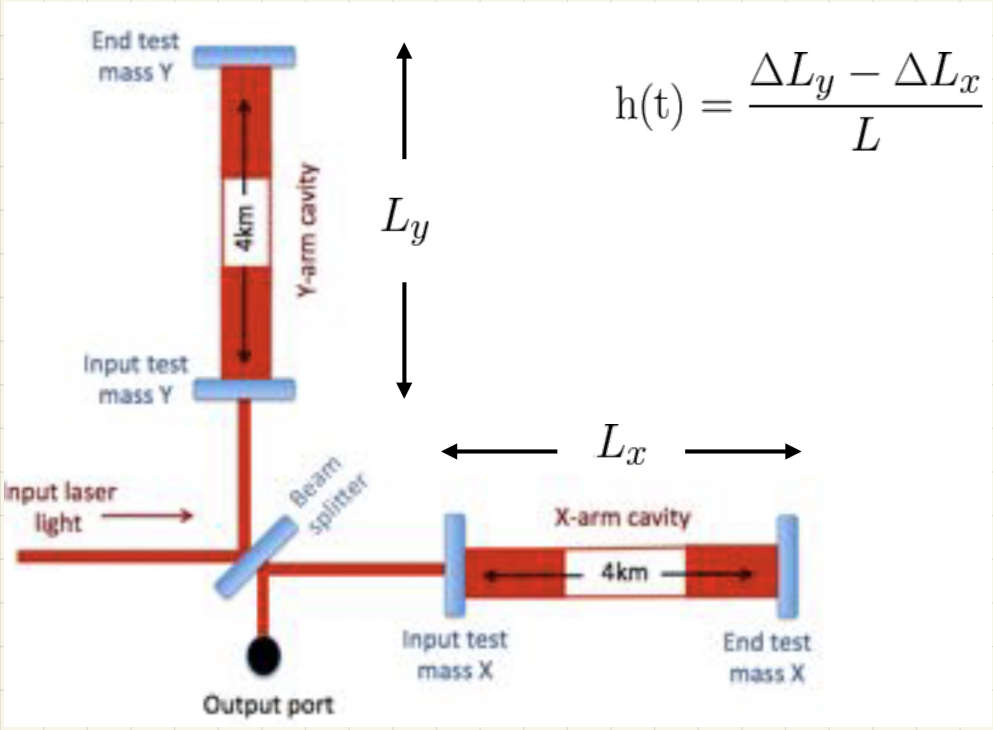
*An introduction to signal processing: the Fourier domain

* Gravitational waves: ripples in the fabric of space-time generated by the acceleration of matter.

$$h_{ij}(t) \propto \frac{G}{c^4 r} \frac{d^2 I_{ij}}{dt^2}$$

* For GW propagation, we can measure spacetime strain h(t) as $\Delta L/L$, such that:



*What's in a LIGO data file?

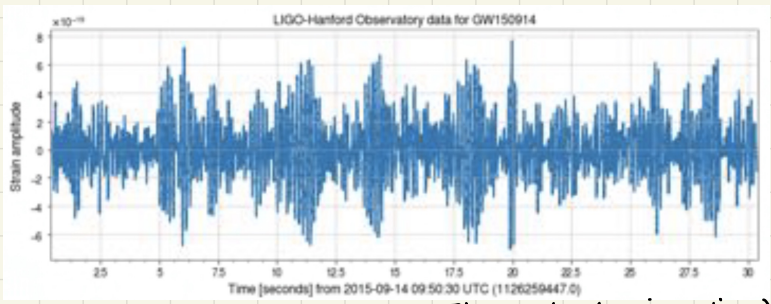
- **meta:** Meta-data for the file. Contains basic info. such as the GPS times covered, which instruments, etc.
- **strain:** strain data for the interferometer. It contains the main measurement of spacetime strain recorded by LIGO detectors.
- **quality:** A 1 Hz time series describing the data quality for each second of data.

LIGO's sampling rate: f_s for h(t) is 16384 Hz.

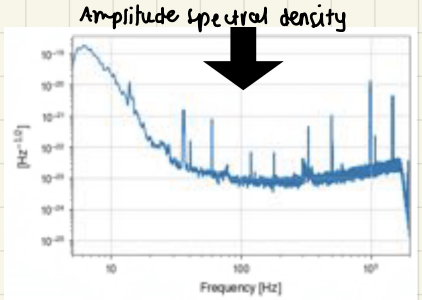
*Nyquist frequency = $f_s/2$: Data can only represent frequency content below the Nyquist frequency. Higher frequency signals will be lost or "aliased" to lower frequencies.

*GWpy: Provides OOP methods to access GW detector data, process and visualize them.

*The idea is to convert the time domain to the frequency domain:



Time series (strain vs time)



$(Hz^{-1/2}$ vs frequency).

• We use the Fourier series for that matter:

→ Fourier series. Any function can be represented as a sum of sines and cosines (w/ some coefficients that can also be functions).

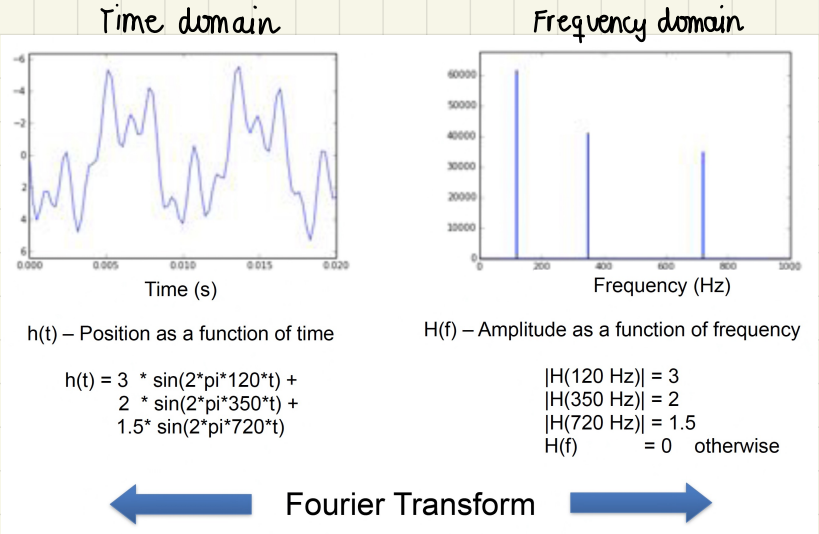
$$f(x) = \sum_{n=0}^{\infty} A_n \cos(n\pi x/L) + \sum_{n=1}^{\infty} B_n \sin(n\pi x/L)$$

→ Fourier transform: When we transform our function or time (or space) into the "frequency domain", we are projecting $f(x)$ onto an orthogonal basis of sines and cosines

$$\tilde{x}(f) = \int_{-\infty}^{\infty} x(t) e^{-2\pi i f t} dt \qquad \text{(Transform).}$$

$$x(t) = \int_{-\infty}^{\infty} \tilde{x}(f) e^{2\pi i f t} df \qquad \text{(Inverse transform).}$$

*We basically decompose the function into its component frequencies.



*Power Spectral Density:

→ Parseval's theorem:

$$\int_{-\infty}^{\infty} dt\, |x(t)|^2 = \int_{-\infty}^{\infty} df\, |\tilde{x}(f)|^2$$

} Total energy in the data can be calculated in either time domain or frequency domain.

$[|\tilde{x}(f)|^2]$ Energy spectral density
(normalize by $1/T$ to get power.
Signal energy per unit frequency (per Hz).

$[\tilde{x}(f)]$ Amplitude spectral density
(sqrt of power for each discrete frequency).
Signal amplitude per unit frequency (per $Hz^{1/2}$).

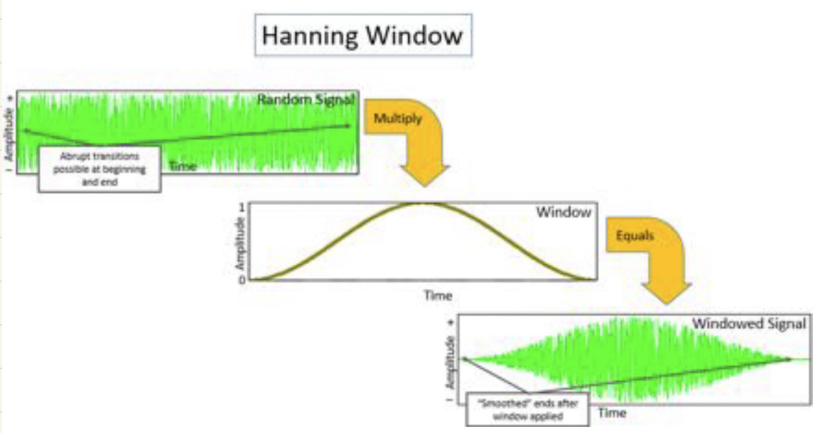
* How to estimate the PSD:

*Step 0: Take a Fast Fourier Transform (FFT), which is any algorithm useful for quickly estimating the Discrete Fourier Transform that describes a discrete time series.

$$X_K = \sum_{n=0}^{N-1} x_n e^{\frac{-2\pi i}{N} Kn} = \sum_{n=0}^{N-1} x_n \cdot [\cos(2\pi Kn/N) - i\sin(2\pi Kn/N)]$$

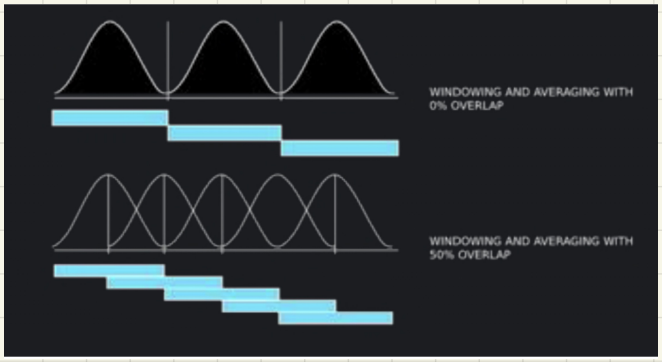
We shift to discretized data.

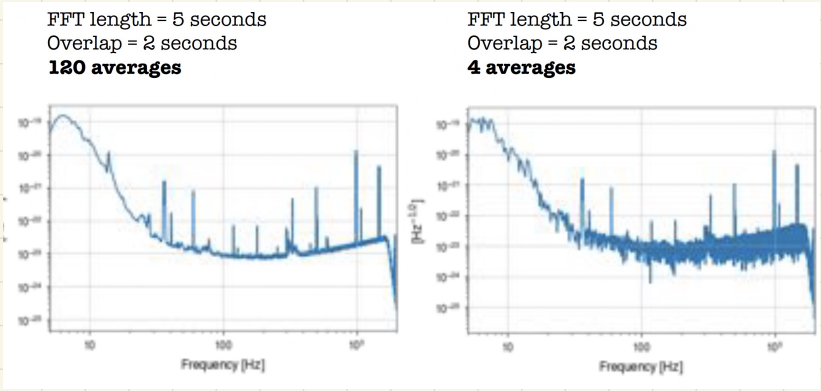
*Step 1: Apply a window to the data (if it's linear, as a time series is) to prevent spectral leakage, from the assumption the signal is periodic.



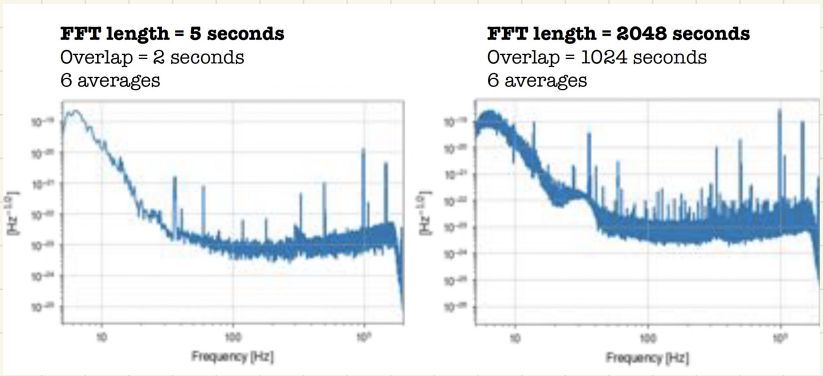
A single windowed FFT is unbiased (i.e. will give the correct mean PSD), but has high variance. In this sense, we can average several FFTs.

*Step 2: Divide data into shorter time segments, take a windowed FFT of each, and average these together. Some frequency resolution may be lost this way, though. Welch's method averages the mean value for each frequency bin across FFTs, with some overlap in the data analyzed.



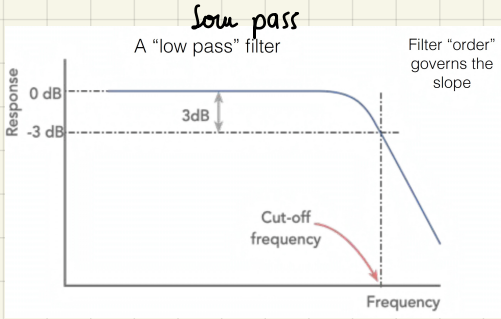


*we can do this using Scipy, which provides FFT wrappers to estimate frequency-domain content of data.

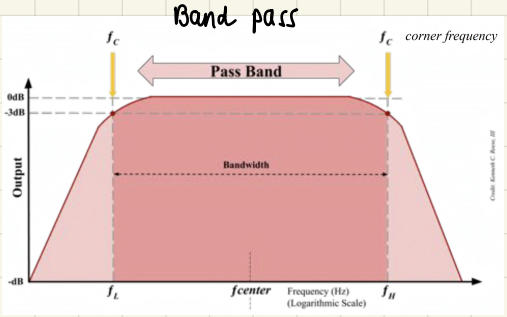


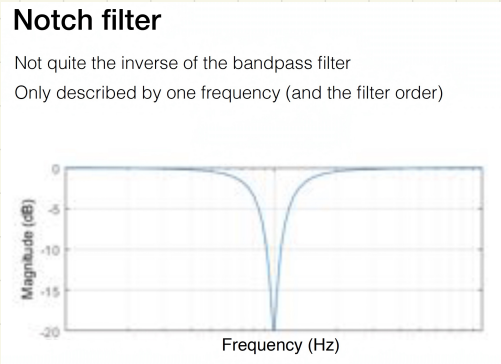
*Signal Processing: GWpy provides lowpass, highpass, bandpass, notch, and whitening filters.

low pass



Band pass





*The Q transform:

- Relative time-frequency aspect ratios for sine-Gaussian basis tiles with the same frequency and varying Q, w/ $Q = f_0/\Delta f$