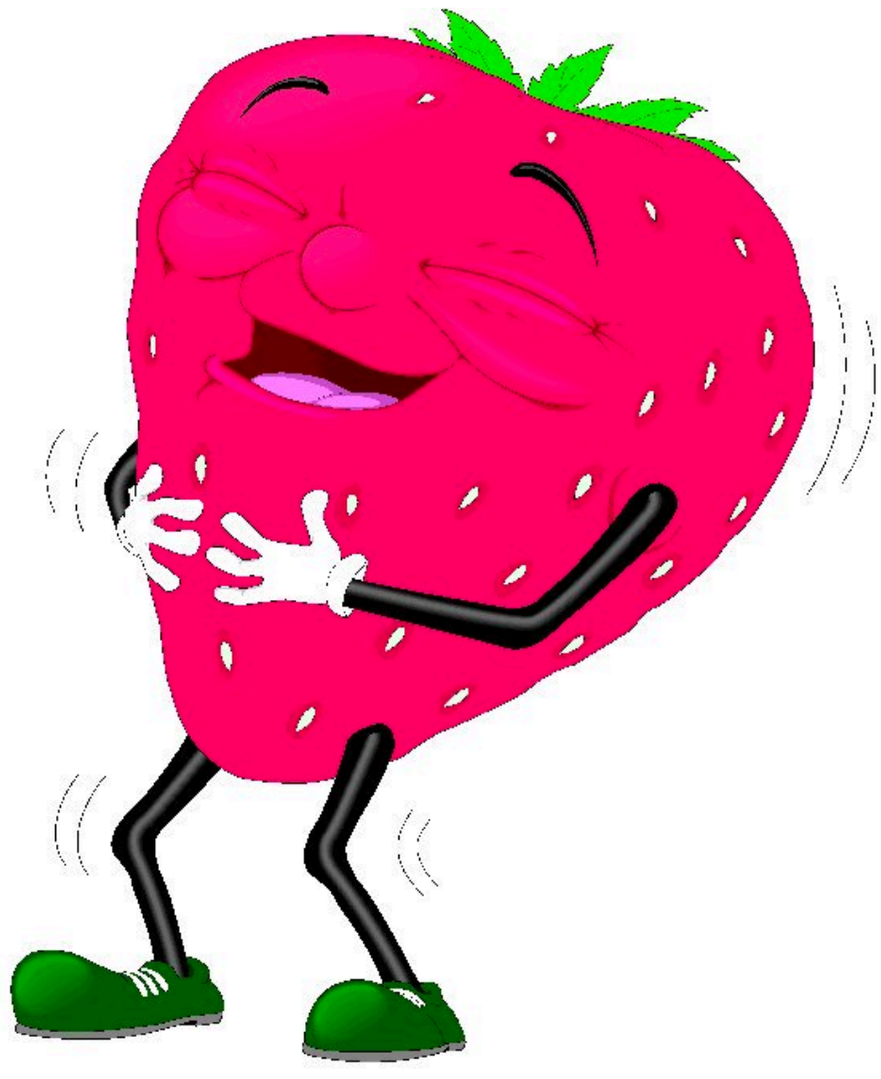
TIMESERIES 4

10.28.2020

PROBLEM SET 4

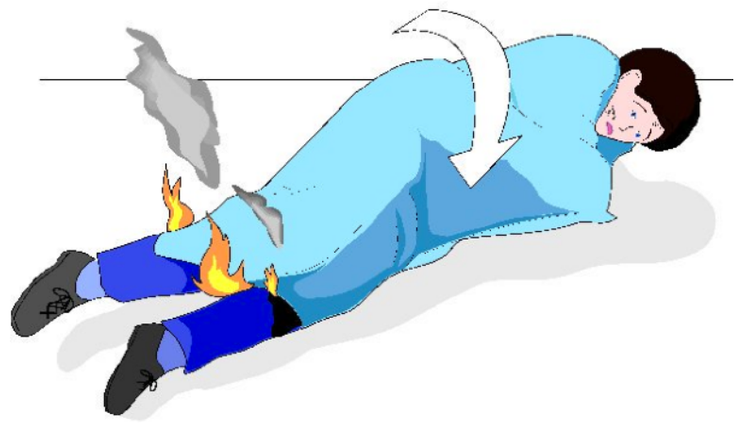
* is due Friday!



PROBLEM SET 5

* will be posted Friday

* but you will have **3 weeks** to finish it



RECAP: SPECTROGRAM

* if we compute the fourier transform for small snippets of time and then stack them together into an array

* this is the **spectrogram**

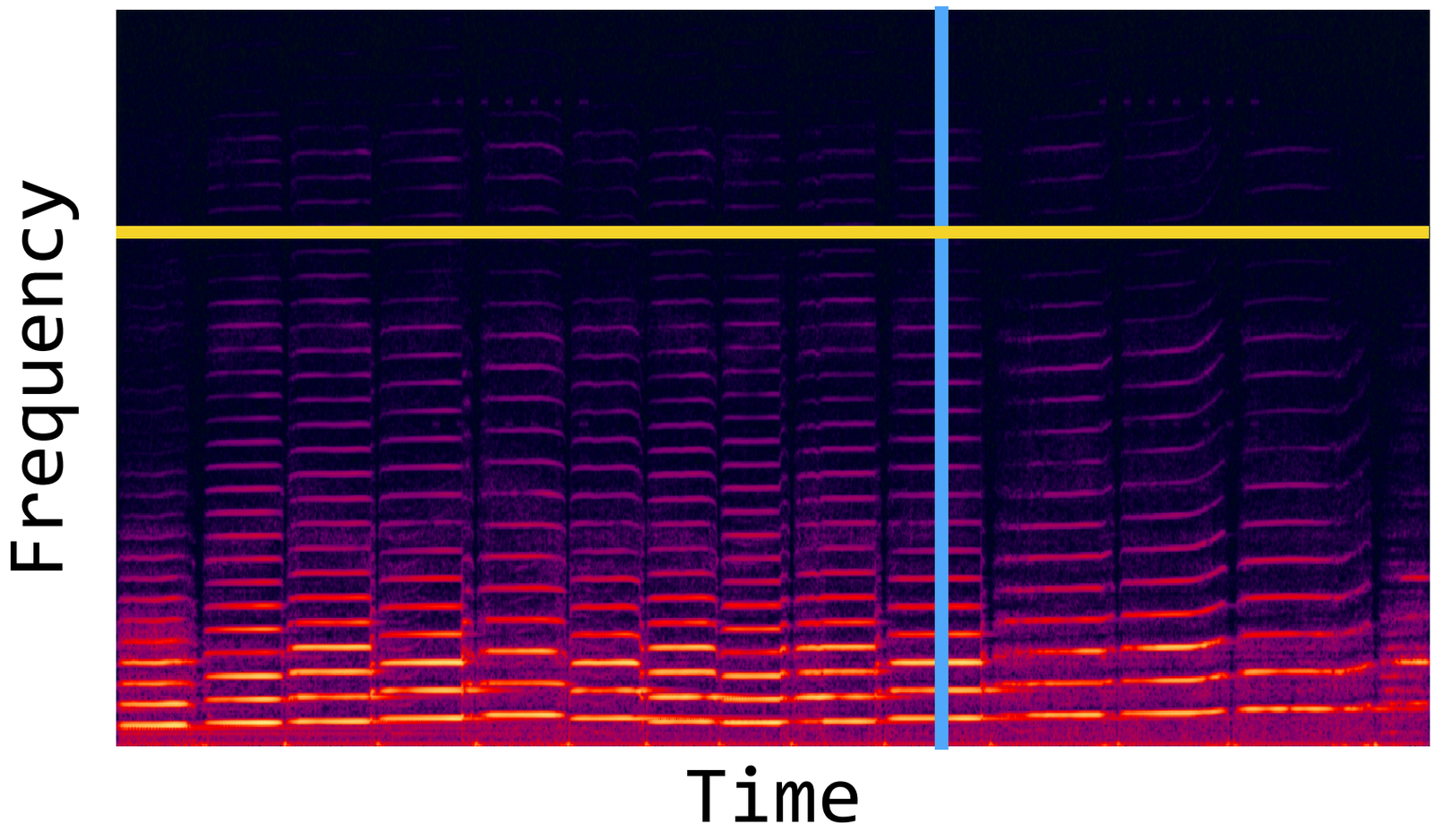
* it shows which frequencies are present in a timeseries at each point in time

* *you should know how to read a spectrogram*

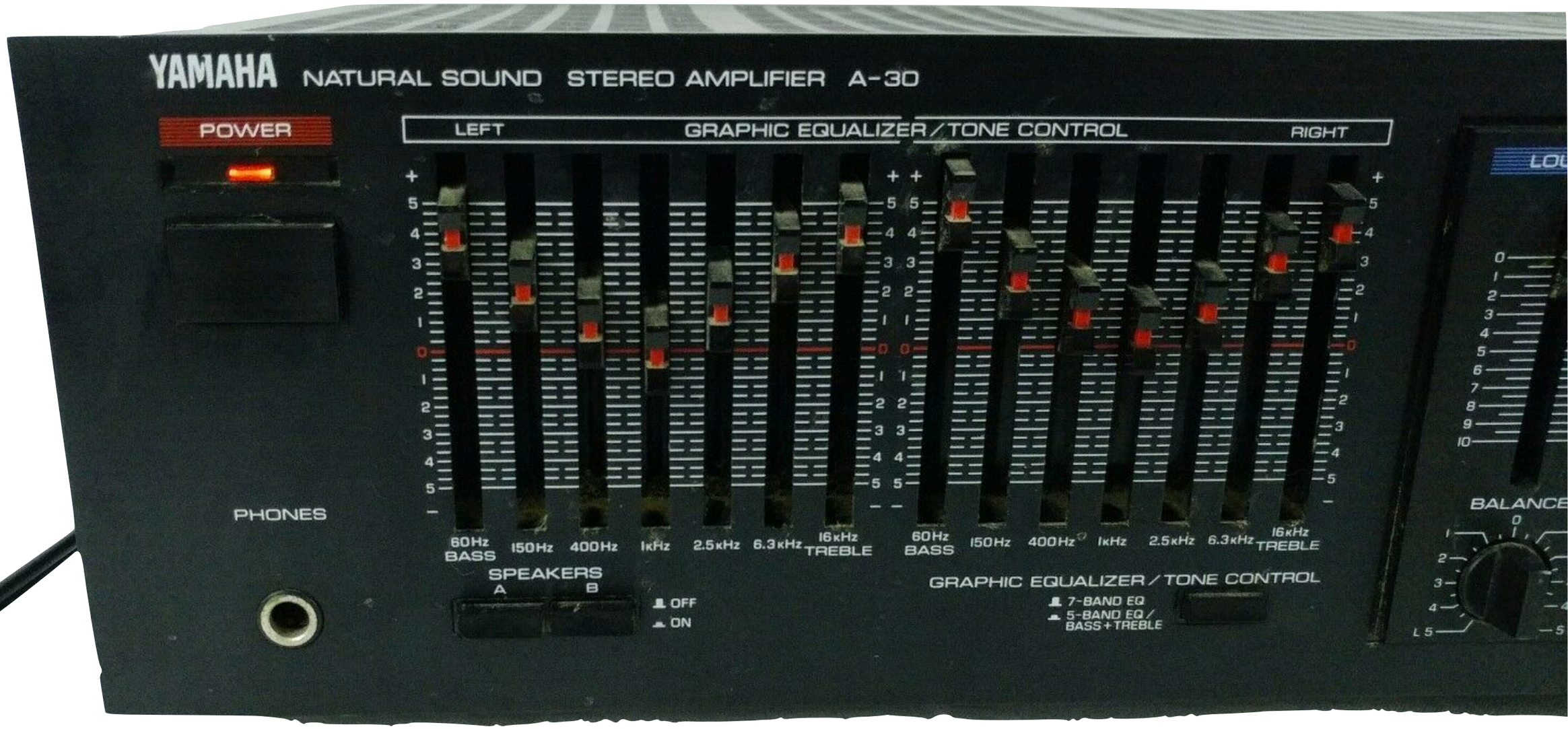
THE SPECTROGRAM

* each **column** is the fourier transform of a short snippet

* what about each **row**? what does one row mean?



FILTERING


YAMAHA
NATURAL SOUND STEREO AMPLIFIER A-30
POWER
LEFT
GRAPHIC EQUALIZER / TONE CONTROL
RIGHT
PHONES
60Hz BASS
150Hz
400Hz
1kHz
2.5kHz
6.3kHz
16kHz TREBLE
SPEAKERS
A
B
OFF
ON
GRAPHIC EQUALIZER / TONE CONTROL
7-BAND EQ
5-BAND EQ / BASS + TREBLE
BALANCE

FILTERING

* **filtering** is a process that removes some frequencies from a timeseries and lets others remain (or even amplifies them)

* this is accomplished by convolving your timeseries with a **filter**, a small array that is designed to have a specific effect

FILTERING

* **low-pass filter:** removes high frequencies, allows low frequencies through

* **high-pass filter:** removes low frequencies, allows high frequencies through

* **band-pass filter:** removes all frequencies except for a specific band (the “pass band”)

FILTERING

* **low-pass filter:** removes high frequencies, allows low frequencies through

* **high-pass filter:** removes low frequencies, allows high frequencies through

* **band-pass filter:** removes all frequencies except for a specific band (the “pass band”)

FILTERING

* back to the spectrogram:
* one row of a spectrogram is a lot like a **band-pass filtered** version of a timeseries

FILTERING

* suppose we have some EEG data from a human subject and we want to filter it so that only alpha-band oscillations remain

  * (this is a band-pass filter)

* how do you make a filter that has the properties you want?

FILTERING

* **scipy.signal** is a module in scipy that contains lots of useful functions for filter design

* **scipy.signal.firwin** creates “finite impulse response” filters with desired properties

ANALYZING A FILTER

* **scipy.signal.freqz** is a great function that tells you what the *frequency response* of your filter looks like

* i.e. it tells you what the filter is going to do to your signal

RECALL:
FOURIER ANALYSIS

* fourier transforms have an interesting property related to convolution:

* given two timeseries, f and g, the fourier transform of their convolution = the element-wise product of their fourier transforms

$$\mathrm{FT}(f \star g) = F \cdot G$$

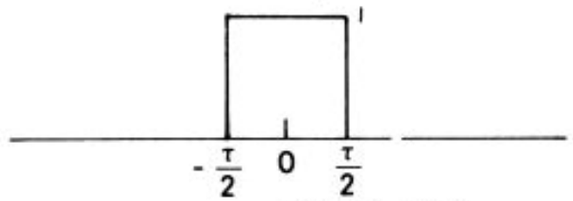
* the reverse is also true:

$$F \star G = \mathrm{FT}(f \cdot g)$$

Time Function	Frequency Function

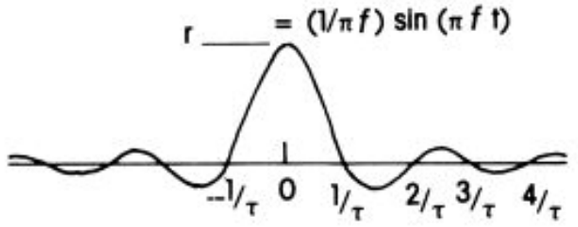
Boxcar

$$G(t) = \begin{cases} 1, & |t| < \tau/2 \\ 0, & |t| > \tau/2 \end{cases}$$



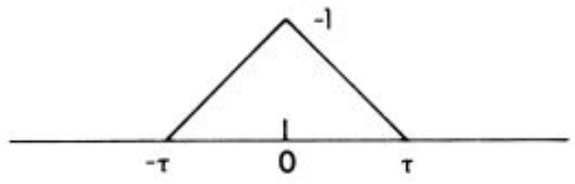
Sinc

$$S(f) = \tau \,\mathrm{sinc}\,(f\tau)$$
$$= (1/\pi f)\sin(\pi f t)$$



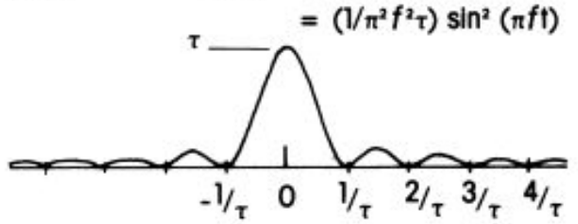
Triangle

$$G(t) = \begin{cases} 1-|t|/\tau, & |t| < \tau \\ 0, & |t| > \tau \end{cases}$$



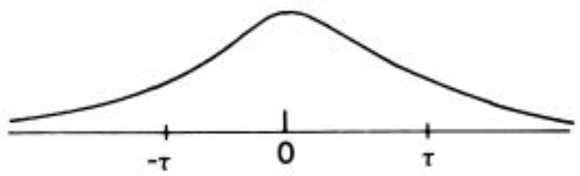
Sinc²

$$S(f) = \tau \,\mathrm{sinc}^2\,(ft)$$
$$= (1/\pi^2 f^2 \tau)\sin^2(\pi f t)$$



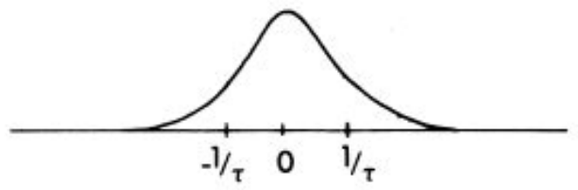
Gaussian

$$G(t) = e^{-\frac{1}{2}t^2}$$



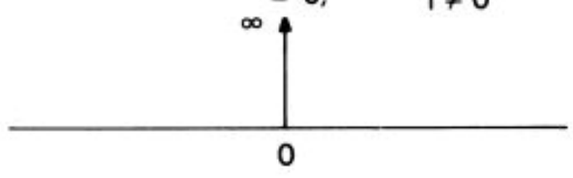
Gaussian

$$S(f) = \tau(2\pi)^{1/2}\, e^{-(\pi f \tau)^2}$$



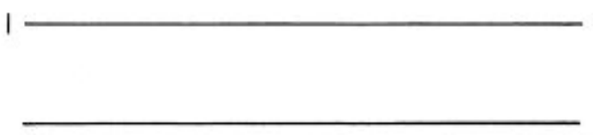
Impulse

$$G(t) = \delta(t)$$
$$= 0, \quad t \neq 0$$



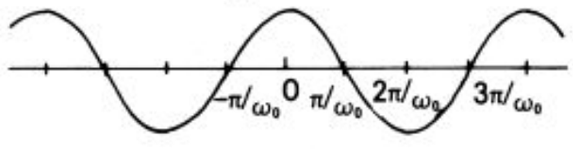
DC Shift

$$S(f) = 1$$



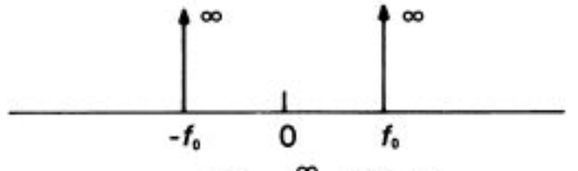
Sinusoid

$$G(t) = \cos \omega_0 t$$



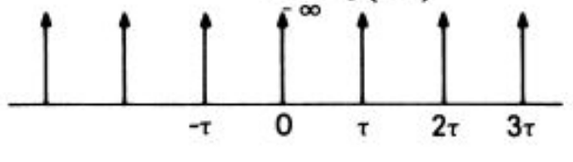
Single Freq.

$$S(f) = \tfrac{1}{2}(\delta(f+f_0) + \delta(f-f_0))$$



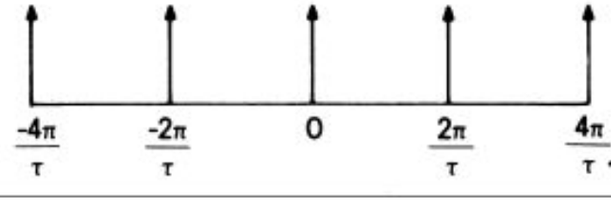
Comb.

$$G(t) = \mathrm{comb}\,(t)$$
$$= \sum_{-\infty}^{\infty} \delta(t-n\tau)$$



Comb.

$$S(f) = \sum_{-\infty}^{\infty} \delta(f-n/\tau)$$



END