Discrete Fourier Transform

Valerio Velardo

Join the community!

thesoundofai.slack.com

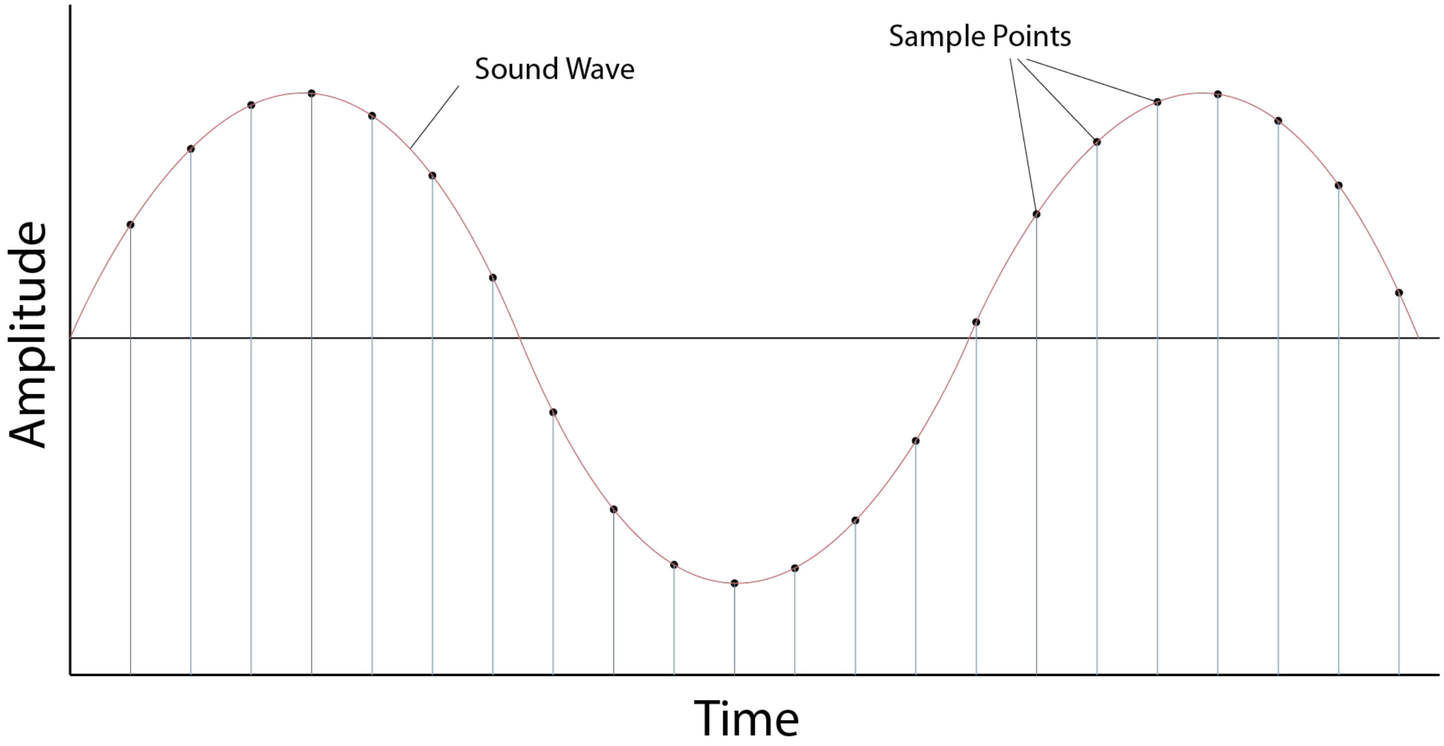
Previously...

$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt$$

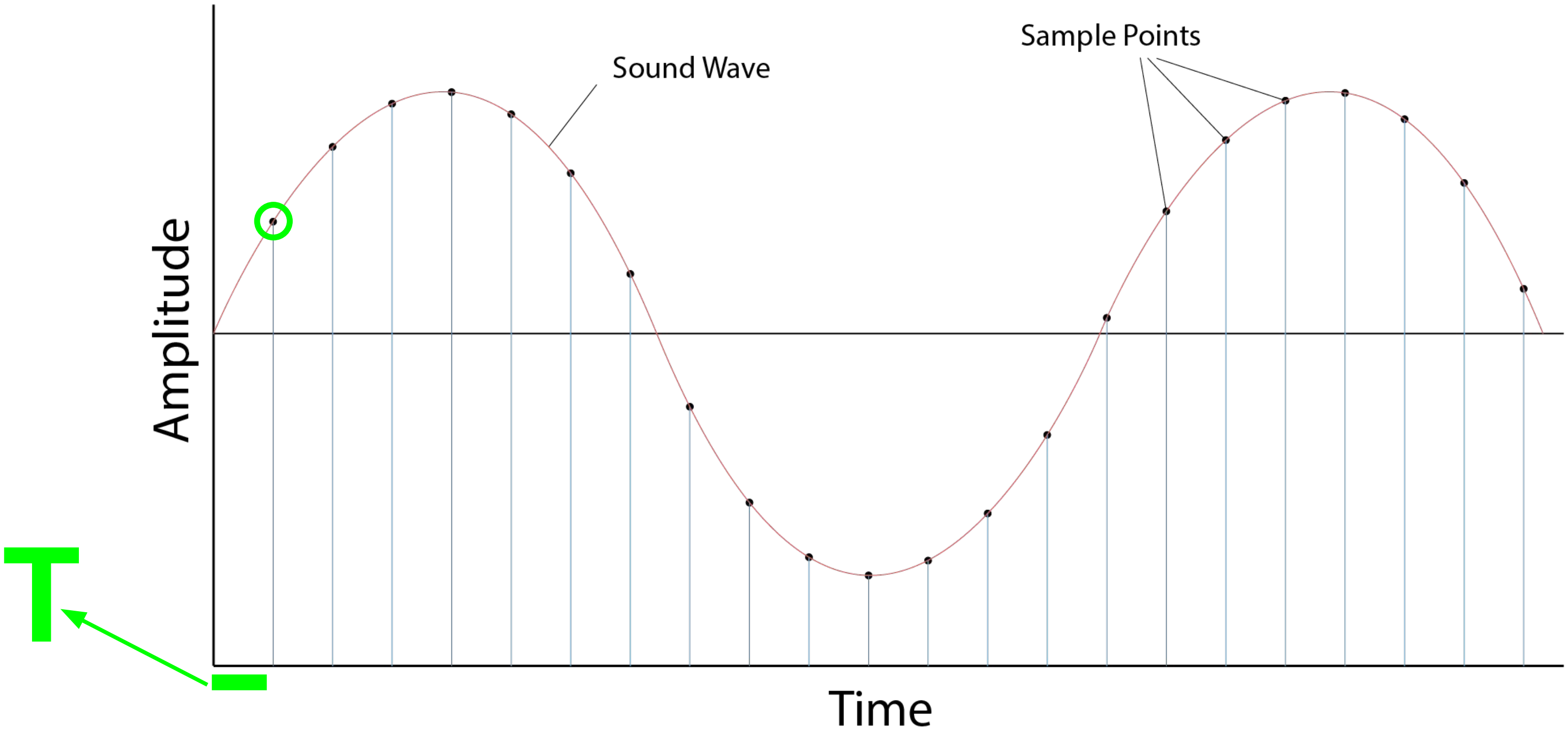
Previously...

$$\hat{g}(f) = \int \boxed{g(t)} \cdot e^{-i2\pi ft} dt$$

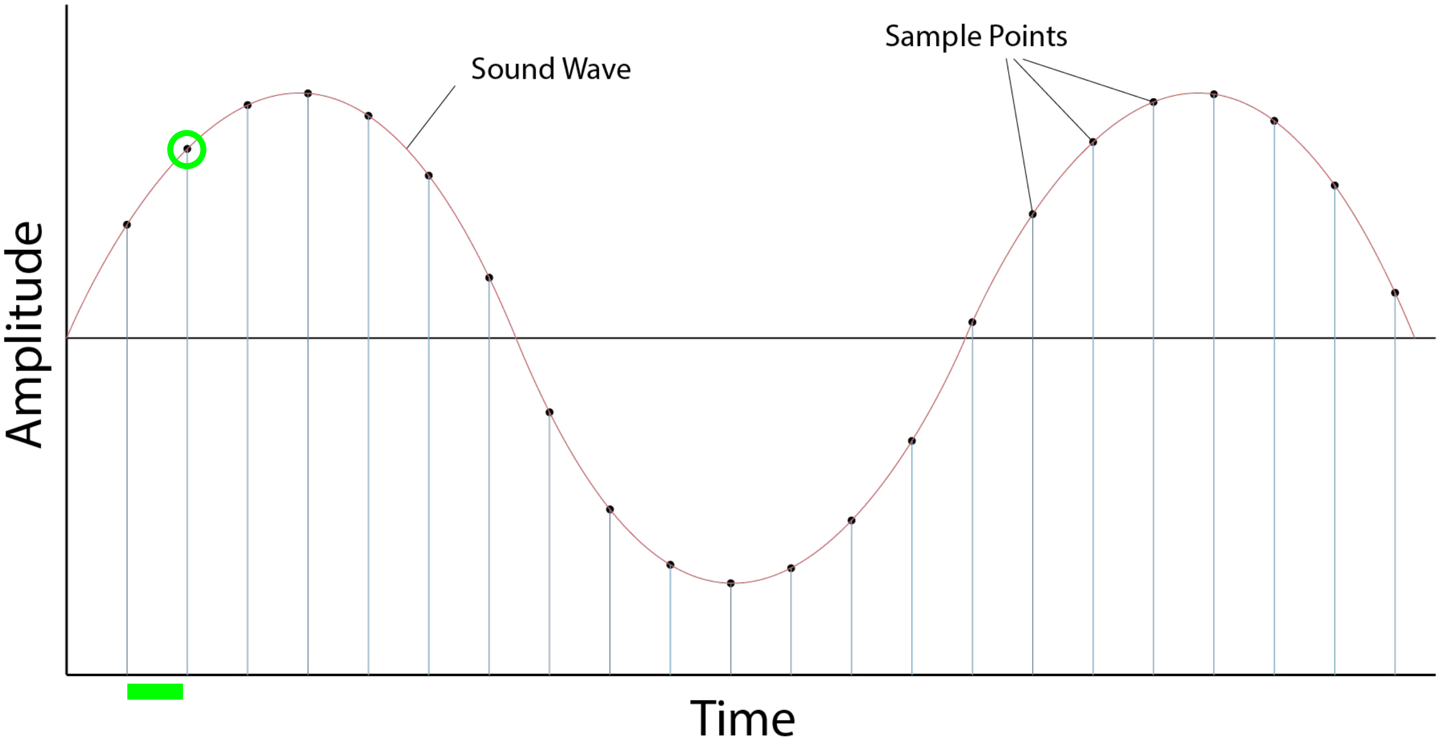
Digitalization



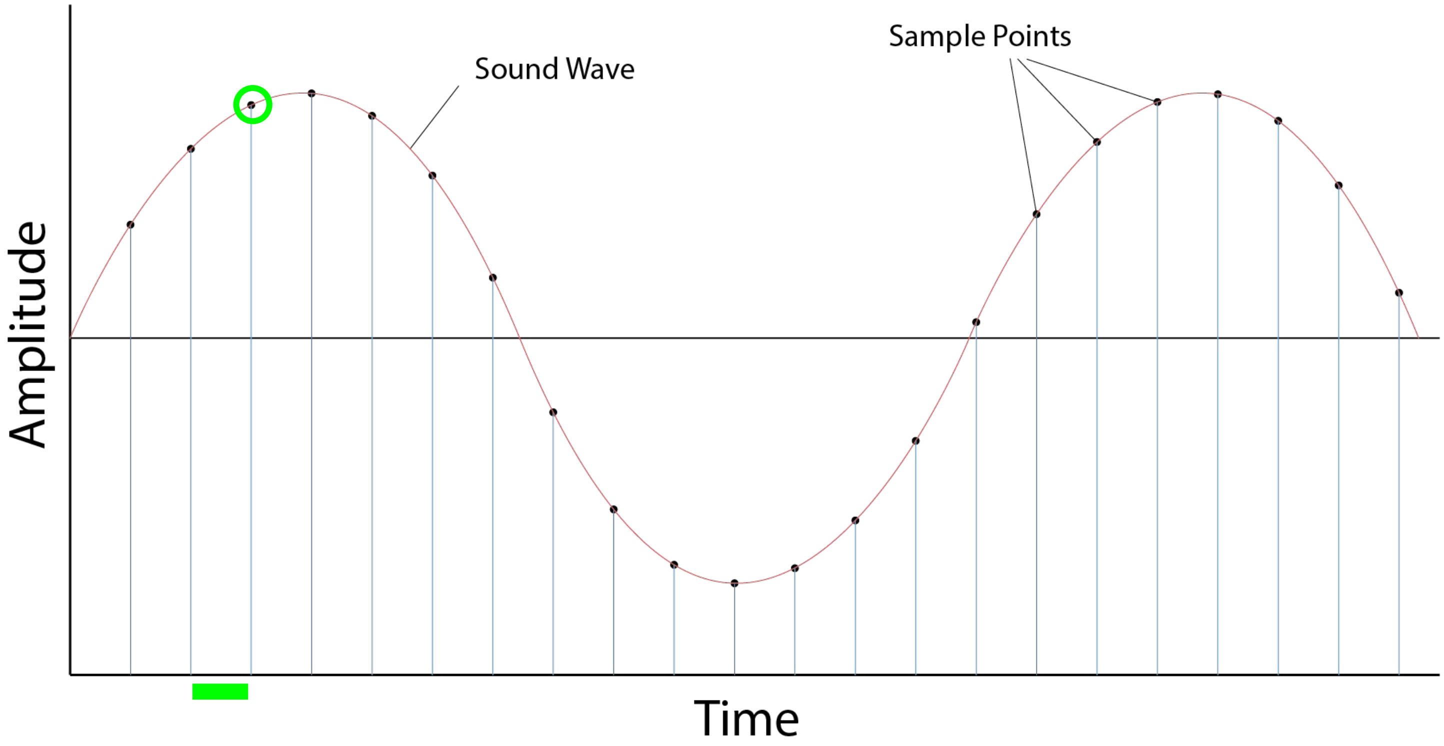
Digitalization



Digitalization



Digitalization



Digital signal

$$g(t) \mapsto x(n)$$

Digital signal

$$g(t) \mapsto x(n)$$

$$t = nT$$

Building a discrete Fourier transform

$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi f t} dt$$

Building a discrete Fourier transform

$$\boxed{\hat{g}(f)} = \int g(t) \cdot e^{-i2\pi f t} dt$$

$$\boxed{\hat{x}(f)}$$

Building a discrete Fourier transform

$$\hat{g}(f) = \boxed{\int} g(t) \cdot e^{-i2\pi f t} \boxed{dt}$$

$$\hat{x}(f) = \boxed{\sum_{n}}$$

Building a discrete Fourier transform

$$\hat{g}(f) = \int \boxed{g(t)} \cdot e^{-i2\pi ft} dt$$

$$\hat{x}(f) = \sum_{n} \boxed{x(n)}$$

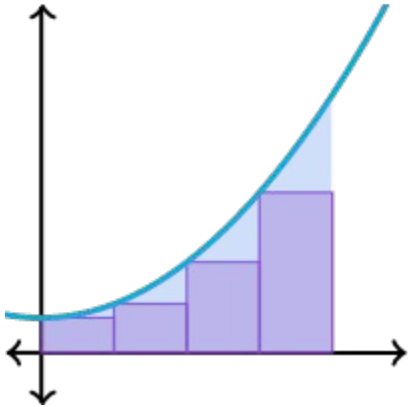
Building a discrete Fourier transform

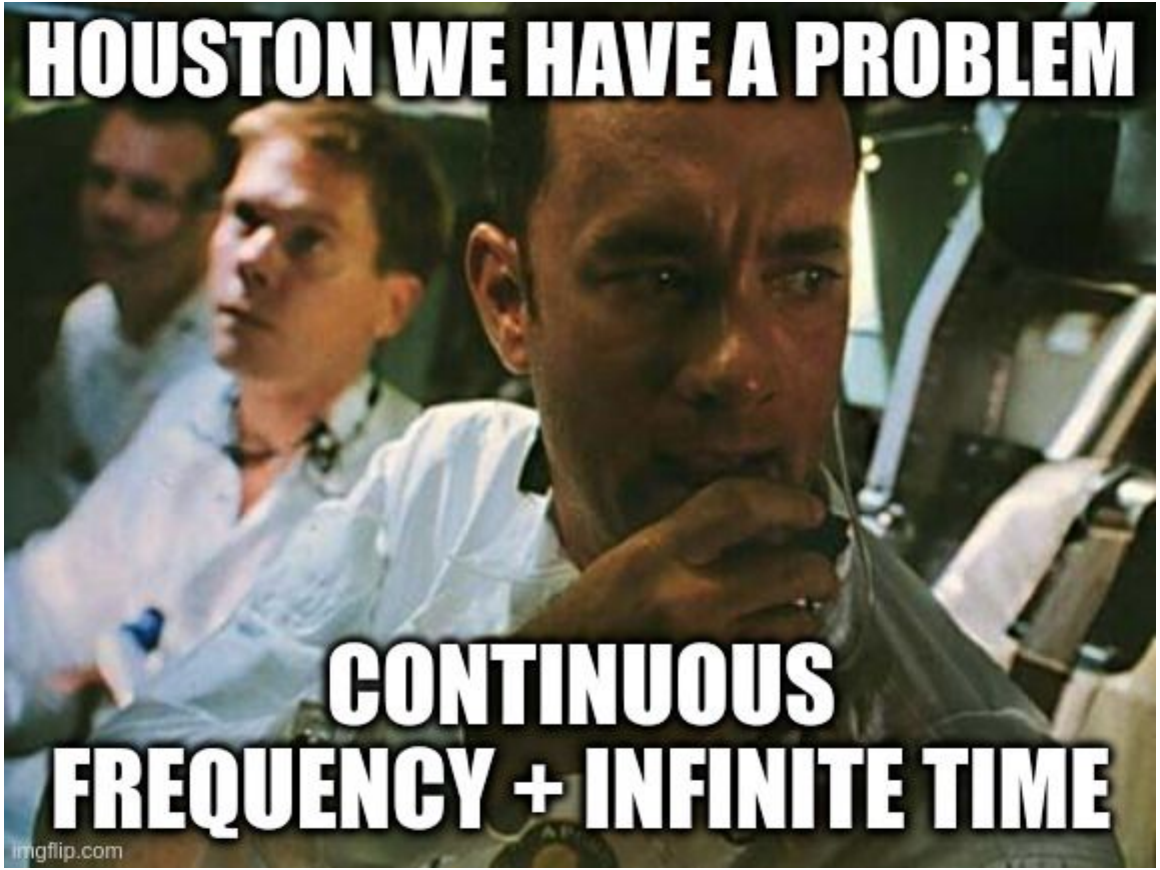
$$\hat{g}(f) = \int g(t) \cdot \boxed{e^{-i2\pi f t}} dt$$

$$\hat{x}(f) = \sum_{n} x(n) \cdot \boxed{e^{-i2\pi f n}}$$

DFT: Visual interpretation

$$\hat{x}(f) = \sum_{n} x(n) \cdot e^{-i2\pi f n}$$




HOUSTON WE HAVE A PROBLEM
CONTINUOUS
FREQUENCY + INFINITE TIME
imgflip.com

Building a discrete Fourier transform

$$\hat{x}(f) = \sum_{n} x(n) \cdot e^{-i2\pi fn}$$

Hack 1: Time

- Consider *f* to be non 0 in a finite time interval
- $x(0), x(1), \ldots, x(N-1)$

Hack 2: Frequency

- Compute transform for finite # of frequencies

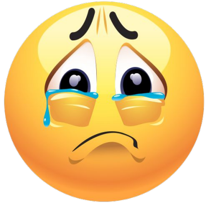
Hack 2: Frequency

- Compute transform for finite # of frequencies
- # frequencies (M) = # samples (N)

Hack 2: Frequency

- Compute transform for finite # of frequencies
- # frequencies (M) = # samples (N)
- Why M = N?
    - Invertible transformation
    - Computational efficient

Hacking our way around...



$$\hat{x}(f) = \sum_n x(n) \cdot e^{-i2\pi f n}$$

Hacking our way around...

$$\hat{x}(f) = \sum_{n} x(n) \cdot e^{-i2\pi f n}$$

Hacking our way around...

$$\hat{x}(f) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi f n}$$

Hacking our way around...

$$\hat{x}(f) = \boxed{\sum_{n=0}^{N-1}} x(n) \cdot e^{-i2\pi f n}$$

Hacking our way around...

$$\hat{x}(f) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi f n}$$

Hacking our way around...

$$\hat{x}(k/N) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi n \frac{k}{N}}$$

Hacking our way around...

$$\hat{x}(\boxed{k/N}) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi n \boxed{\frac{k}{N}}}$$

Hacking our way around...

$$\boxed{k = [0, M-1] = [0, N-1]}$$

$$\hat{x}(k/N) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi n \frac{k}{N}}$$

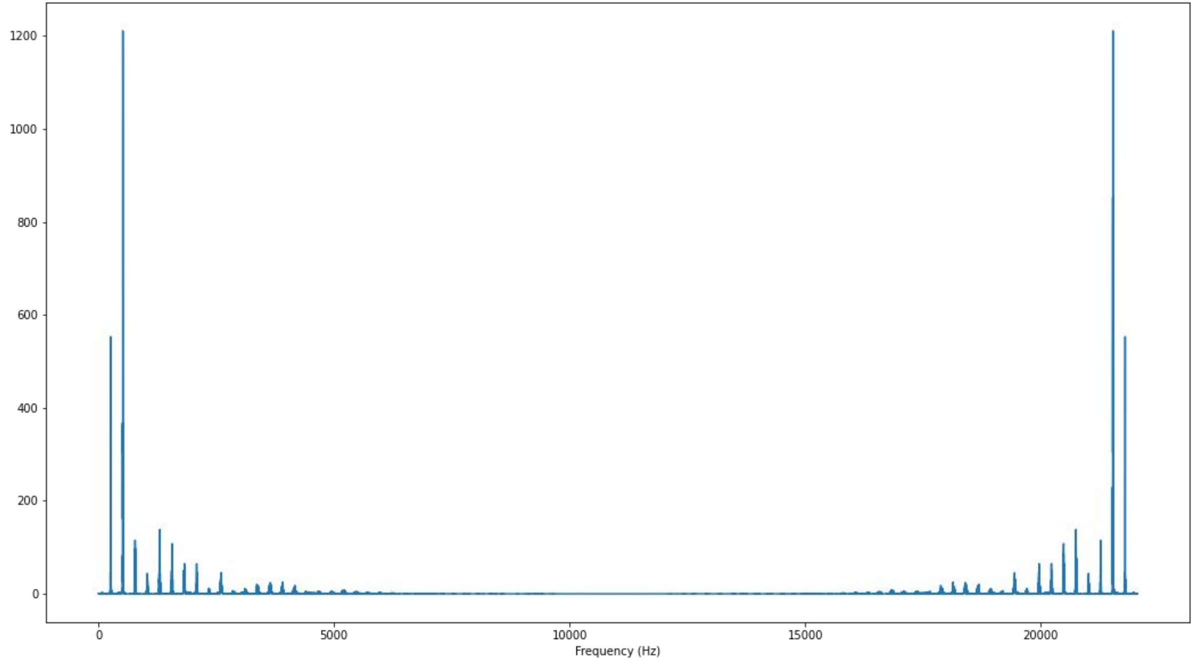
Hacking our way around...

$$F(k) = \frac{k}{NT}$$

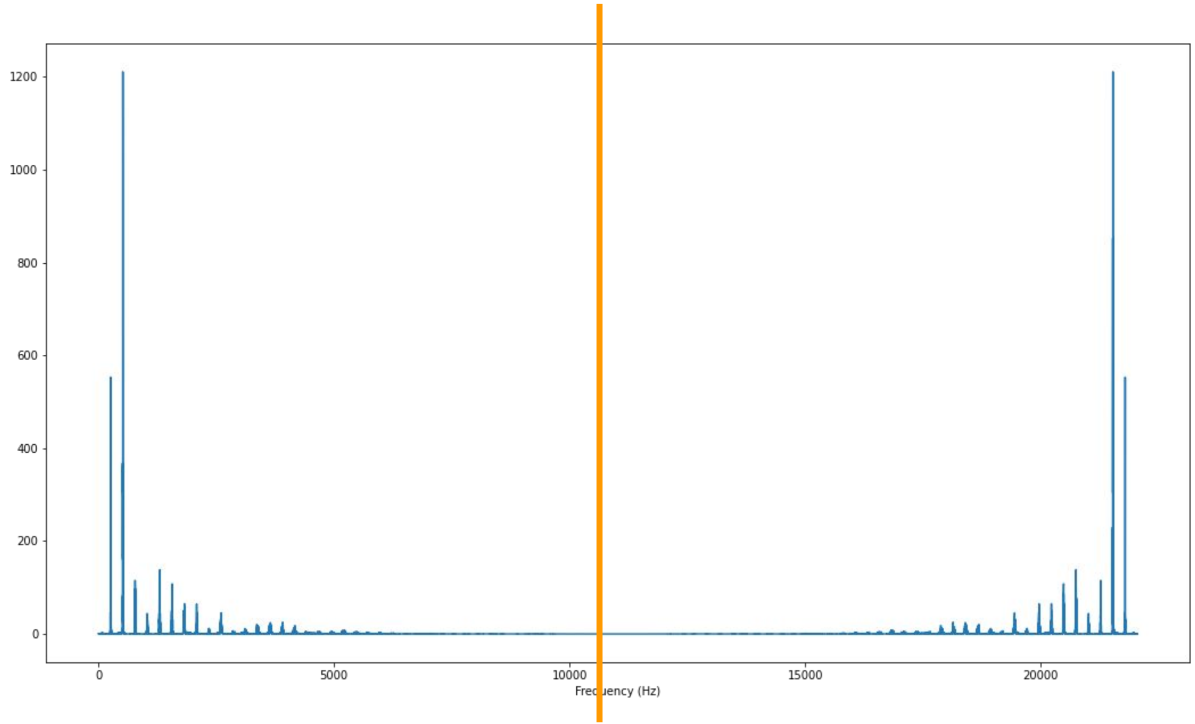
Hacking our way around...

$$F(k) = \frac{k}{NT} = \frac{k s_r}{N}$$

Redundancy in DFT

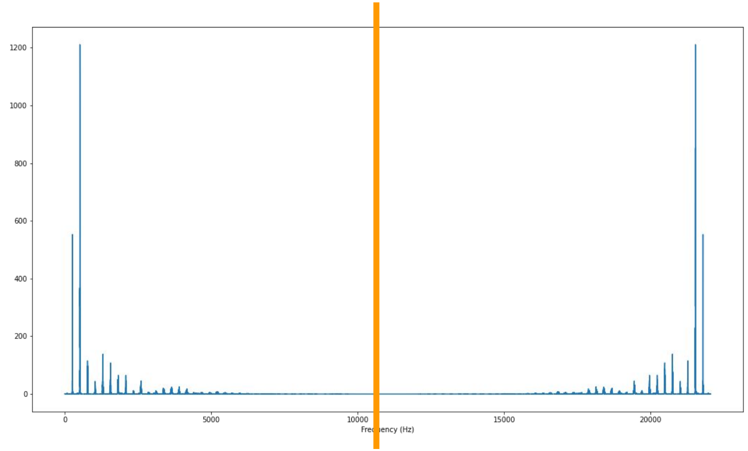


Redundancy in DFT



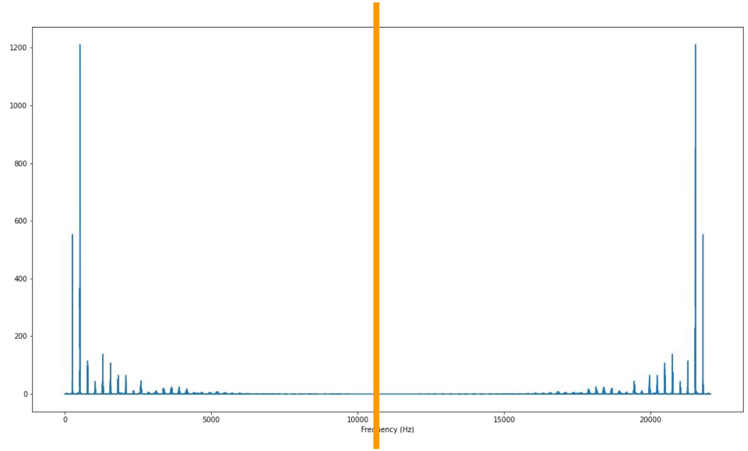
Redundancy in DFT

$$\boxed{k = N/2}$$



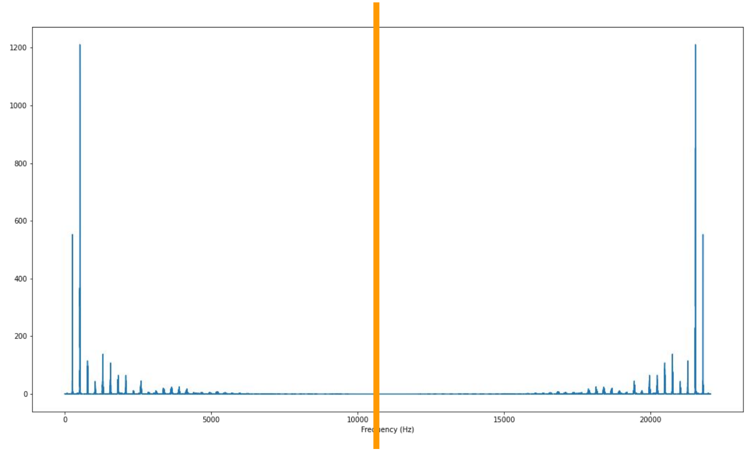
Redundancy in DFT

$$k = N/2 \rightarrow \boxed{F(N/2) = s_r/2}$$



Redundancy in DFT

$$k = N/2 \rightarrow \boxed{F(N/2) = s_r/2}$$



Nyquist Frequency

From DFT to Fast Fourier Transform

- DFT is computationally expensive (N^2)
- FFT is more efficient ($Nlog_2N$)
- FFT exploits redundancies across sinusoids
- FFT works when N is a power of 2

What’s up next?

- Play around with FFT