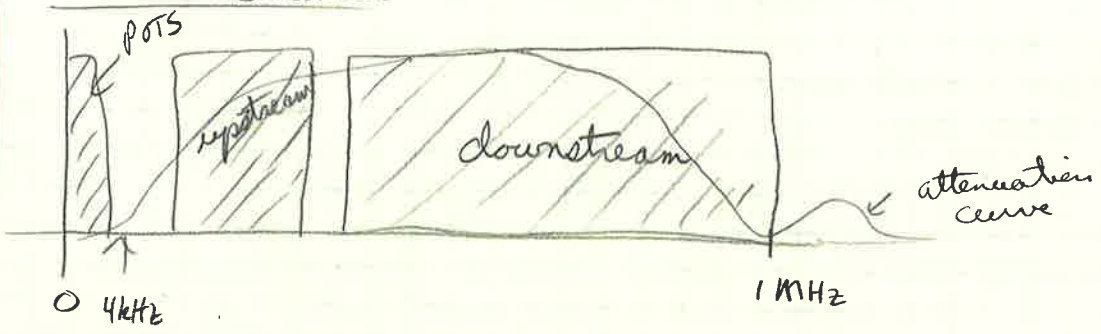
adaptive signal processing

6.6 ADSL

6.6.a ADSL design

- The last mile
  - copper wire (twisted pair) between home and nearest CO
  - very large bandwidth (well over 1 MHz)
  - very uneven spectrum: noise, attenuation, interference, etc.

- The ADSL channel



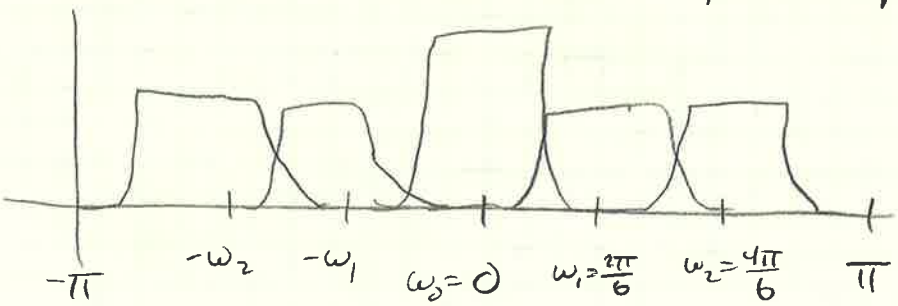
Idea, break into subchannels

- Subchannel structure

- allocate N subchannels over the total positive bandwidth
- equal subchannel bandwidth F_{max}/N
- equally spaced subchannels with center frequency kF_{max}/N, $k=0,\ldots,N-1$

- The digital design

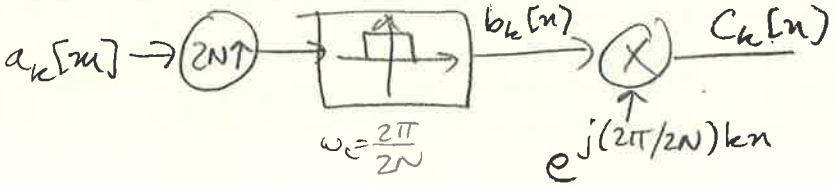
- pick $F_S = 2F_{max}$ (F_{max} is high now!)
- center frequency for each subchannel $\omega_k = 2\pi \dfrac{kF_{max}/N}{F_S} = \dfrac{2\pi}{2N}k$
- bandwidth of each subchannel $\frac{2\pi}{2N}$
- to send symbols over a subchannel: upsampling factor $K \geq 2N$



$(N=3)$

- put a QAM modem on each channel
- decide on constellation size independently
- noisy or forbidden subchannels send zeros

The subchannel modem



$\left(\omega_{c,k} = \dfrac{2\pi}{2N}k\right)$

Then sum them and take $\mathrm{Re}[c[n]] = s[n]$

6.6.6 Discrete multitone modulation

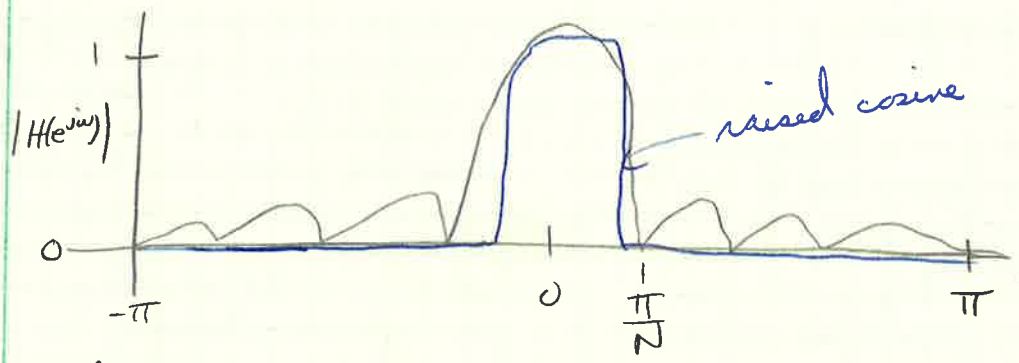
- DMT vs IFFT

  - we will show that transmission can be implemented efficiently via an IFFT
  - Discrete Multitone Modulation

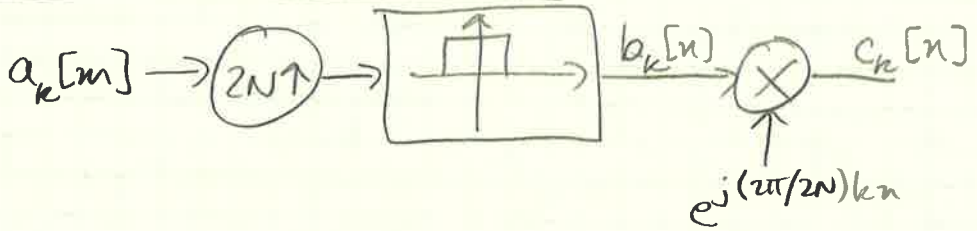
- The great ADSL trick

Instead of using a good lowpass filter, use the 2N-tap interval indicator:

$$h[n] = \begin{cases} 1, & 0 \le n < 2N \\ 0, & \text{otherwise} \end{cases}$$

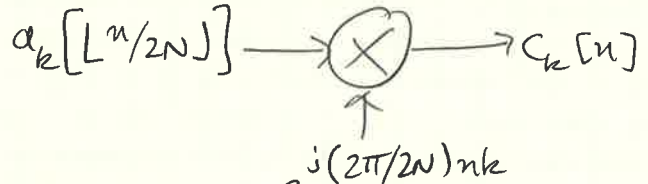


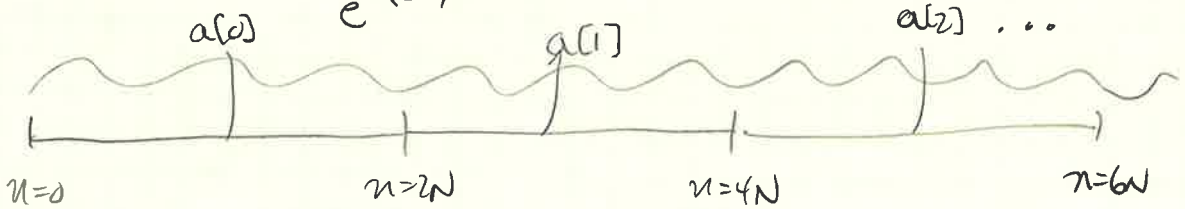
- Back to the subchannel modem



rate: B symbols/sec 2NB samples/sec

by using the indicator function as a lowpass:

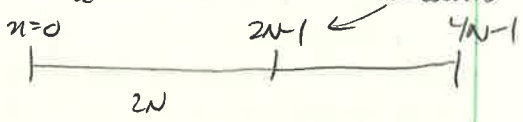




Then, sum to get $c[n]$

- The complex output signal

a_k are constant over intervals



$$c[n] = \sum_{k=0}^{N-1} a_k[\lfloor n/2N \rfloor] e^{j\frac{2\pi}{2N}nk}$$

$$= 2N \cdot IDFT_{2N}\{[a_0[m]\ a_1[m] \dots a_{N-1}[m]\ 0\ 0 \dots 0]\}[n]$$

$$(m = \lfloor n/2N \rfloor)$$

- We can do even better!

- we are interested in $s[n] = Re\{c[n]\} = (c[n] + c^*[n])/2$
- it is easy to prove (exercise) that:

$$IDFT\{[x_0\ x_1\ x_2 \dots x_{N-2}\ x_{N-1}]\}^* = IDFT\{[x_0\ x_{N-1}\ x_{N-2} \dots x_2\ x_1]^*\}$$

- $c[n] = 2N \cdot IDFT\{[a_0[m]\ a_1[m] \dots a_{N-1}[m]\ 0\ 0 \dots 0]\}[n]$
- therefore $s[n] = N \cdot IDFT\{[2a_0[m]\ a_1[m] \dots a_{N-1}[m]\ a_{N-1}^*[m] \dots a_1^*[m]]\}[n]$

($2a_0[m]$: R; $a_{N-1}^*[m] \dots a_1^*[m]$: C)

- ADSL Specs

- $F_{max} = 1104\ kHz$
- $N = 256$
- each QAM can send from 0 to 15 bits per symbol
- forbidden channels: 0 to 7 (voice)
- channels 7 to 31: upstream data
- max theoretical throughput: 14.9 Mbps (downstream)