$$\alpha_k = \frac{\langle v, v_k \rangle}{\langle v_k, v_k \rangle}$$

Ex:

$$v = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad v_1 = \begin{bmatrix} 1/3\sqrt{2} \\ 1/3\sqrt{2} \\ -4/3\sqrt{2} \end{bmatrix} \quad v_2 = \begin{bmatrix} 2/3 \\ 2/3 \\ 1/3 \end{bmatrix} \quad v_3 = \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \\ 0 \end{bmatrix}$$

$\langle v_1 v_2 v_3 \rangle$
$\Downarrow$
orthogonal

$$\alpha_1 = \frac{\left\langle \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1/3\sqrt{2} \\ 1/3\sqrt{2} \\ -4/3\sqrt{2} \end{bmatrix} \right\rangle}{1} = -2/3\sqrt{2}, \quad \alpha_2 = 5/3, \quad \alpha_3 = 0$$

$$\boxed{v = -\frac{2}{3\sqrt{2}} v_1 + \frac{5}{3} v_2 + 0 \cdot v_3}$$

DISCRETE FOURIER TRANSFORM →

$$v_k[n] = e^{j\frac{2\pi k n}{N}}, \quad n \in [0, \dots N-1]$$

$$v_0 = [1 \quad 1 \quad 1 \quad \dots \quad 1]$$

$$v_1 = \left[1 \quad e^{\frac{2\pi j}{N}} \quad e^{\frac{4\pi j}{N}} \quad \dots \quad e^{\frac{2\pi (N-1)}{N} j}\right]$$

$$\langle v_k, v_l \rangle = \sum_{n=0}^{N-1} v_k(n) \cdot \bar{v}_l(n)$$

$$= \sum_{n=0}^{N-1} e^{j\frac{2\pi k n}{N}} \cdot e^{-j\frac{2\pi l n}{N}}$$

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

$$\Rightarrow \langle v_k, v_l \rangle = \sum_{n=0}^{N-1} e^{\frac{j 2\pi n}{N}(k-l)}$$

① $K = l$;

$$\langle v_K, v_l \rangle = \sum_{n=0}^{N-1} 1 = \underline{N}$$

② $K \neq l$;

$$= e^{j2\pi 0 \frac{(K-l)}{N}} + e^{j2\pi 1 \frac{(K-l)}{N}} + \cdots + e^{j2\pi \frac{(N-1)(K-l)}{N}}$$

$$= \frac{1 - \left(e^{j2\pi \frac{K-l}{N}}\right)^{N}}{1 - e^{j\frac{2\pi(K-l)}{N}}} = 0$$

(*) $$x = \sum_{K=0}^{N-1} X_K v_K \qquad X_K = \frac{\langle x, v_K \rangle}{\langle v_K, v_K \rangle}$$

$$x = X_0 v_0 + X_1 v_1 + X_2 v_2 \cdots + X_{N-1} v_{N-1}$$

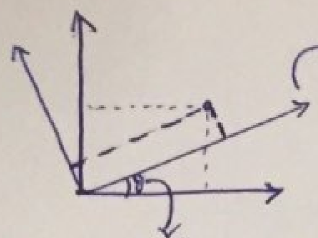
$$x[n] = \sum_{K=0}^{N-1} X_K \cdot e^{j\frac{2\pi Kn}{N}}$$

$$x_K = \frac{1}{N} \sum_{K=0}^{N-1} x[n] \cdot e^{-j\frac{2\pi Kn}{N}}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} = \frac{1}{N} \begin{bmatrix} z^0 & z^0 & z^0 & z^0 & \cdots \\ z^0 & z^1 & z^2 & & z^{N-1} \\ \vdots & & & & \\ z^0 & z^{N-1} & z^{2(N-1)} & \cdots & z^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}$$

$$e^{-j\frac{2\pi Kn}{N}} = z^{Kn} \Rightarrow z = e^{-j\frac{2\pi}{N}}$$

$$\Rightarrow \boxed{z = e^{-j\frac{2\pi}{N}}}.$$



transforms of a basis

≡ Euclidean space

forms a basis

$$X_K = \frac{1}{N}\sum_{n=0}^{N-1} x[n].\,e^{-j\frac{2\pi K n}{N}} \qquad \left(\text{Discrete Fourier Transform}\right)$$

$$x(n) = \sum_{K=0}^{N-1} X_K \cdot e^{j\frac{2\pi K n}{N}}$$

Ex: Let the signal be ≡ $(N=4)$

$$x(t) = 5 + 2\cos\left(2\pi t - \frac{\pi}{2}\right) + 3\cos \underbrace{4\pi t}_{2\pi(2)t}$$

$f_{max} = 2\ Hz$

Sampling frequency $= 4\ Hz$, $t = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$

$x[0] = 5 + 2\cos\left(-\frac{\pi}{2}\right) + 3\cos 0 = 8$

$x[1] = 5 + 2\cos\left(\frac{\pi}{2} - \frac{\pi}{2}\right) + 3\cos\pi = 4.$

$x[2] = 5 + 2\cos\frac{\pi}{2} + 3\cos 2\pi t = 8$

$x[3] = 5 + 2\cos\pi + 3\cos 3\pi = 0.$

$x[n] = \{8, 4, 8, 0\}$

$$\frac{1}{4}\begin{bmatrix} 20 \\ -4j \\ 12 \\ 4j \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}\begin{bmatrix} 8 \\ 4 \\ 8 \\ 0 \end{bmatrix} \begin{array}{l} k=0. \\ k=1 \\ k=2 \\ \ \end{array}$$

roots of unity $= x^{1/4} = 1.$

0 1Hz 2Hz → conjugate of 1 => |j| + |-j| = 2

Ans ≡ [5 -j 3 j] $\boxed{X[K] = X^*[N-K]}$

sinusoid of 1Hz amplitude ≡ 2

DC Value

sinusoid of 2Hz where amplitude is 3.

since it is sampled at 4HZ, Max freq } = 2Hz

Since it is complex; there may be a phase shift in the 1Hz signal. $[-\pi/2]$

First Value ≡ Mean of the signal. [No frequency term] DC Value.

magnitude of 1Hz frequency ≡ distributed between the two values [-j, j]

> Length of the signal, {8, 4, 8, 0} ≡ 4.

> 2Hz value => It is real; No phase shift
Amplitude = 3.
Frequency = 2Hz.
} By convention it is "cos"

frequency vs time (just like Heisenberg's Uncertainity Principle).

If we take more samples ≡ frequency shifts accordingly

CONTINUOUS FOURIER TRANSFORM [FT]

$$X(f) = \int_{-\infty}^{\infty} x(t) . e^{-j2\pi f t} \, dt$$

$$X(\omega) = \int_{-\infty}^{\infty} x(t) . e^{-j\omega t} \, d\omega.$$

① $\delta(t)$.

$$X(f) = \int_{-\infty}^{\infty} \delta(t) . e^{-j2\pi f t} \, dt \quad ; \text{ at } t=0; 1$$

$X(f) = 1$

Scanned by CamScanner

② $\delta(t-t_0)$

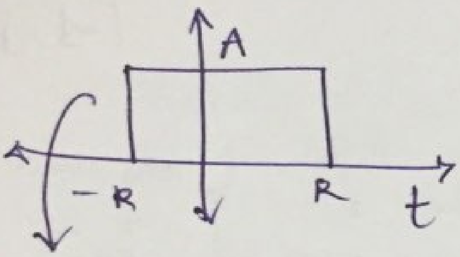
$$X(f) = \int_{-\infty}^{\infty} \delta(t-t_0).\, e^{-j2\pi f t}\, dt \quad , \text{at } t=t_0 , \overset{\delta(t-t_0)}{\curvearrowright} 1$$

$$= e^{-j2\pi f t_0} = e^{-j\omega t_0}.$$

③ $\delta(t-t_0) + \delta(t+t_0)$ (2 symmetric impulse responses)

$\hookrightarrow e^{-j\omega t_0} + e^{j\omega t_0} = 2\cos\omega t_0$

④



box function

"can be reduced from $(-\infty, \infty)$ to $(-R, R)$".

$$\equiv \int_{-\infty}^{\infty} x(t).\, e^{-j\omega t}.\, dt$$

$$\equiv \int_{-R}^{R} A.\, e^{-j\omega t}\, dt$$

$$\equiv A \int_{-R}^{R} e^{-j\omega t}\, dt$$

$$\equiv \frac{A}{-j\omega} \left. e^{-j\omega t} \right]_{-R}^{R}$$

$$\equiv \frac{A}{-j\omega} \left[e^{-j\omega R} - e^{j\omega R} \right]$$

$$\equiv \frac{A}{j\omega} \left[e^{j\omega R} - e^{-j\omega R} \right]$$

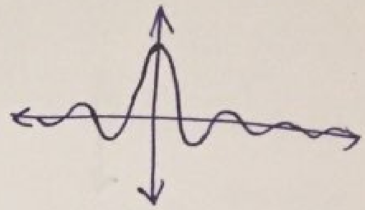
$$\equiv \frac{A}{j\omega} \left[2j\sin\omega R \right] \equiv \frac{2A}{\omega} \sin\omega R$$

$$\equiv 2AR. \left(\frac{\sin\omega R}{\omega R} \right)$$

$$\equiv 2AR\underbrace{\left(\frac{\sin \omega R}{\omega R}\right)}$$

$$\equiv 2AR \text{ sinc } (\omega R)$$

$\text{sinc}(a) = \frac{\sin a}{a}$



light – periodic waves.

FAST FOURIER TRANSFORM →

dft → takes a lot of time to do the large matrix multiplication
vs
fft ↓ very fast

$$X_k = \frac{1}{N}\sum_{n=0}^{N-1} x[n] \cdot e^{-j \cdot \frac{2\pi K n}{N}}$$

(1.) $K = \frac{N}{2}K_1 + K_0$; $K_1 = 0$ (or) 1

$0 \leqslant K_0 \leqslant \frac{N}{2} - 1$

$K_1 = 0$ – first half of the signal K

$K_1 = 1$ – next half of the signal K

$$X_K = \frac{1}{N}\sum_{n=0}^{N-1} x[n] \cdot e^{-2\pi j\left(\frac{N}{2}K_1 + K_0\right)\frac{n}{N}}$$

$$= \frac{1}{N}\sum_{n=0}^{N-1} x[n] \cdot e^{-\frac{2\pi j N K_1 n}{2N}} \cdot e^{-2\pi j \frac{K_0 n}{N}}$$

$$= \frac{1}{N}\sum_{n=0}^{N-1} x[n] \cdot e^{-\pi j K_1 n} \cdot e^{-2\pi j \frac{K_0 n}{N}}$$

$\therefore e^{-\pi j} = -1$

$$= \frac{1}{N}\sum_{n=0}^{N-1} x[n] \cdot (-1)^{K_1 n} \cdot e^{-2\pi j \frac{K_0 n}{N}}$$

↓

divide into even & odd indices

$$X_K = \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1} x[2n] \cdot e^{-\frac{2\pi j K_0 (2n)}{N}} \cdot \underbrace{(-1)^{K_1(2n)}}_{1} + \sum_{n=0}^{N/2-1} x[2n+1] \cdot e^{-2\pi j \frac{K_0 2n+1}{N}} \cdot \underbrace{(-1)^{K_1(2n+1)}}_{(-1)^{K_1}}$$

$$X_K = \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1} x[2n] \cdot e^{-\frac{2\pi j K_0 n}{N/2}} + \sum_{n=0}^{\frac{N}{2}-1} x[2n+1] \cdot e^{-2\pi j \frac{K_0(2n+1)}{N}} (-1)^{K_1}$$

$$X_K = \frac{1}{N} \sum_{n=0}^{\frac{N}{2}-1} x(2n) \cdot e^{-\frac{2\pi j K_0 n}{N/2}} + (-1)^{K_1} \underbrace{e^{-\frac{2\pi j K_0}{N}}}_{\rightarrow Q} \sum_{n=0}^{N/2-1} x(2n+1) \cdot e^{-\frac{2\pi j K_0 n}{N/2}}$$

$\downarrow$ computing just the fourier transforms of half signals.

$$X_K = F_0 + (-1)^{K_1} Q F_1$$

$$X_{K_1=0} = F_0 + QF_1$$

$$X_{K_1=1} = F_0 - QF_1$$

$f(x) =$ $x[0]$ $x[1]$ $x[2]$ . . . . . $x[N-2]$ $x[N-1]$

F_0 $\quad$ F_1 $\quad$ $2T_{M/2}$

$F(u) =$ $\quad$ $F_0 + MF_1$ $\quad$ $F_0 - MF_1$ $\quad$ $2M$

Complexity $\equiv$ $T(N) = 2\,T(N/2) + 2N$

$\downarrow$

$O[N \log N]$

$\downarrow$

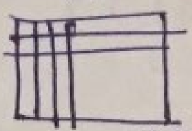
FFT Computation times

2D FOURIER TRANSFORM →

physical Intution?

$$X = W_N x^T \quad (1D \text{ case})$$

A



MxN matrix

row wise ← fourier transform

& then the output is applied fourier transform column wise.

$$\hat{A} = W_{M\times M}\left(A_{M\times N}\, W_{N\times N}\right)$$

if it rotates, it changes
but on transleting, it doesnt change.

CONVOLUTION THEOREM →

$$y(n) = \sum_{K=0}^{N-1} x[k]\,.\,h[n-k]$$

$x[n] = [\underset{\uparrow}{1}\ 2\ 0\ 1]$ $\qquad h[n] = [\underset{\uparrow}{2}\ 2\ 1\ 1]$

1 2 0 1 = 0 2 6 5 5 4 1 1 0 $*$ ← convolution

1 1 2 2

Circuler Convolution : $y[n] = \sum_{K=0}^{N-1} x[k]\,.\,h\,[\underbrace{(n-K) \bmod N}_{?\ \text{what is happening}}]$

$$y[0] = x[0].h[0] + x[1]h[3] + x[2]h[2] + x[3]h[1] = 6$$

$$y[1] = x[0].h[1] + x[1].h[0] + x[2]h[3] + x[3]h[2] = 7$$

1 2 0 1

2 1 1 2
2 2 1 1
1 2 2 1
1 1 2 2

$\circledast$ = 6 7 6 5

↑ Circular Convolution

If $x_K = x_{K \bmod N}$ (a repeating signal)

2 2 1 1

Normal convolution

... 1 2 0 1 1 2 0 1 1 2 0 1 1 2 0 1 ...

1 1 2 2

5 $\boxed{6\ 7\ 6\ 5}$ 6 7 6 5 .

For Ex: Padding the signals with zeros and doing a circular convolution. — to result in the size of "output"

$h[n]$

$[2\ 2\ 1\ 1\ 0\ 0\ 0]$

0 6 5 4 3 2 1

circular convolution

	1	2	0	1	0	0	0
2 ⇐	2	0	0	0	0	1	2
6 ⇐	2	2	0	0	0	1	1
5 ⇐	1	2	2	0	0	0	1
	1	1	2	2	0	0	0

results in

→ linear convolution output.

(Discrete Domain)

$y[n] = x[n] \circledast h[n]$ [∴ Circular Convolution]

$y[n] = F^{-1}(F(x[n]).F(h[n]))$ ⇒ DFT is used.

$y[n] = x[n] * h[n]$ [∴ Convolution]

$y[n] = F^{-1}(F(x(n)).F(h(n)))$ → (Continuous Domain)

FT is used

CFT : $Y(n) = \int_{-\infty}^{\infty} y(n).\, e^{-j\omega n}\, dn$

$= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} x(k)\, h(n-k)\, dk \right) e^{-j\omega n}\, dn$

$n - k = m$

$$= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} x(k) . h(m)\, dk \right) e^{-j\omega(k+m)}\, dm$$

$u - k = m$

$du = dm$

$$= \int_{-\infty}^{\infty} h(m) . e^{-j\omega m}\, dm . \int_{-\infty}^{\infty} x(k) . e^{-j\omega k}\, dk .$$

$$= FT(x) . FT(h) .$$

Discrete Domain:

(circular) x mod m)

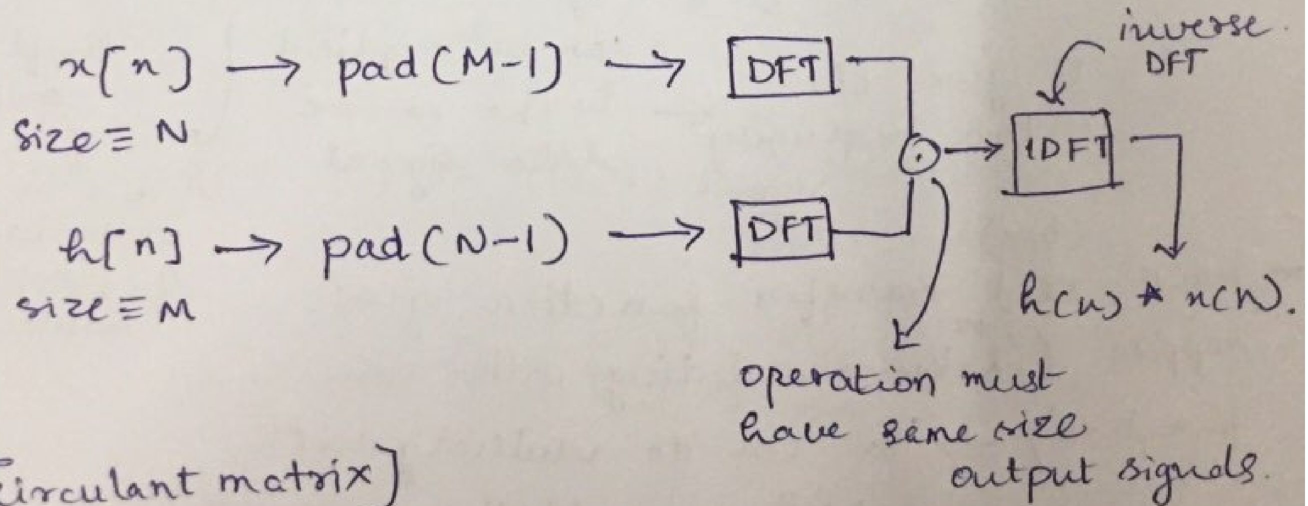
$$y(n) = \sum_{r=0}^{N-1} x(r) . h(n-r) \quad \text{(convolution formula).}$$

$$Y_K = \frac{1}{N} \sum_{u=0}^{N-1} y(n) . e^{-j\frac{2\pi K n}{N}} \qquad n - r = m .$$

$$= \sum_{u=0}^{N-1} \left(\sum_{r=0}^{N-1} x(r) . h(n-r) \right) . e^{-j\frac{2\pi K(u)}{N}}$$

$$= \sum_{u=0}^{N-1} \left(\sum_{r=0}^{N-1} x(r) . h(m) \right) e^{-j\frac{2\pi K(m+r)}{N}}$$

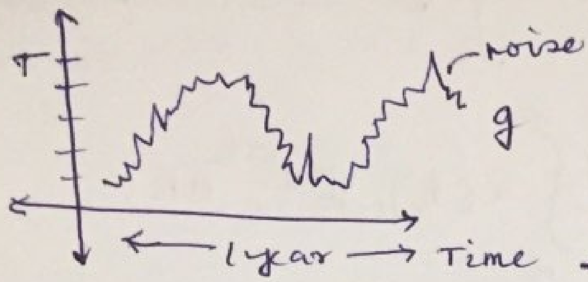
$$= \sum_{u=0}^{N-1} h(m) . e^{-j\frac{2\pi K m}{N}} \sum_{r=0}^{N-1} x(r) . e^{-j\frac{2\pi K r}{N}} .$$



[circulant matrix]

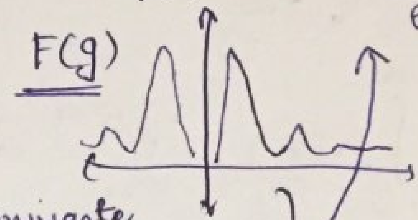
Transfer Function?

Real Life Examples: – Turbidity of lake (data)



FFT

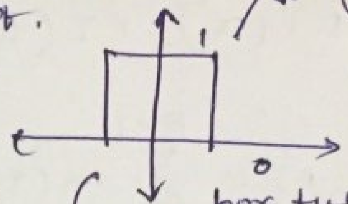
makes the high freq zero



→ Conjugate should be taken care of.

* (point to point multiplication)

☆



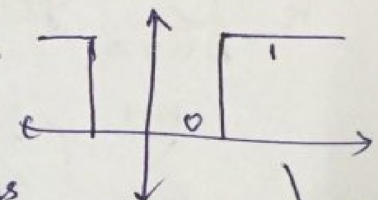
Fourier transform of ~~these~~ is similar to convolution with sinc

– low pass filtering

box function transfer function.

– Band pass filtering

fixed band width size.



– high pass filtering (removing the lower frequencies).

inverse fourier

you'll get the ~~noise~~ – ~~less~~ filtered Signal.

Moving Average Filter

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

Ex: $\left[\frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5}\right]$

can be applied to the above data signal

to give the low frequency signal.

– shift occurs due to the contribution of previous values.

⇒ Since (box) rect transfer function gives ripples (just like convoluting with sinc function) ⇒ we can do multiply with an "exponentially decaying function"

⇓

gaussian function

pepper noise = max filter
salt noise = min filter
salt and pepper noise = median filter

Leaky Integrator -

M previous values

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

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

$K+1 = k$

$* \; y_M[n-1] = \frac{1}{M}\sum_{K=1}^{M} x(n-K)$.

$\rightarrow y_{M-1}[n] = \frac{1}{M-1}\sum_{K=0}^{M-2} x(n-K)$

$\rightarrow y_{M-1}[n-1] = \frac{1}{M-1}\sum_{K=0}^{M-2} x(n-1-K)$.

$= \frac{1}{M-1}\sum_{K=1}^{M-1} x(n-K)$.

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

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

from above;

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

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

let $\frac{M-1}{M} = \lambda$ $\quad 1 - \frac{1}{M} = \lambda \Rightarrow \frac{1}{M} = 1 - \lambda$

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

Suppose M is very Large, $\lambda \approx 1$ & that results in
(window is large)

$y_M[n-1] = y_{M-1}[n-1]$

mean of m values is equal to mean of m-1 values.

[just the position]

Leaky Integrator

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

* current value * weight + Previous (value) Mean = mean.

Though you're integrating the values, it gives slight weightage to the current value (there's a leak).

used for Analog signals.

Ex: $y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x(n-k)$.

Impulse Response $\longrightarrow$ $h[n] = \frac{1}{M} \sum_{k=0}^{M-1} \delta(n-k)$.

for $n=0$; $k=0 \Rightarrow \frac{1}{M}$

$n=1$; $k=1 \Rightarrow \frac{1}{M}$.

$$h[n] = \begin{cases} \frac{1}{M} & ; 0 \leq n < M \\ 0 & ; \text{otherwise} \end{cases}$$

FIR filter (finite values).

{"Finite Impulse Response".}

leaky integrator — impulse response ?

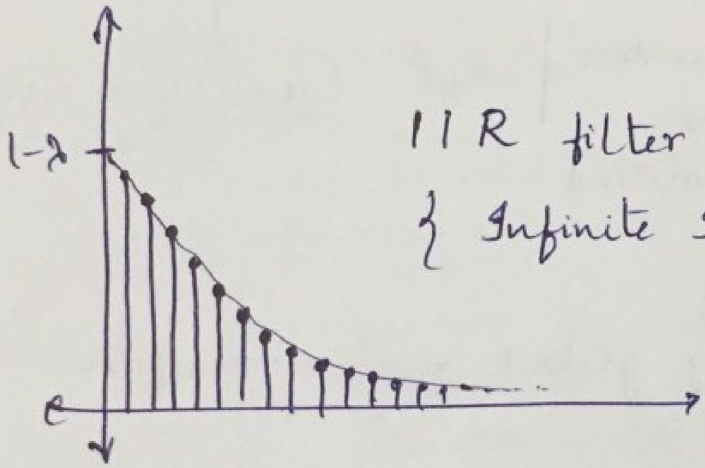
$$h[n] = (1-\lambda)\,\delta(n) + \lambda\, h[n-1]$$

for Ex: $n=0;\quad h[0] = (1-\lambda)\,\delta(0) + \lambda \underbrace{h[-1]}_{0} = 1-\lambda$

$$h[1] = (1-\lambda)\,\delta(1) + \lambda\, h[0] = \lambda(1-\lambda)$$

$$\vdots \qquad\qquad \vdots$$

$$h[n] = \begin{cases} 1-\lambda & ; n=0 \\ \lambda(1-\lambda) & ; n=1 \\ \vdots & \\ \lambda^{K}(1-\lambda) & ; n=K \end{cases}$$



IIR filter

{ Infinite Impulse Response }

Leaky Integrator ≡ IIR filter.

$$\sum_{k=0}^{p-1} a_k\, x(n-k) = \overbrace{\sum_{r=0}^{q-1} b_r\, y(n-r)}^{\text{CDE}}$$

leaky integrator can be expressed like this.

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

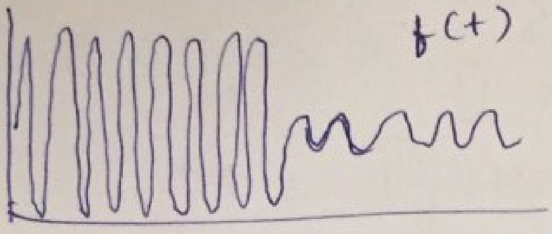
{ λ shouldn't be too high, as the current value contribution decreases. }

Short Time Fourier Transform:

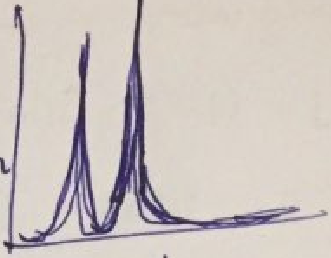
For Ex:

$$g(t) = \begin{cases} 2 \times \sin(2\pi \times 39t) & , 0<t<1/2 \\ \sin(2\pi \times 15t) & , 1/2<t<1 \end{cases}$$

f(t)

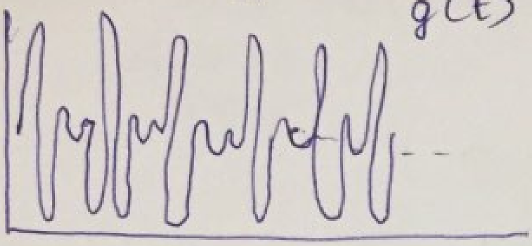


$\xrightarrow[\text{transform}]{\text{Fourier}}$

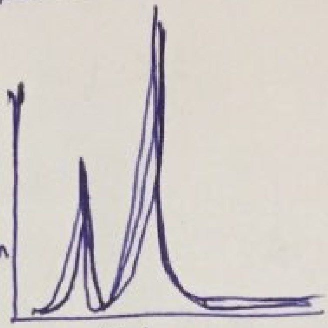


or

g(t)



$\xrightarrow[\text{transform}]{\text{Fourier}}$



f(t) & g(t) = though are different in time domain

↓

give same Fourier

↓

<u>GLOBAL</u> IN NATURE

Telephones, on dialling numbers, different frequencies are (very carefully chosen) transmitted | Why?

coprime numbers

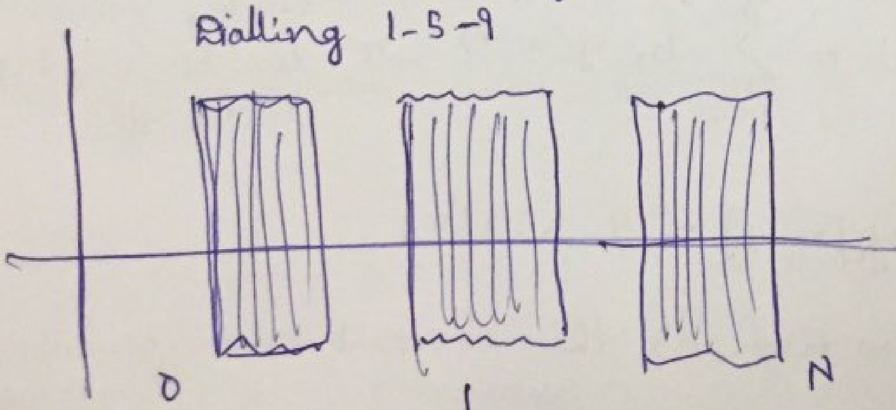
↓

On taking global fourier transform

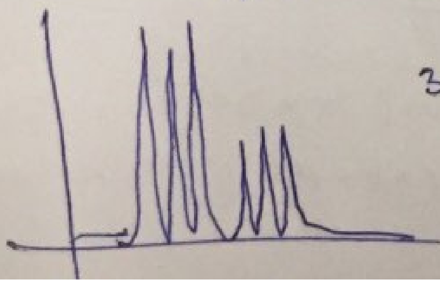
— Limitations —

— order of dialling } can't be
— Number of dials } figured out.

Dialling 1-5-9



↓ FFT



3 pairs for 3 ranges.

To resolve this issue ; → sliding window fourier transform.

but we lose on "resolution in fourier transform"

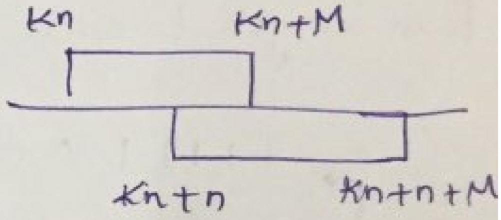
intervals – less, frequency is roughly calculated.
(samples are less)

$\boxed{\text{uncertainity: } \Delta t \cdot \Delta f = 2\pi}$

resolution in time & frequencies

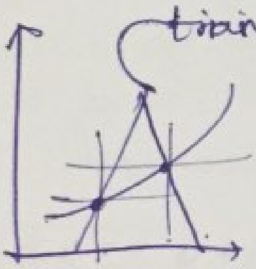
Windowing → break the signal into blocks "windows" in time domain.

↓ Apply DFT to each window independently



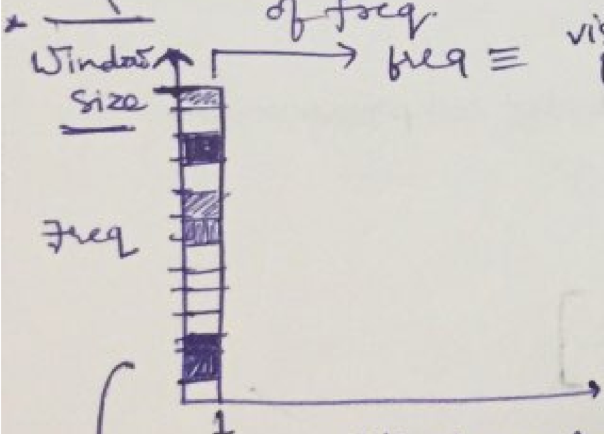
Adjacent windows may overlap.

n = M
(No overlap).



triangular window can be used [instead of rect]

* gaussian is not advisable for ~~and~~ analog



freq ≡ visualised in terms of colour | as per each window.

→ graph ≡ Spectrogram.
↓
can be used in Music also

Lighter ≡ ↑ DFT Magnitude
Darker ≡ ↓ DFT Magnitude

Compression →

1 movie (2hrs) = $2 \times 60 \times 60 \times 30 \times 1920 \times 1080 \times 24$ = 1.074×10^{13} = 10TB

What parameters can be reduced to reduce the size?

How come you are able to store it on the computer?.

- temporal redundancy
- spatial redundancy
- $\boxed{\text{perceptual redundancy}}$ ✓ frames that are not at all perceived.

Run length Encoding → RLE

1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0.

(1, 5) (0, 8) (1, 5) (0, 2).

Encoding

W W W W A A A S S S B S S 13 × 8

W W 4 A A 3 S S 3 B S S 2

[W A S B S] 5 × 8
[4 3 3 1 2] 5 × 8

Signal with high entropy – hard to compress.

Huffman Coding –

[Idea: smaller encoding – more frequent characters]

Ex:

AABRAA_KA_DAABRAA!!!

20 X 8 bit encoding

frequency:	A	!	_	B	R	D	K	F
	9	3	3	2	2	1	1	

min: add them up

Variable Length Encoding

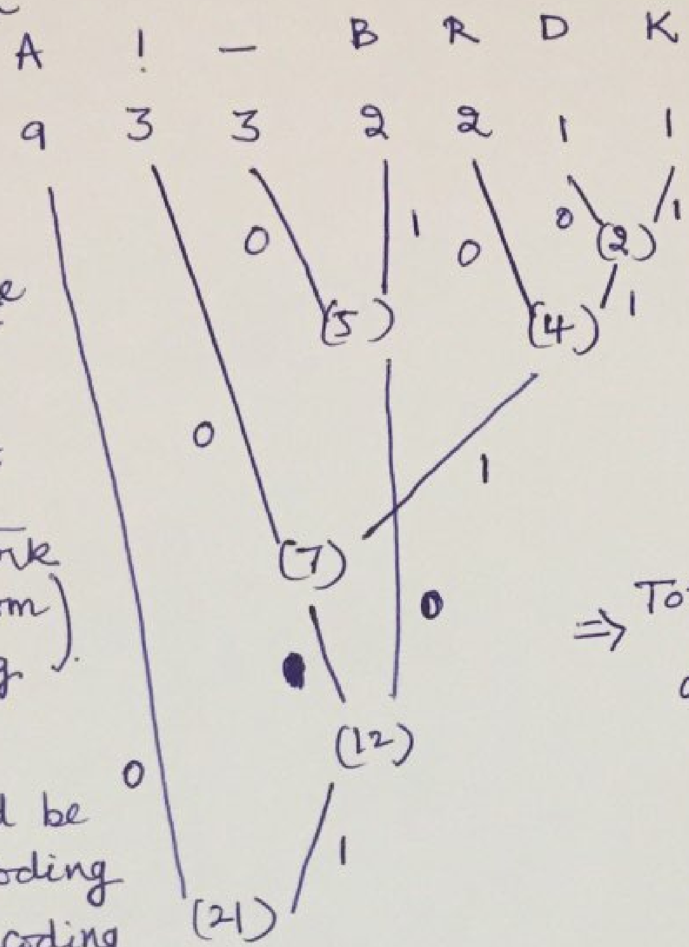
↓

structure to ~~decrypt~~ decode back is important

(doesnt work for random encoding).

↓

it should be prefix encoding as one encoding should be within the other encoding.



* prefix encoding
* dictionary coding

↓ to decode it back.

A – 0
! – 110
_ – 100
B – 101
R – 1110
K – 11110
D – 11111.

⇒ Total Encoding =

$9\times1 + 3\times3 + 3\times3 + 2\times3 + 2\times4 + 1\times5 + 1\times5$

= 51 bits.

⇒ To encode, we need the dictionary

Initial = 21×8, since there are only 7 characters we can use 3 bits only so, possible bits = 21×3.

Compression: $\frac{21\times3}{51}$.

"frequentist view coding"

↓

only looking at the frequency of the characters.