LING 490 - SPECIAL TOPICS IN LINGUISTICS

# Fundamentals of Digital Signal Processing

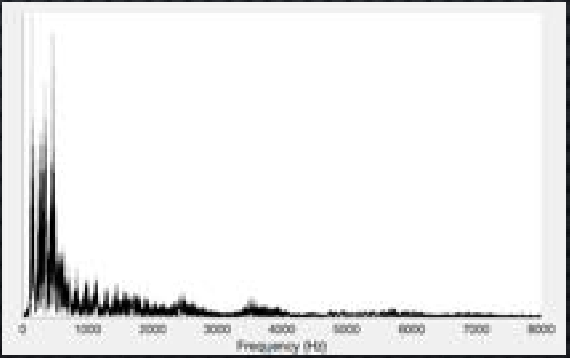
*Yan Tang*

Department of Linguistics, UIUC

Week 9-10

# Last week…

- From the time domain to the frequency domain
  - Frequency components with properties of a signal
- The Fourier Transform: Joseph Fourier 1822
  - A complex signal can be represented as a sum of a series of simple sinusoids with different frequencies and phases
  - Discrete Fourier Transform:

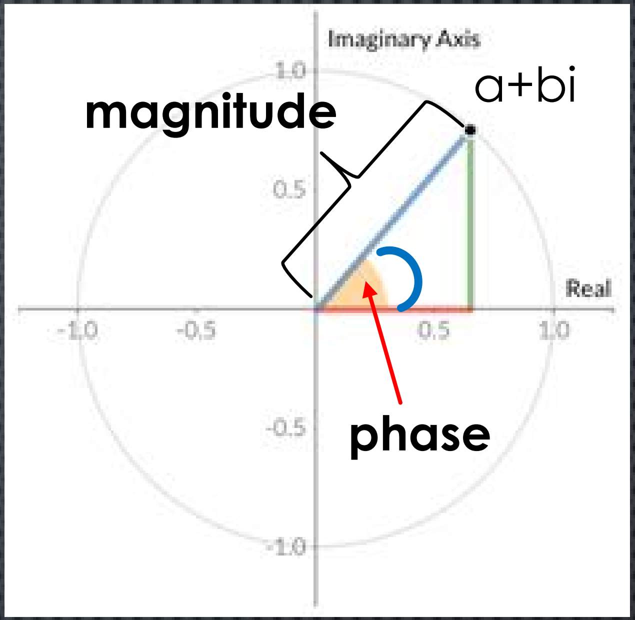


$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i\varphi}, \varphi = \frac{2\pi kn}{N}$$

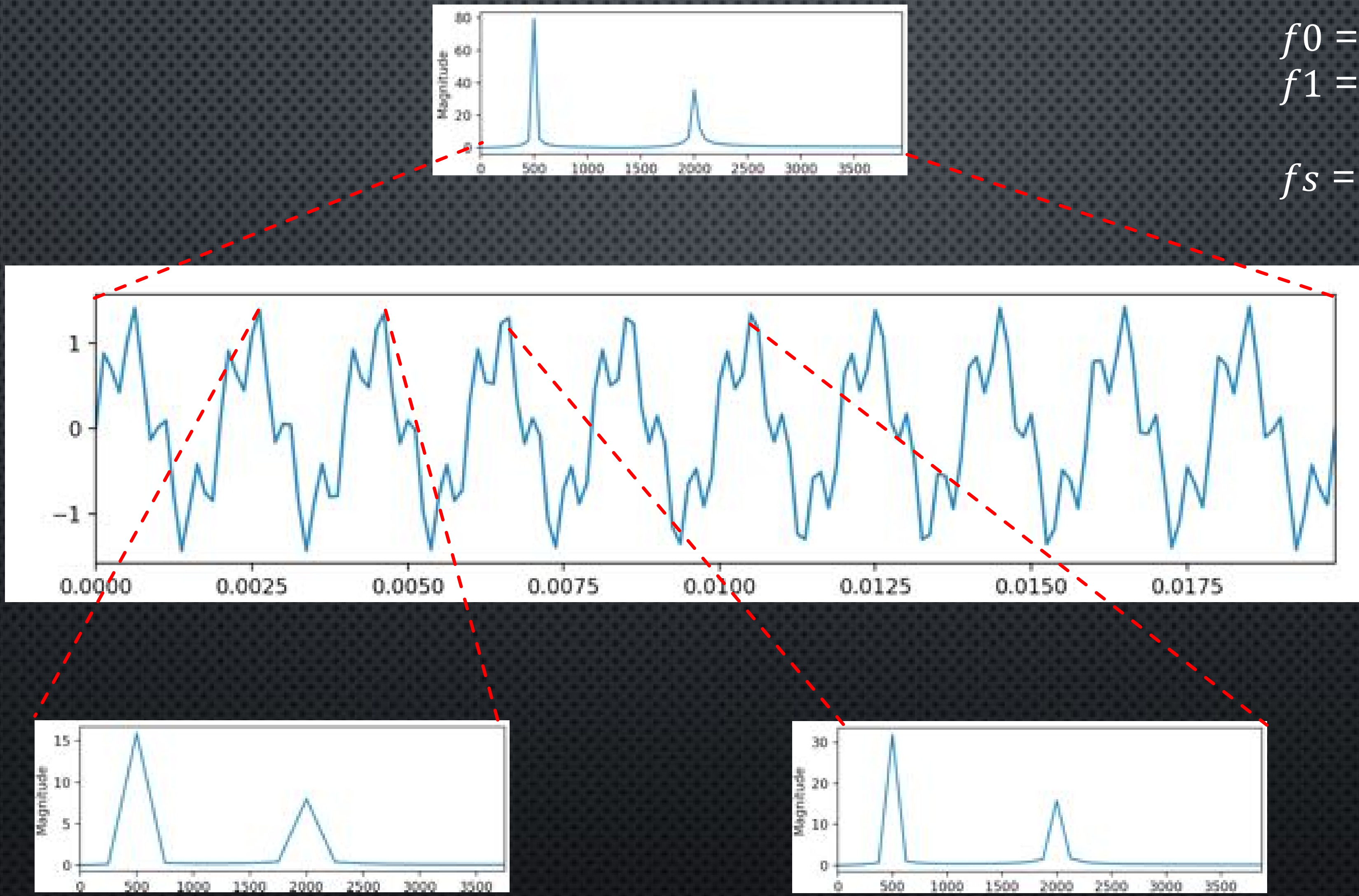
$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot (\cos(\varphi) - \sin(\varphi)i)$$

# Last week…

- Complex number
  - The real part and imaginary part
  - $magnitude = \sqrt{a^2 + b^2}$
  - $phase = \tan^{-1}\frac{b}{a}$




- Fast Fourier Transform (FFT)
  - An algorithm for the efficient calculation of the DFT (not another transform)

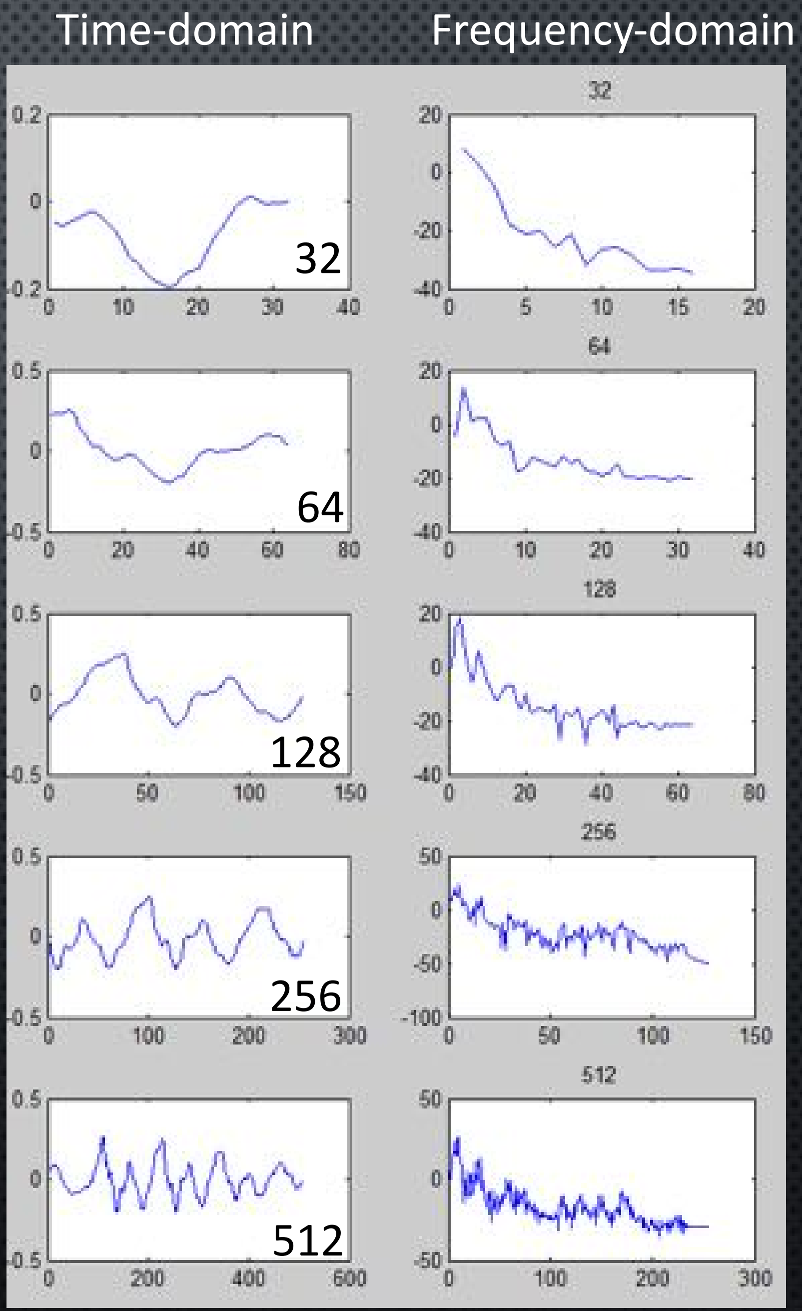



$f0$ = 500
$f1$ = 2000

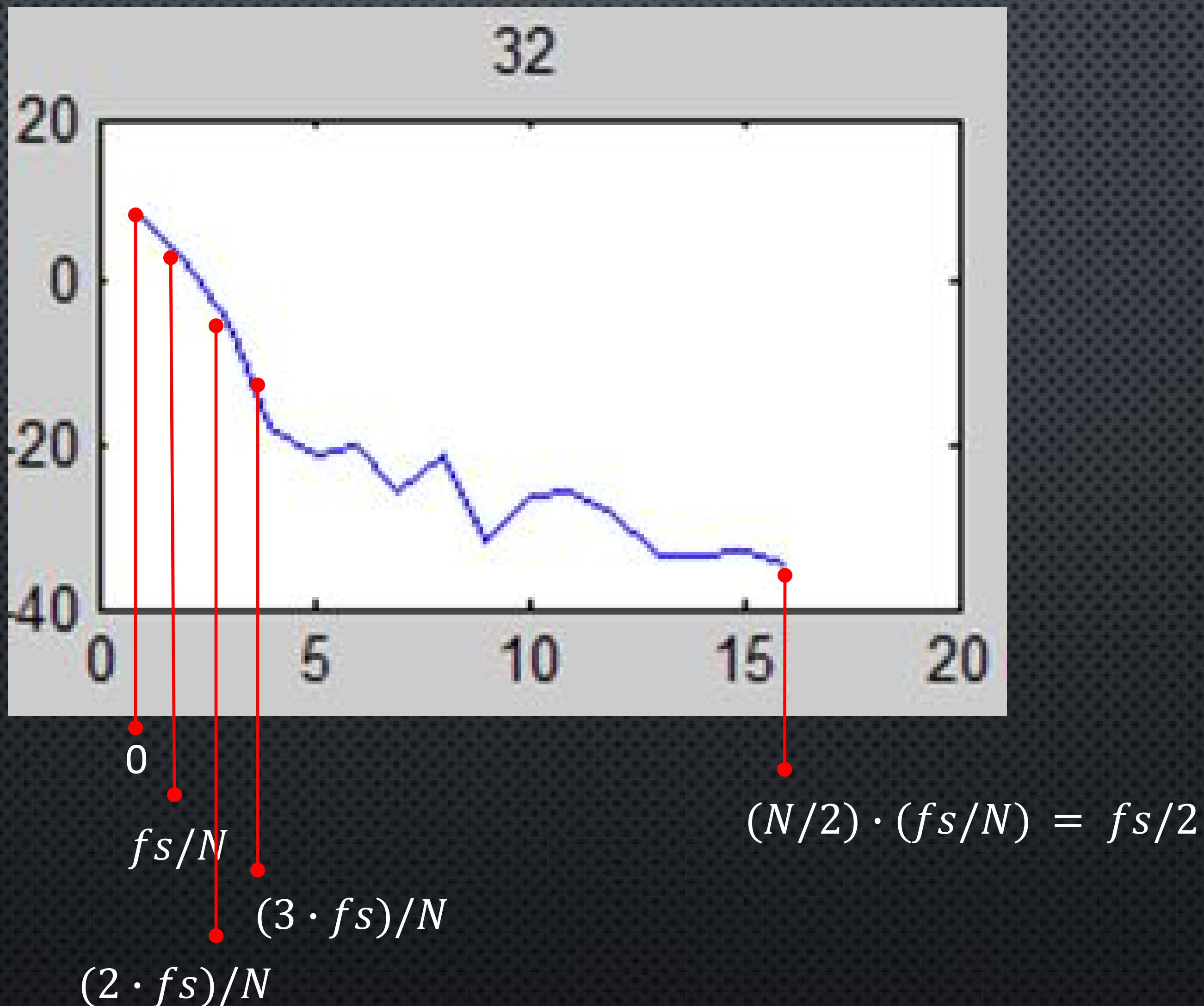
$fs$ = 4000

# Time/frequency resolution

- The Fourier transform always stretches from 0 to $fs$ Hz. $fs$: sampling frequency
- Each point on the frequency axis represents a distance of $(fs/N)$ Hz. $N$: number of samples
- The first point is 0 Hz, 2nd is fs/N Hz, 3rd point is $(2 \cdot fs)/N$, 4th is $(3 \cdot fs)/N$, etc



# When $N = 32$

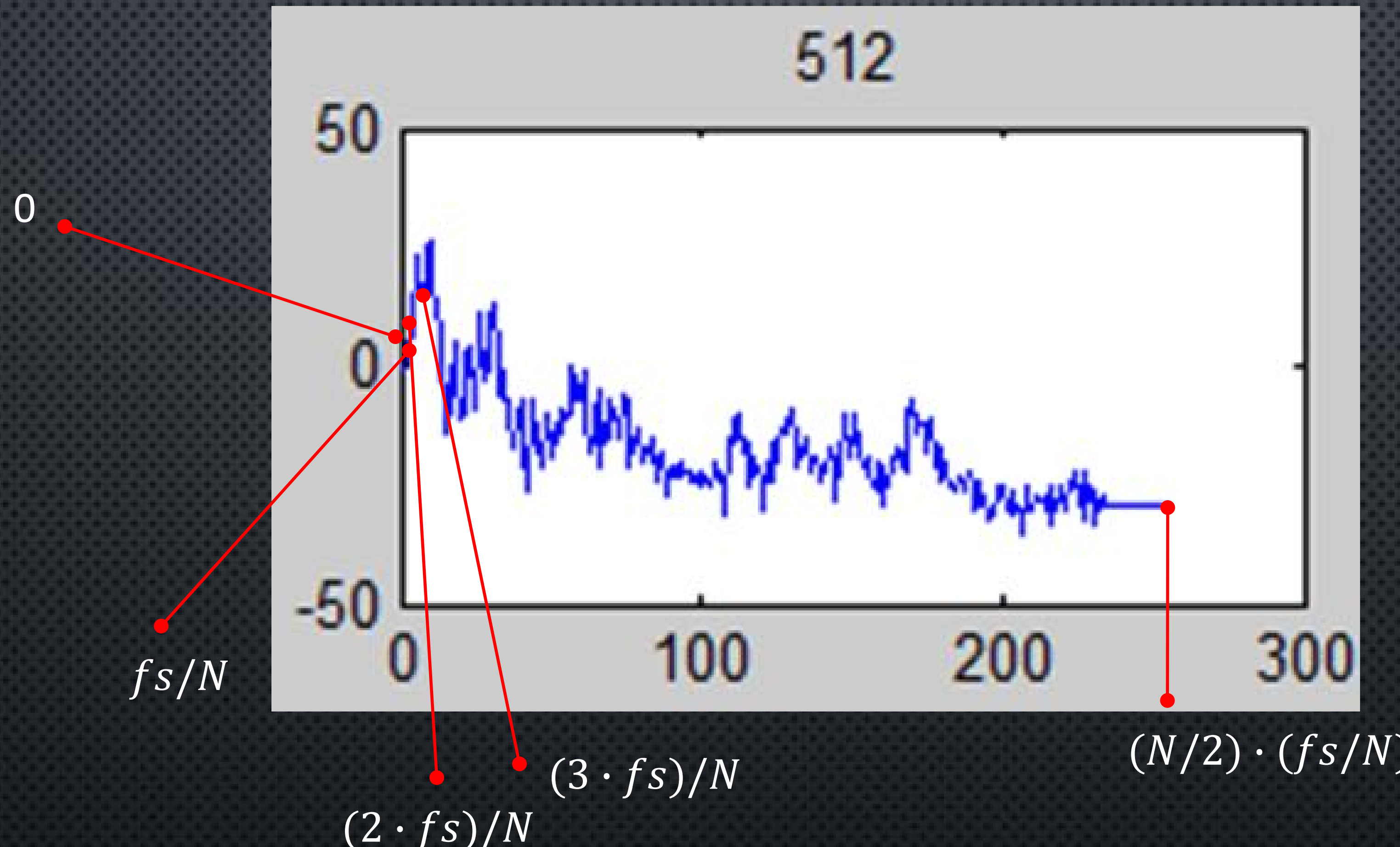



for fs=10000 Hz: 0 312.5 625 937.5 .................................... 5000 Hz



**CRUDE frequency resolution!**

# When $N = 512$




for fs=10000 Hz: 0 19.5 39 58.6 ..................................... ........... 5000 Hz



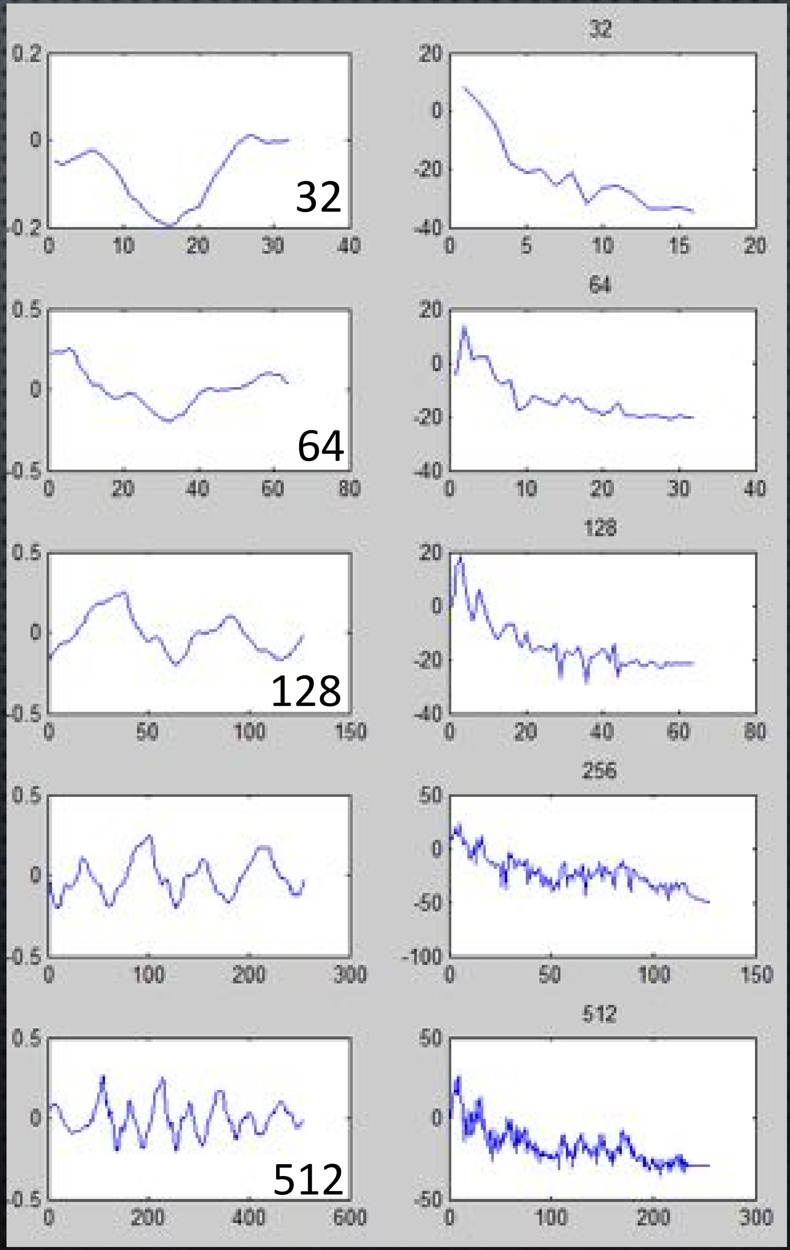
**FINE frequency resolution!**

# Time/frequency resolution

- Frequency resolution is proportional to number of signal samples, $N$
    - A short stretch of signal leads to poor frequency resolution
    - A long stretch leads to fine frequency resolution

Q: *why not always use long stretches of samples since this delivers very fine frequency resolution?*

time-domain frequency-domain

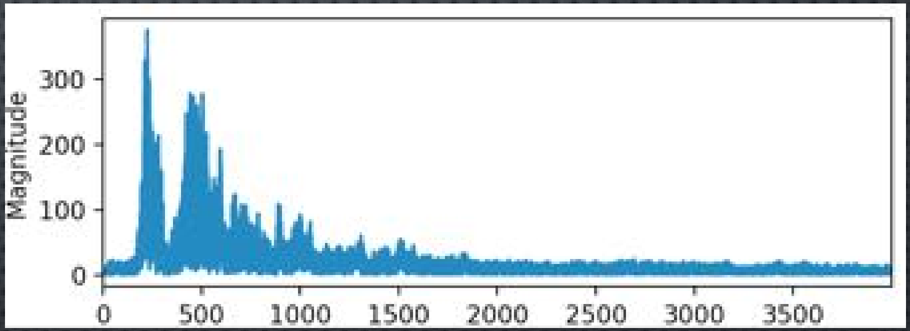


# Time-frequency resolution tradeoff

Consider the extreme case: we transform the entire signal. This results in very fine frequency resolution (here, 0.16 Hz!!)

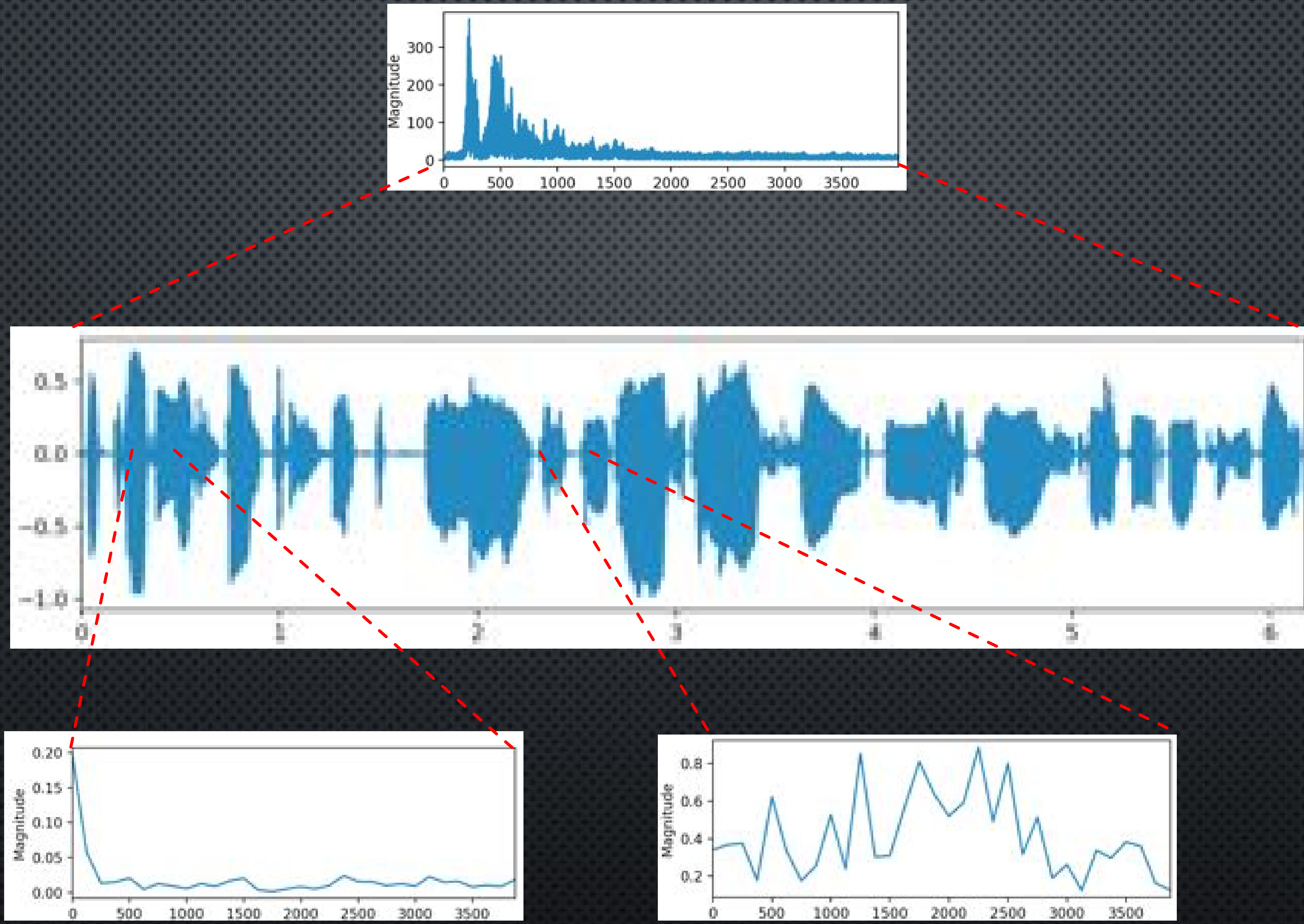
BUT: we only get ONE measurement of signal energy at each frequency. This is fine if the distribution of energy across frequency doesn't change ("Stationary signal").

Speech is certainly not like that!

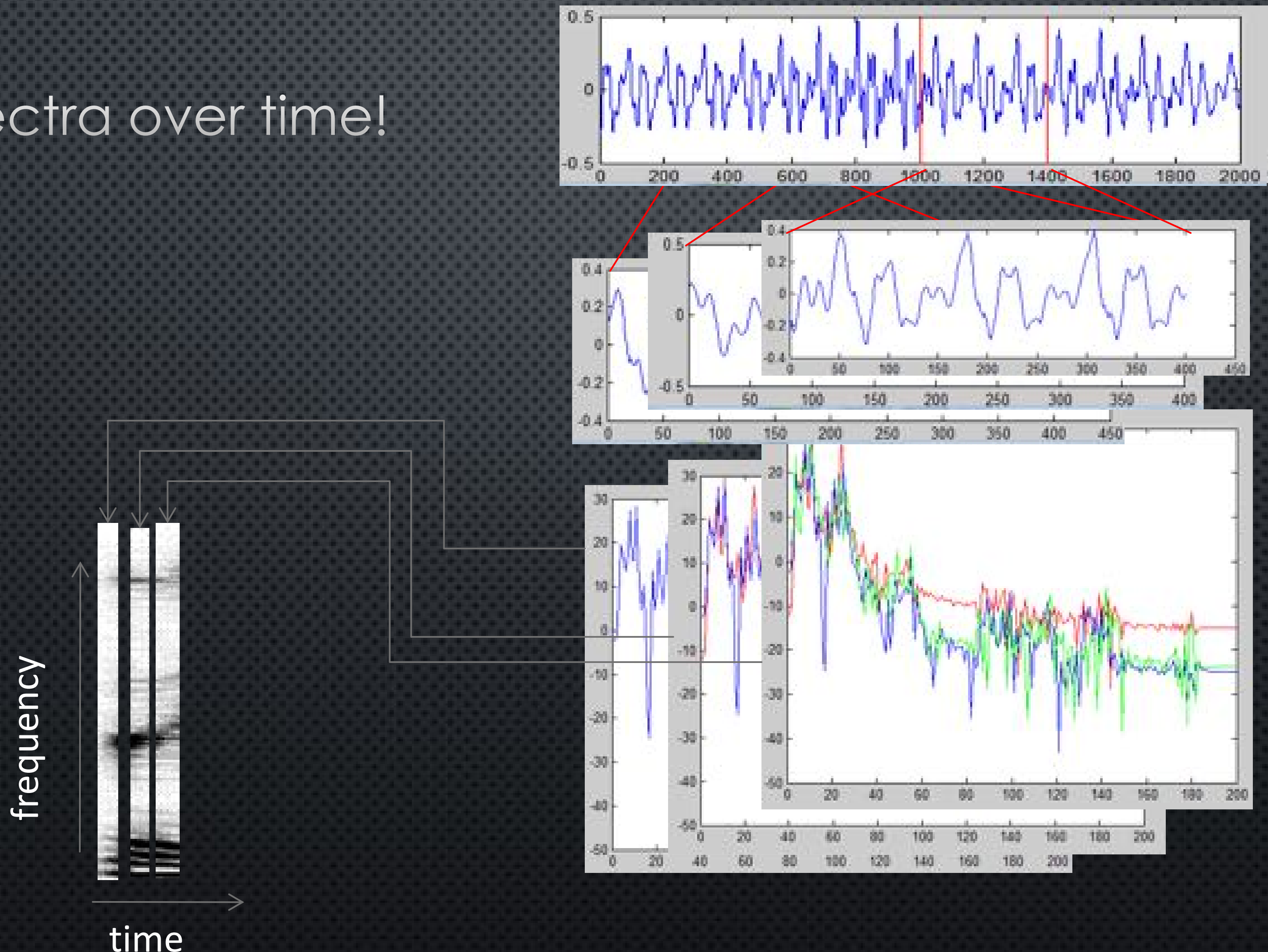

**THE TIME-FREQUENCY TRADEOFF**

Frequency resolution is proportional to the **inverse** of time resolution (and vice versa)


Magnitude
300
200
100
0
0 500 1000 1500 2000 2500 3000 3500
0.5
0.0
−0.5
−1.0
0 1 2 3 4 5 6
Magnitude
0.20
0.15
0.10
0.05
0.00
0 500 1000 1500 2000 2500 3000 3500
Magnitude
0.8
0.6
0.4
0.2
0 500 1000 1500 2000 2500 3000 3500

# Towards the spectrogram

Spectra over time!



# Windowing

- Transforming the entire signal at once
  - **Long term spectrum** of the signal

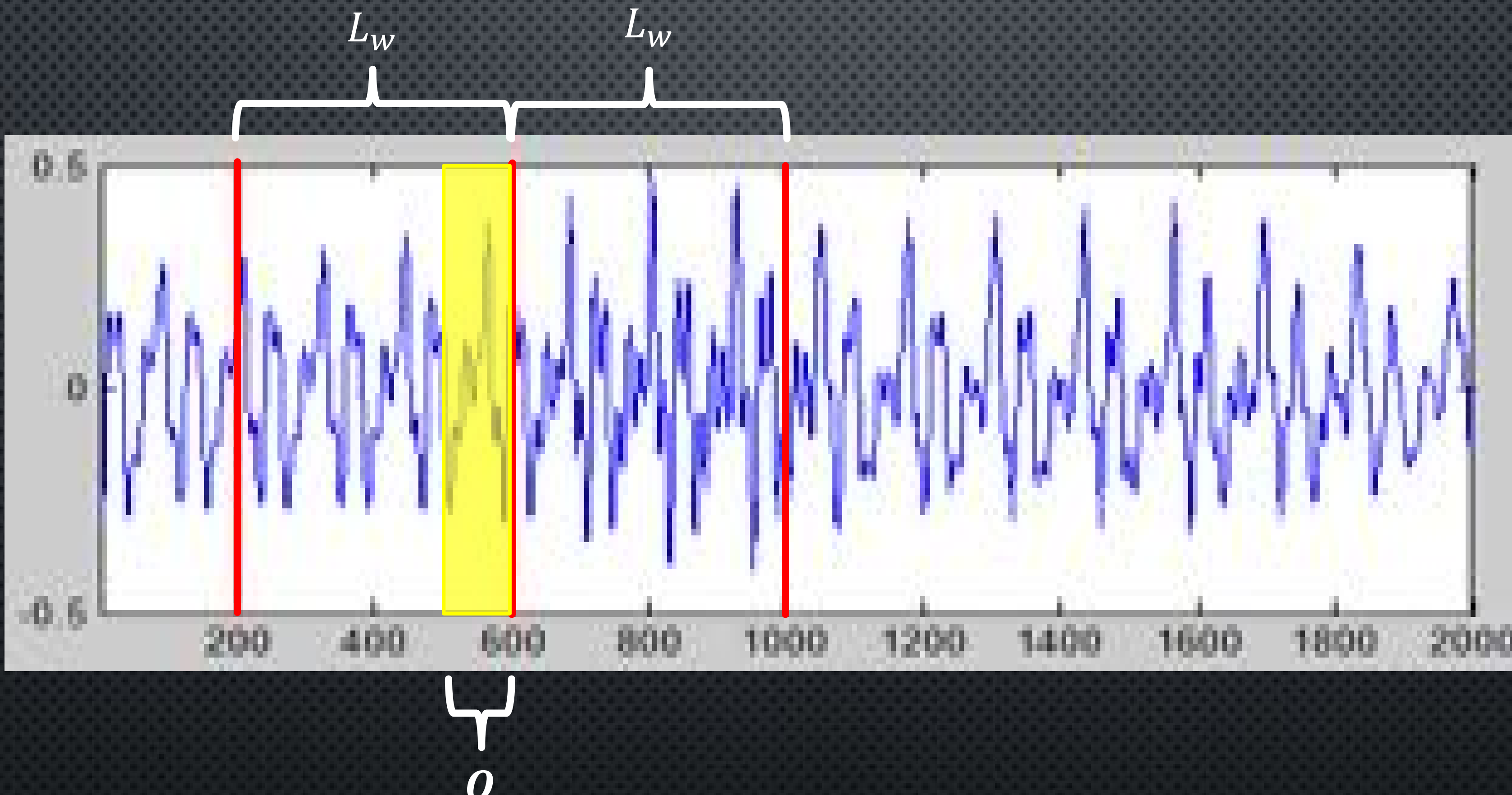
- In order to capture frequency contents changing over time
  - a sequence of **short-term spectra**

- This involves selecting successive segments of the signal, a process known as **windowing**

# Issues with windowing

- **Size of window $L_w$:**
  - Determines time-frequency trade-off
- **Overlap between successive windows $O$:**
  - Improves quality of spectrogram display
  - Compensates the information loss due to the application of window function
- **Choice of window function**
  - Reduce artefacts
  - Increase frequency selectivity

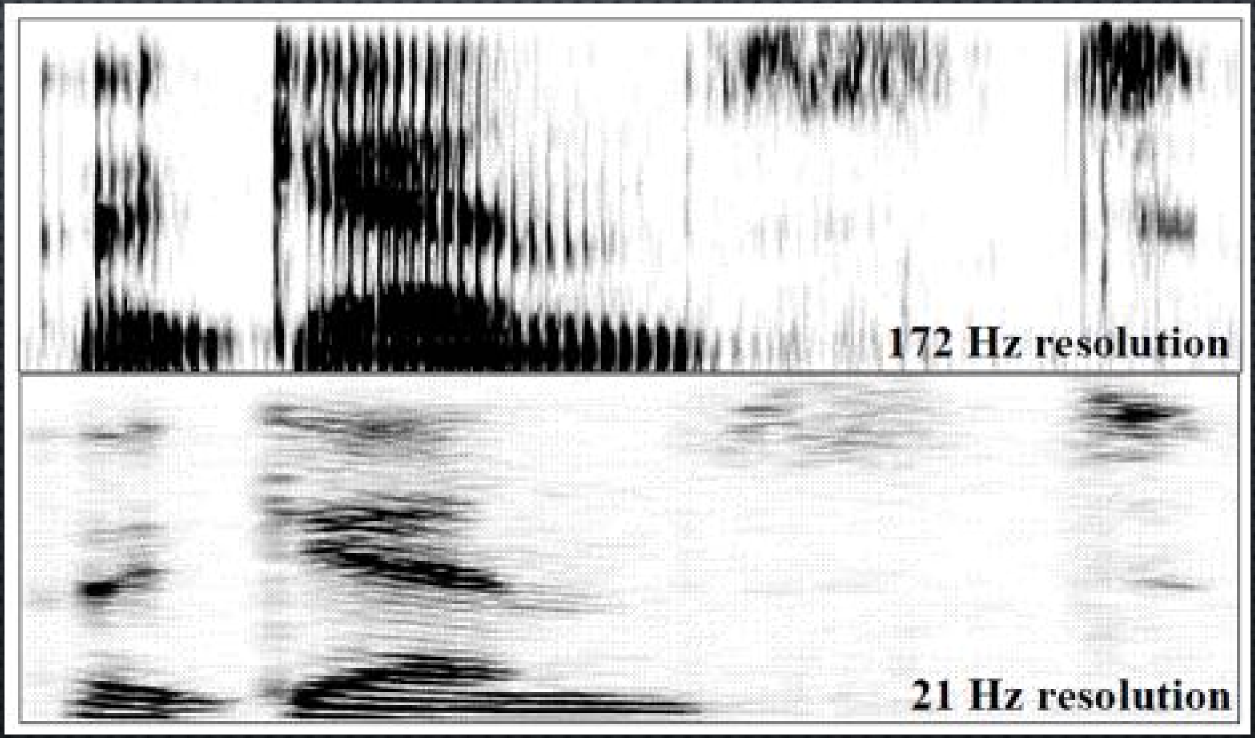
# Issues with windowing



# Window size: wideband vs narrowband spectrograms

Wideband (upper) and narrowband (lower)

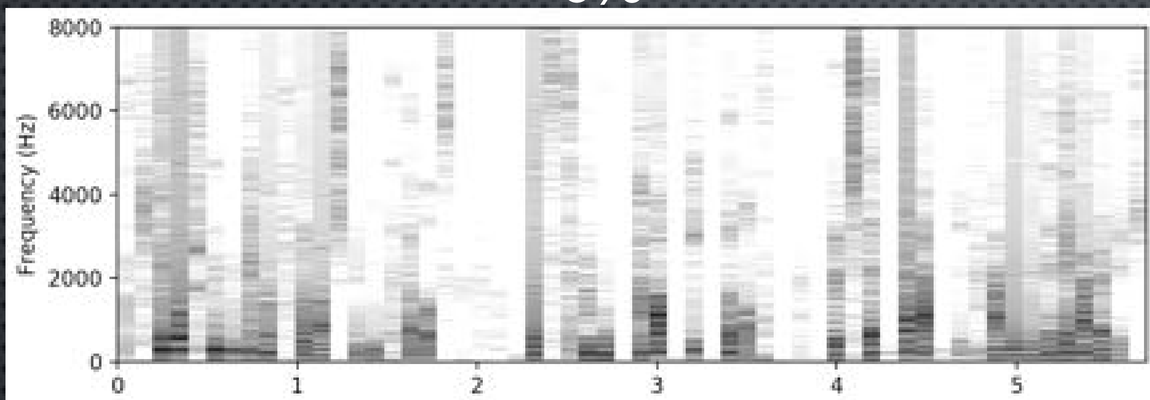
**Small** window size
**High** temporal resolution
**Low** spectral resolution



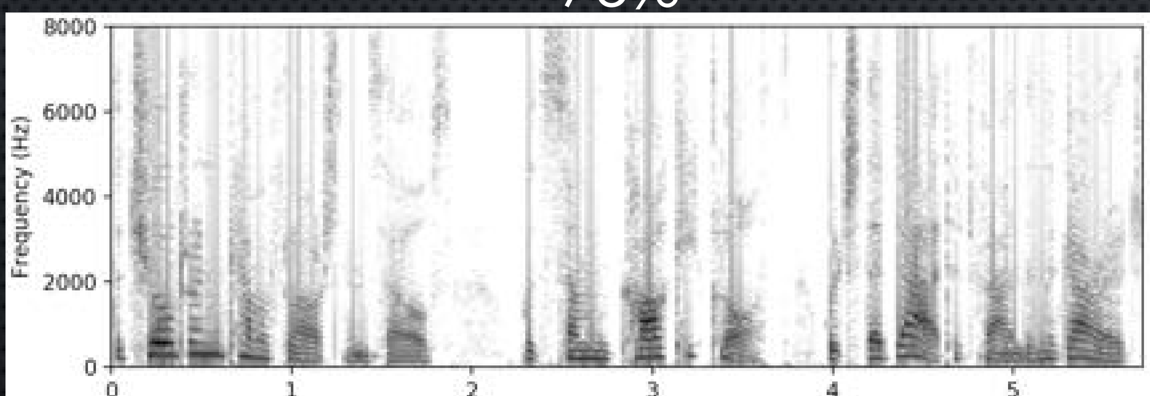

**Large** window size
**Low** temporal resolution
**High** spectral resolution

# Window overlapping

0%




75%

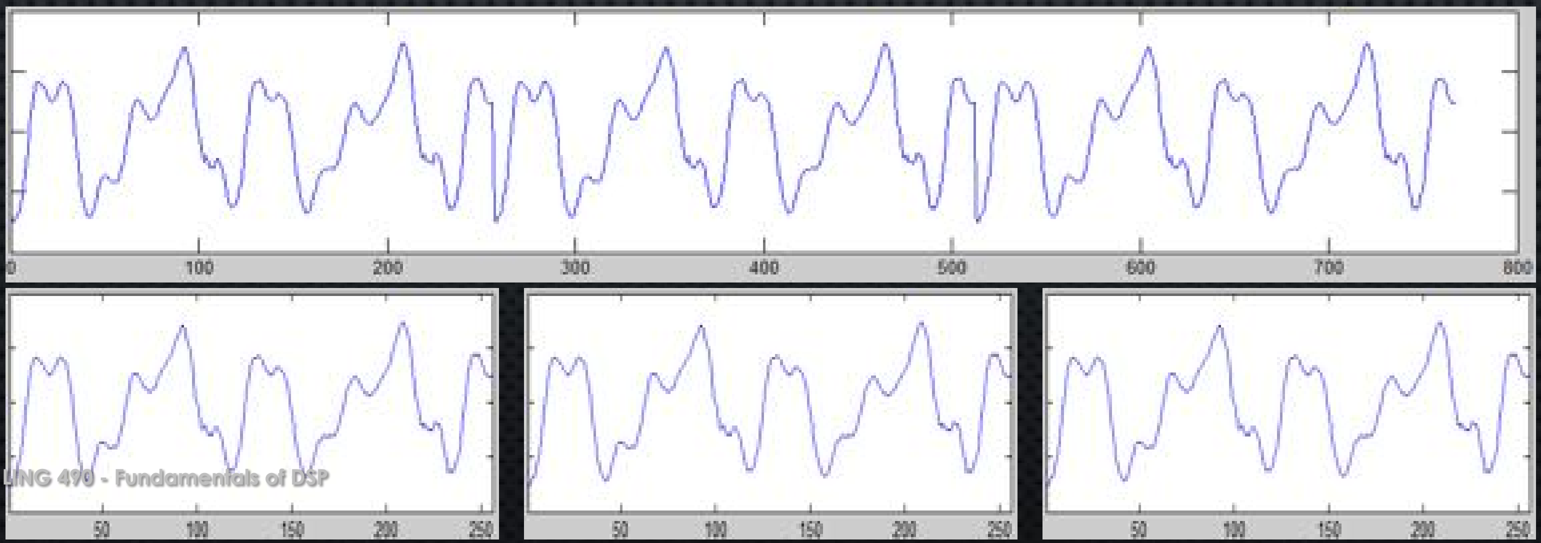



Window size: 100 ms

**Rule of thumb**: **minimum 50%** overlap for speech parameter extraction; more for nice display

# "Rectangular" windowing

- Fourier transform assumes that signals are **periodic**
  - When extracting a portion of the signal, a **rectangular window** is implicitly used
  - The signal is treated as a periodic signal with a period of the window size
  - The Fourier transform finds frequency components to model **this** signal:
- The signal now has implicit discontinuities which lead to lots of spurious frequency components and hence "noise"

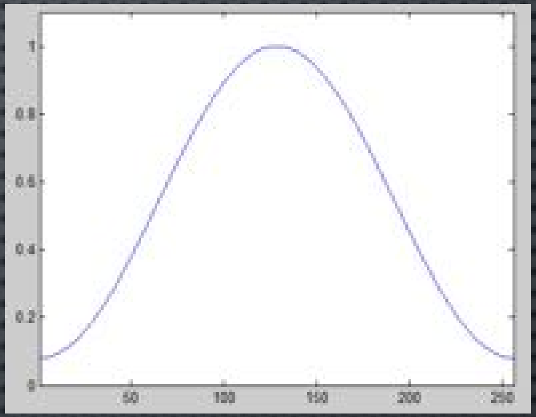


# Recall quantisation and clipping

- Both result in sharp edges in the time domain waveform
- "edges" introduce energy across a wide range of frequencies

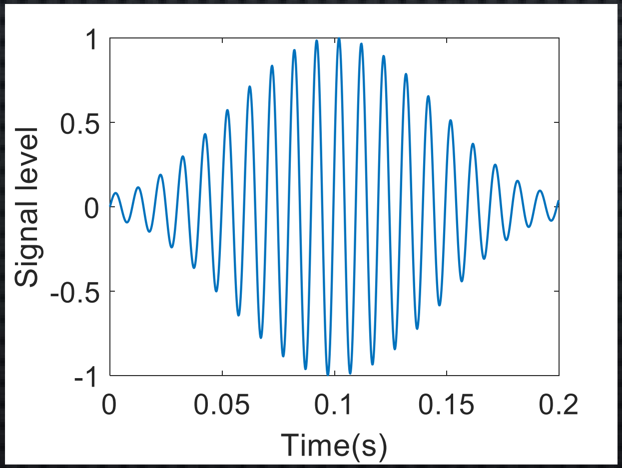
# Solution: use a tapering window

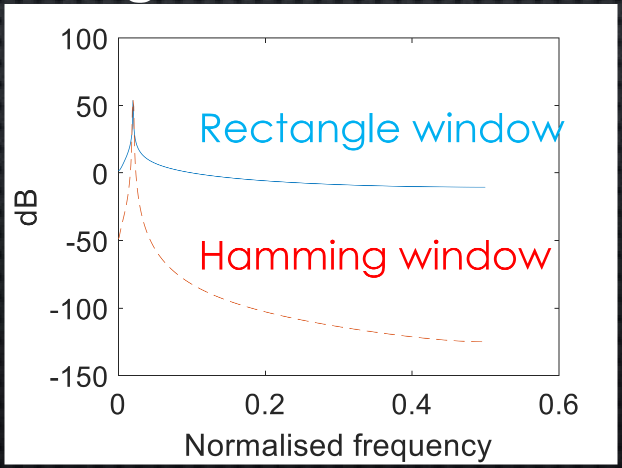
- Multiply the signal by a window like this:
  - **Hamming** window, also known as a **raised cosine** window, which reduces the signal to near zero at the edges



$$w = 0.54 - 0.46 \cdot \cos(2\,\pi\,(t/N))$$

- Similar windows: Hanning

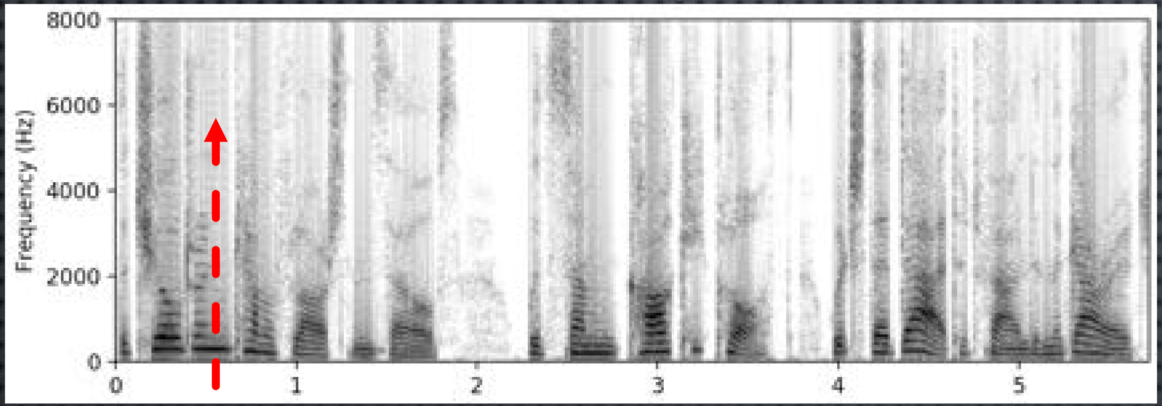


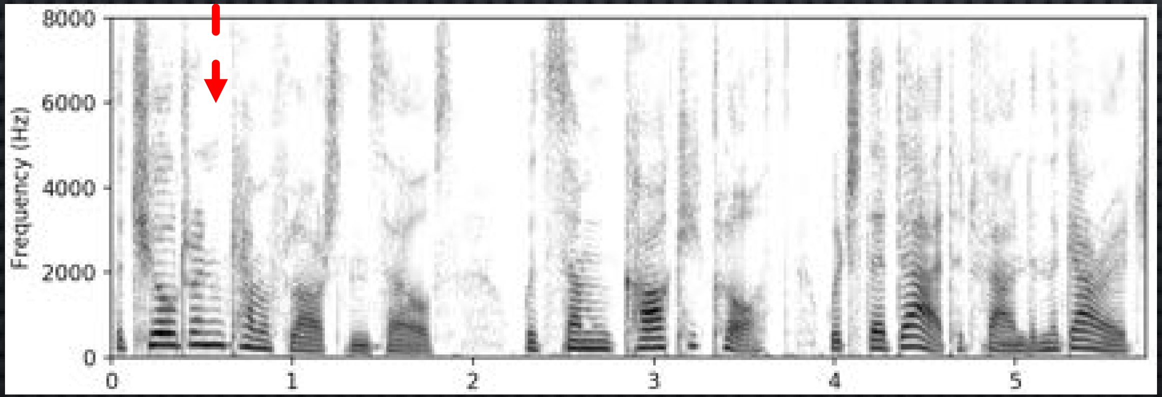

**There is a price**: loss of information at the window edges. For this reason, use windows which overlap by at least 50%

# Effect of window function

With rectangle window



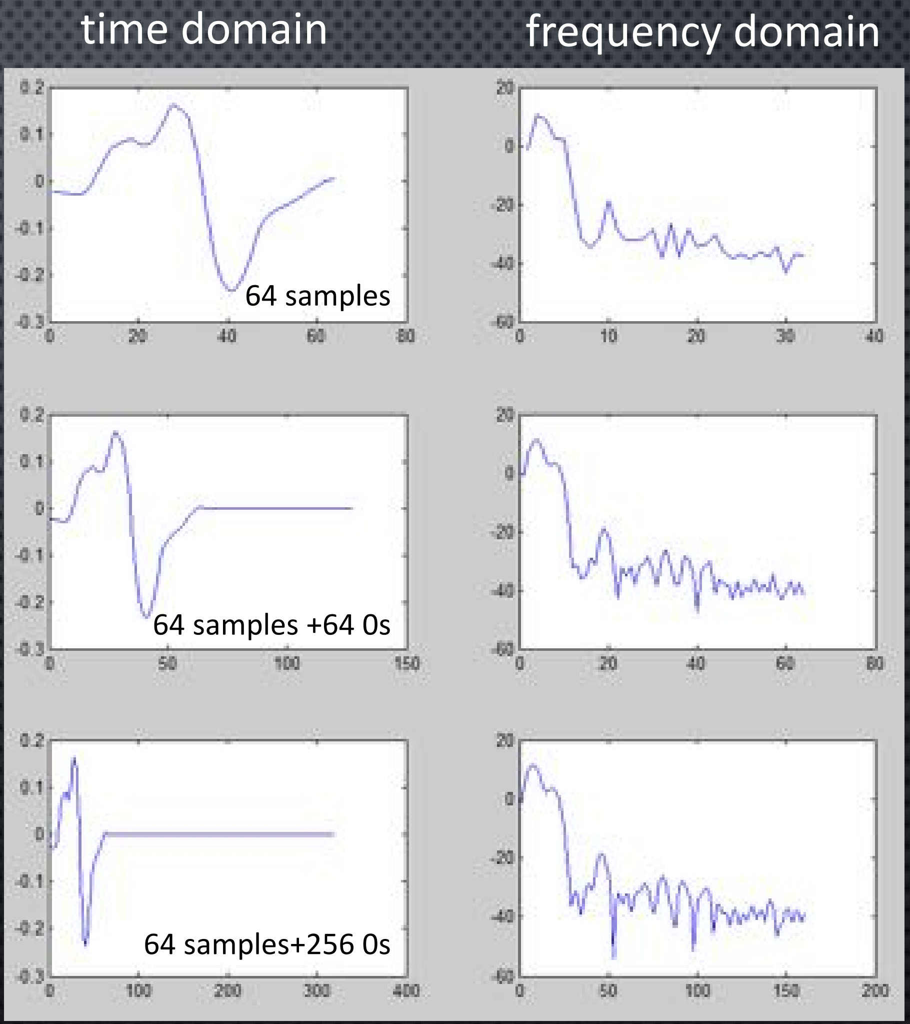

With Hamming window




Window size: 20 ms
Overlap: 50%

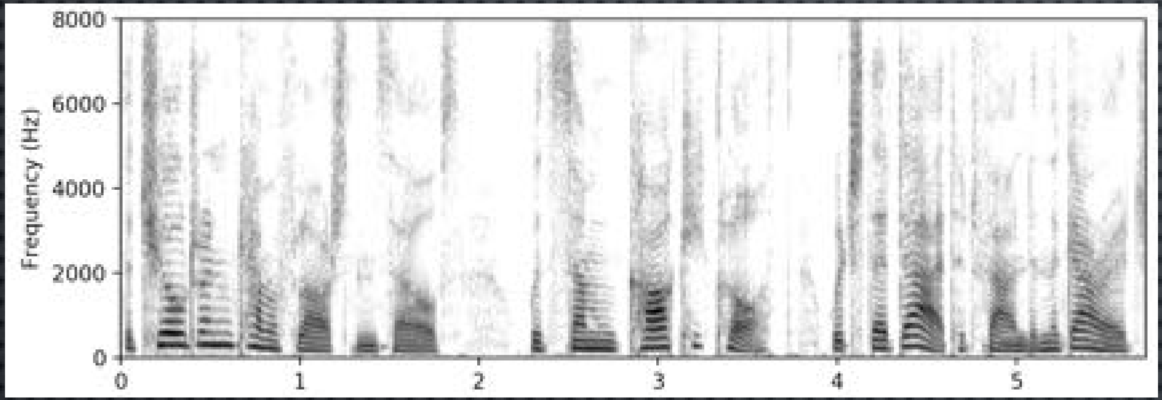
# Further improvement: Spectral smoothing

- How can a broadband spectrogram (with low frequency resolution) be improved for better display?
- Solution: add zeros ("zero padding") to shorter signal;
  - the effect is to interpolate (smooth) the spectrum
- **We do NOT gain spectral resolution**!



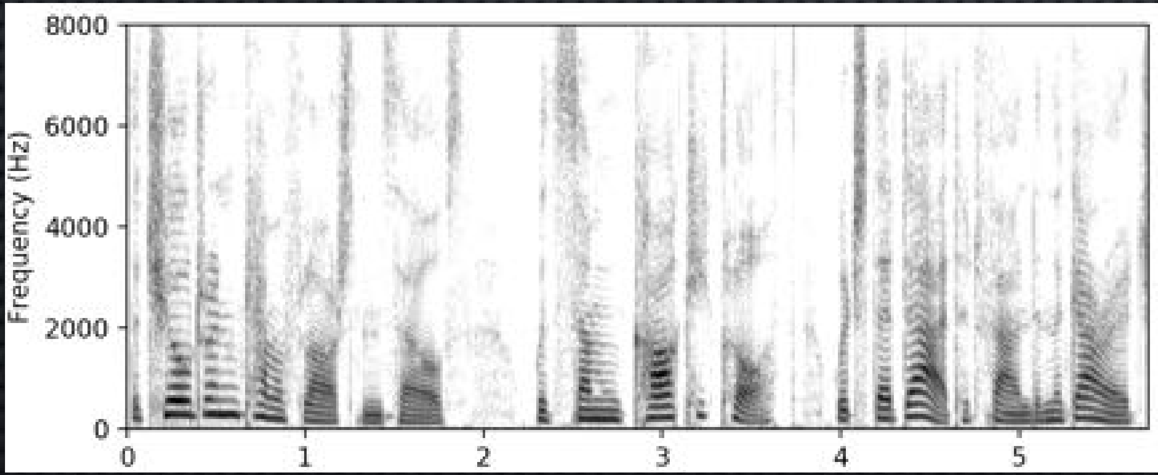
# Effect of spectral smoothing

FFT size: 320 (without zero padding)




Window size: 20 ms
Overlap: 50%
Hanning window

FFT size: 1024 (704 0s padded)



# Implementation steps:

1. Divide s into *N* windows(frames) of *K* samples
   - If the N * K is greater than the total number of samples, pad zeros to the end of of s.
   - For overlapping, work out the number of sample shift for each consecutive frame. Apply Hamming or Hanning window function to the signal of each frame by multiplying the two
2. Perform FFT on each frame, with given FFT size
3. Use the first half of FFT outputs for the spectrum of each frame
4. Compute the magnitude of FFT outputs for each frame

# Implementation steps:

5. Flip the array/matrix upside down, making sure low frequencies at the bottom of the spectrogram

6. Plot the spectrogram