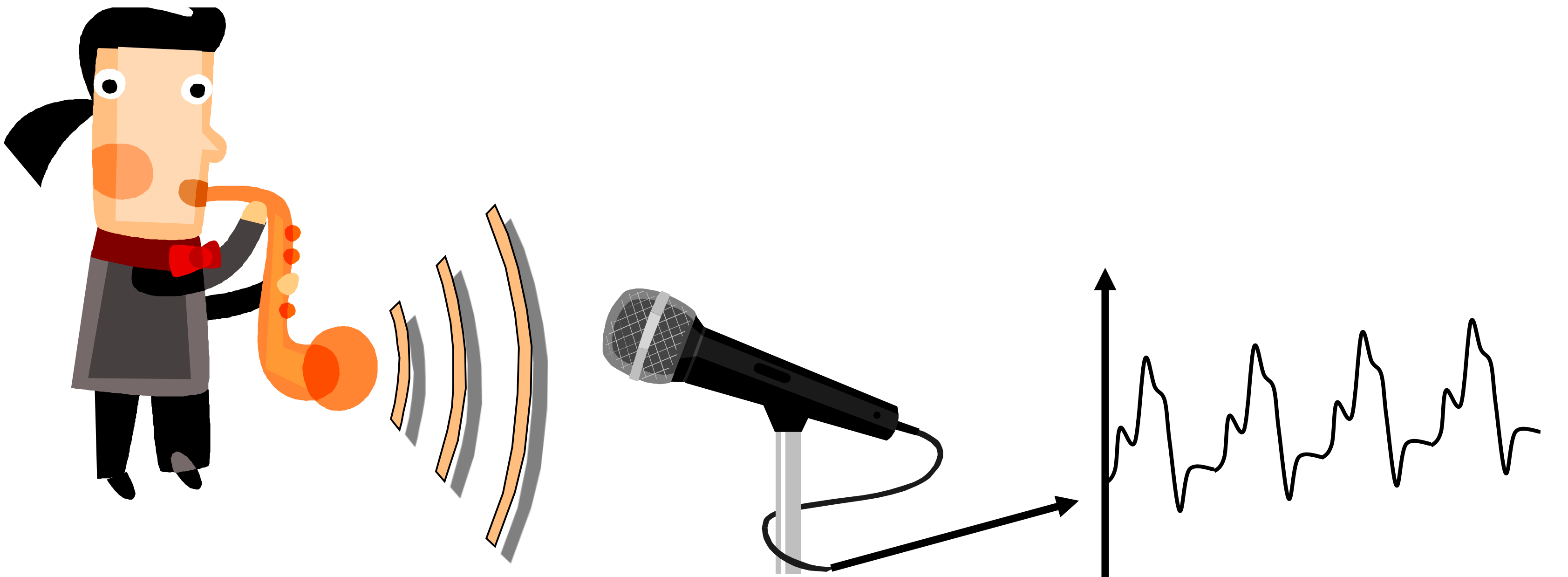
Topic 2

Signal Processing Review

(Some slides are adapted from Bryan Pardo’s course slides on Machine Perception of Music)

Recording Sound



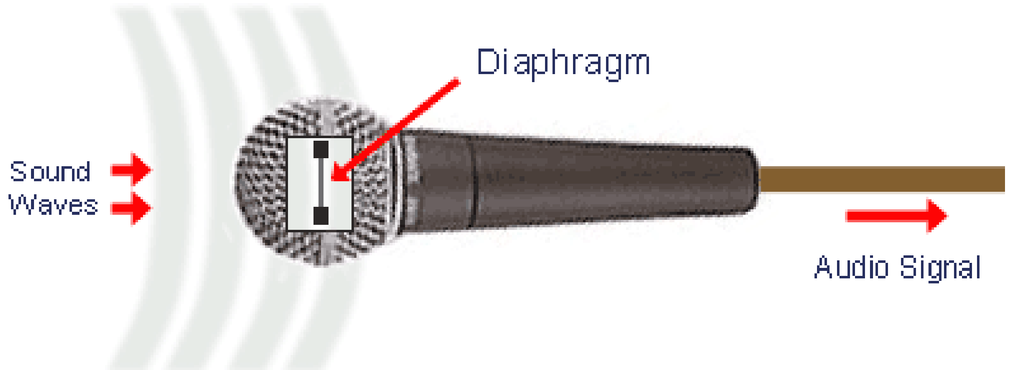
Mechanical Vibration

Pressure Waves

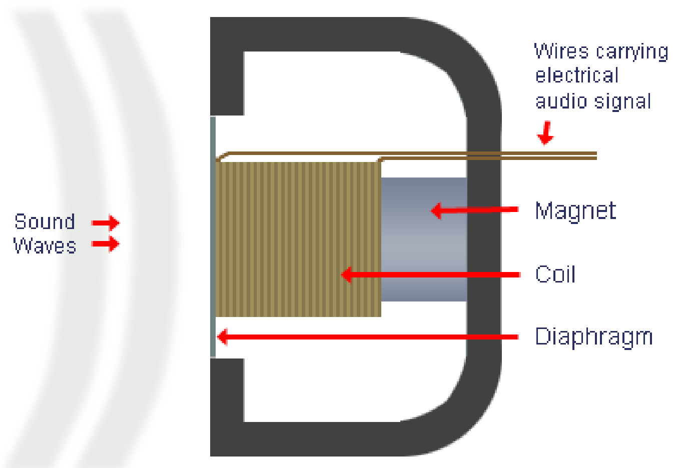
Motion->Voltage Transducer

Voltage over time

Microphones

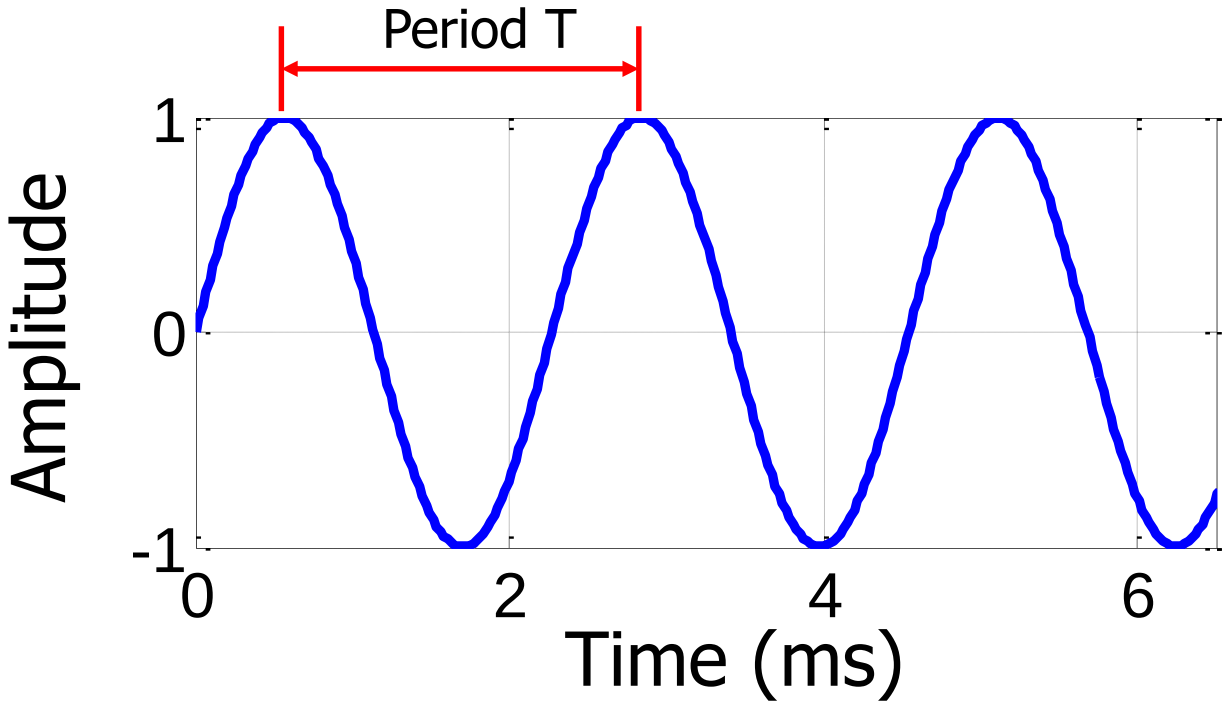


Cross-Section of Dynamic Microphone



http://www.mediacollege.com/audio/microphones/how-microphones-work.html

Pure Tone = Sine Wave



440Hz

$$x(t) = A \sin(2\pi f t + \varphi)$$

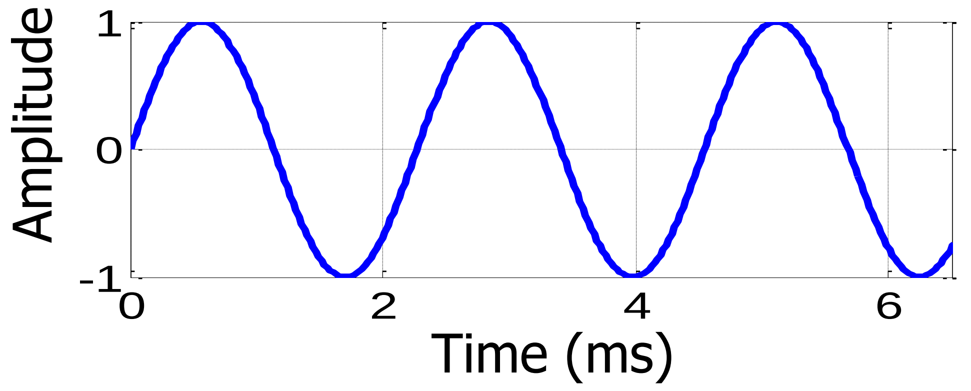
time amplitude frequency initial phase

Reminders

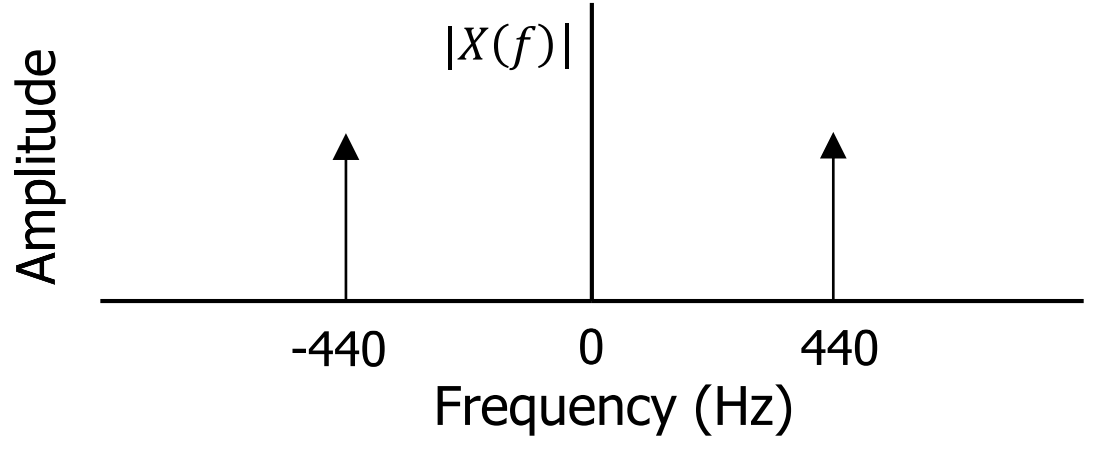
- Frequency, $f = 1/T$, is measured in cycles per second , a.k.a. *Hertz (Hz).*
- One cycle contains 2π *radians.*
- Angular frequency Ω, is measured in radians per second and is related to frequency by $\Omega = 2\pi f$.
- So we can rewrite the sine wave as

$$x(t) = A\sin(\Omega t + \varphi)$$

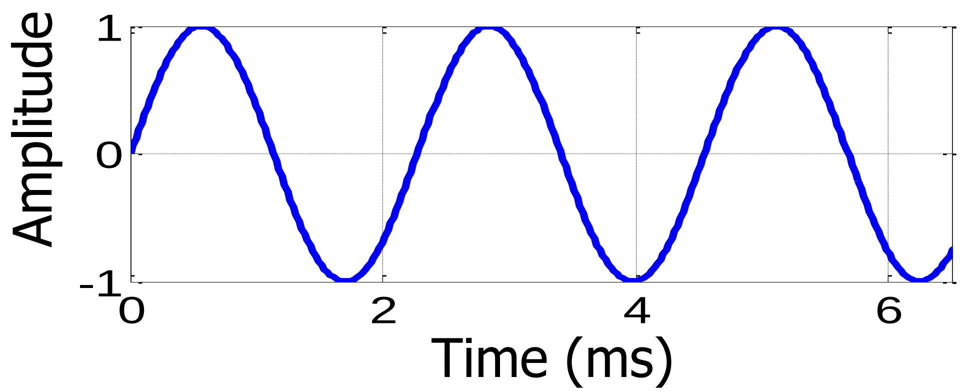
Fourier Transform



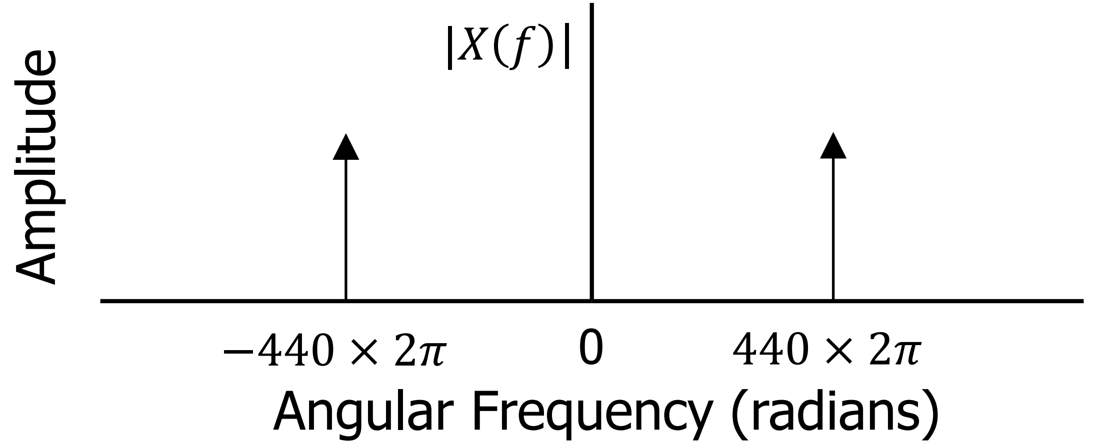
$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi f t} dt$$



We can also write

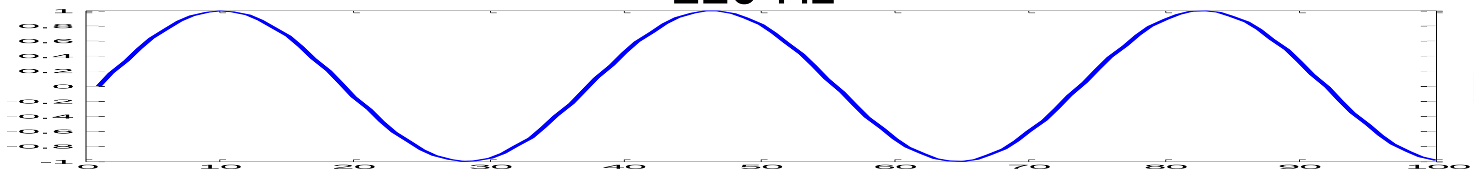


$$X(\Omega) = \int_{-\infty}^{\infty} x(t) e^{-j\Omega t} dt$$



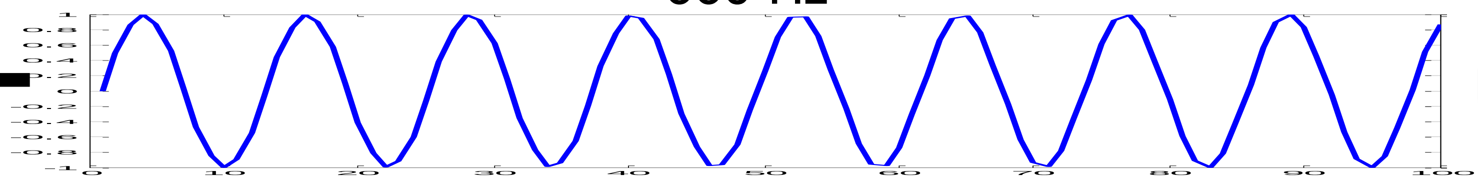
Complex Tone = Sine Waves

220 Hz



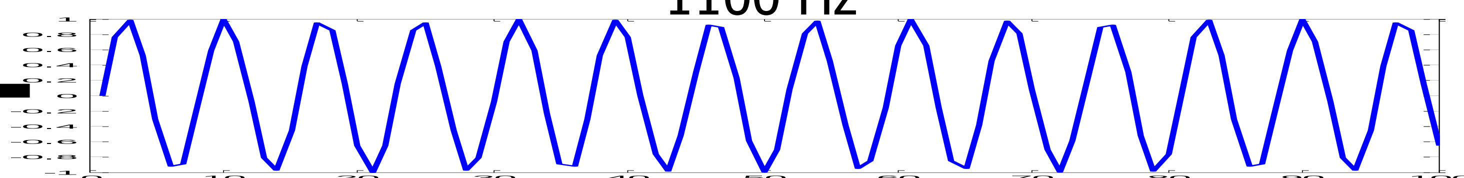
\+

660 Hz

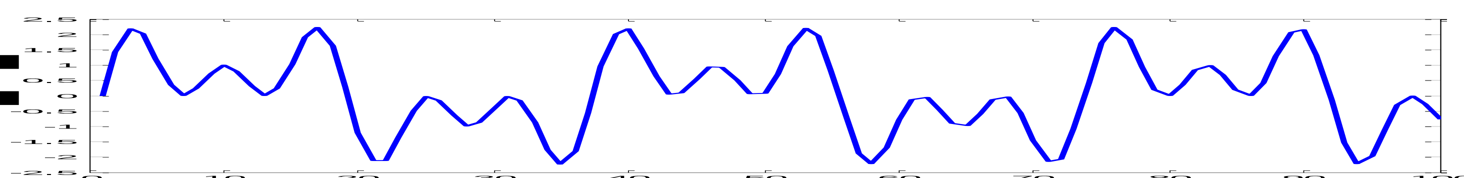


\+

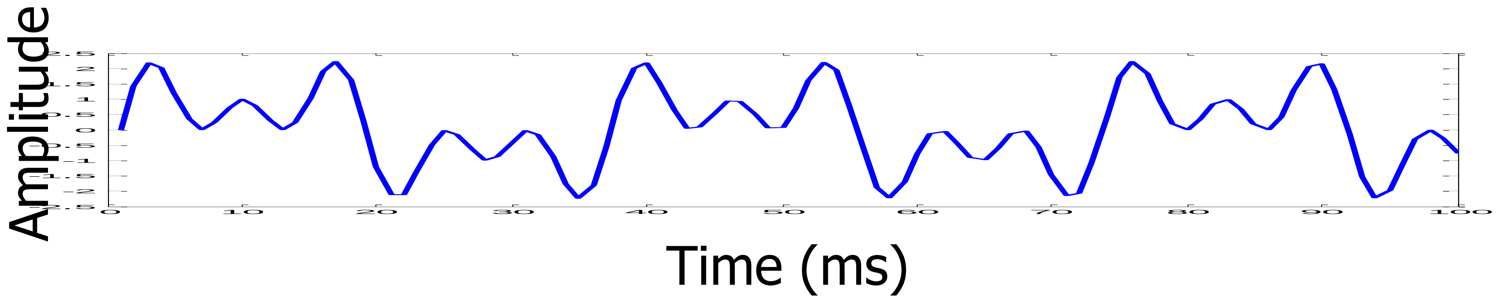
1100 Hz



=

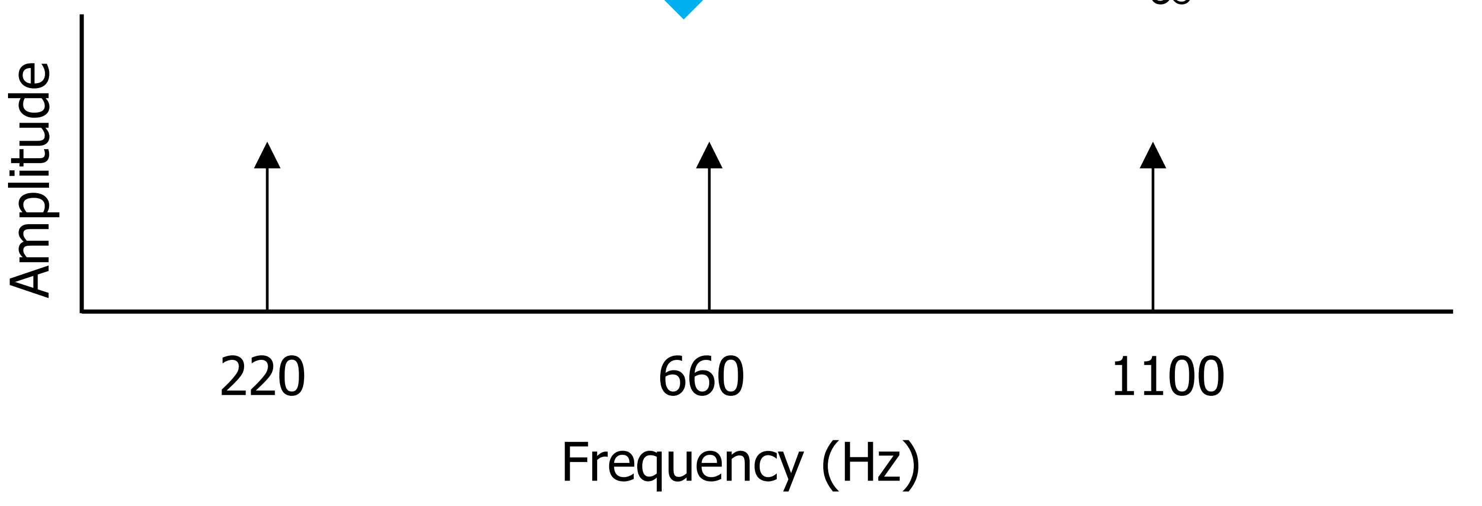


Frequency Domain



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

$|X(f)|$



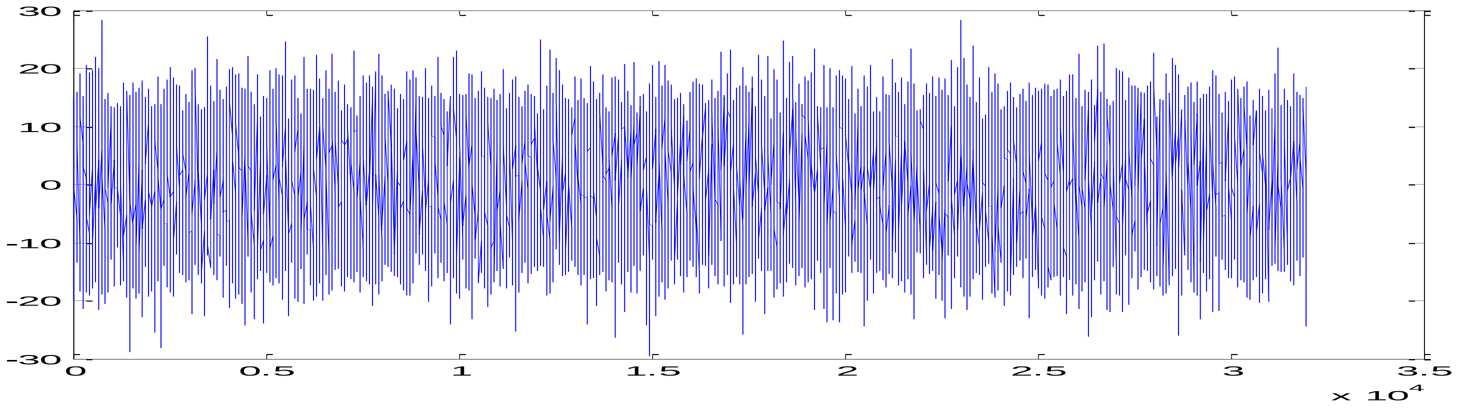
Harmonic Sound

- 1 or more sine waves
- Strong components at integer multiples of a fundamental frequency (F0) in the range of human hearing (20 Hz ~ 20,000 Hz)

- Examples
  - 220 + 660 + 1100 is harmonic
  - 220 + 375 + 770 is not harmonic

Noise

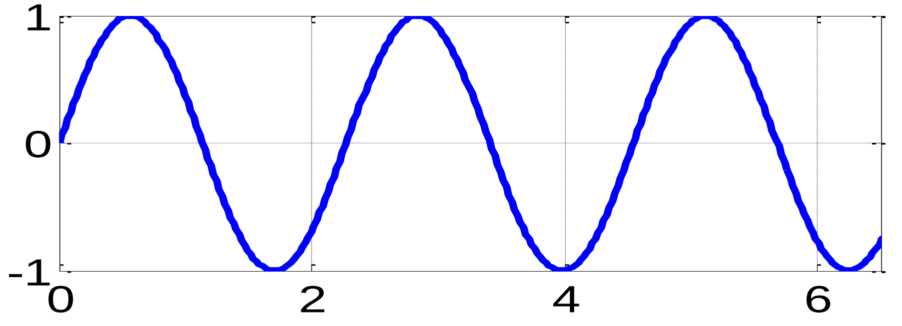
- Lots of sines at random freqs. = NOISE
- Example: 100 sines with random frequencies, such that $100 < f < 10000$.

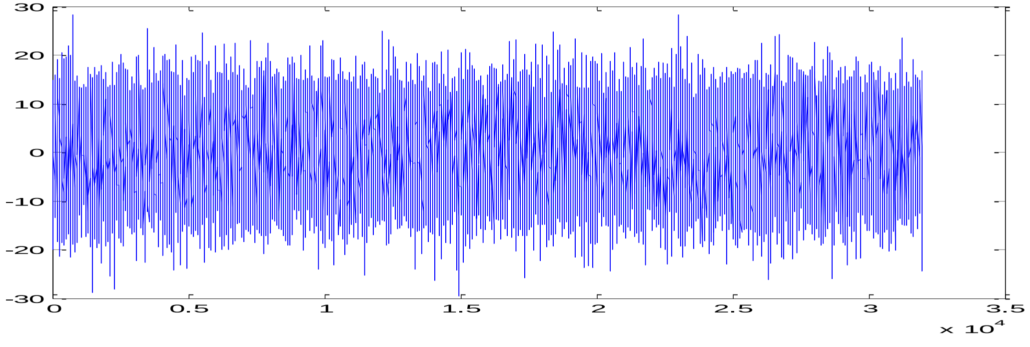


How strong is the signal?

- Instantaneous value?
- Average value?
- Something else?

$x(t)$





Acoustical or Electrical

- Acoustical

Average intensity

$$I = \frac{1}{\rho c}\frac{1}{T_D}\int_0^{T_D} x^2(t)dt$$

View $x(t)$ as sound pressure

density

sound speed

- Electrical

Average power

$$P = \frac{1}{R}\frac{1}{T_D}\int_0^{T_D} x^2(t)dt$$

View $x(t)$ as electric voltage

resistance

Root-Mean-Square (RMS)

$$x_{RMS} = \sqrt{\frac{1}{T_D}\int_0^{T_D} x^2(t)dt}$$

- T_D should be long enough.
- $x(t)$ should have 0 mean, otherwise the DC component will be integrated.
- For sinusoids

$$x_{RMS} = \sqrt{\frac{1}{T}\int_0^{T} A^2\sin^2(2\pi f t)dt} = \sqrt{A^2/2} = 0.707A$$

Sound Pressure Level (SPL)

- Softest audible sound intensity 0.000000000001 watt/m^2
- Threshold of pain is around 1 watt/m^2
- 12 orders of magnitude difference
- A log scale helps with this
- The decibel (dB) scale is a log scale, with respect to a reference value

The Decibel

- A logarithmic measurement that expresses the magnitude of a physical quantity (e.g., power or intensity) relative to a specified reference level.
- Since it expresses a ratio of two (same unit) quantities, it is dimensionless.

$$L - L_{\text{ref}} = 10 \log_{10}\left(\frac{I}{I_{\text{ref}}}\right)$$

$$= 20 \log_{10}\left(\frac{x_{RMS}}{x_{\text{ref},RMS}}\right)$$

Lots of references!

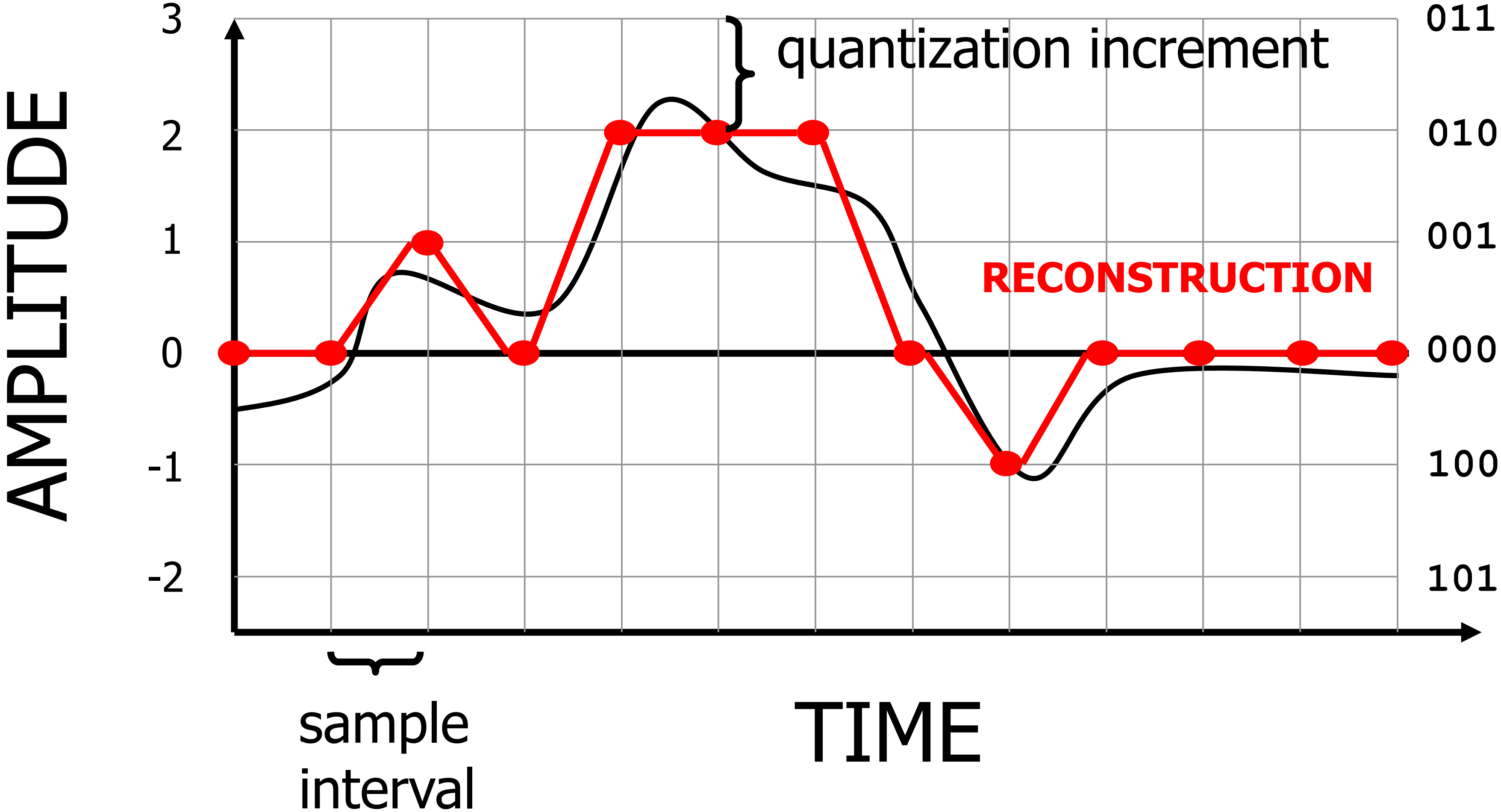
- **dB-SPL** – A measure of sound pressure level. 0dB-SPL is approximately the quietest sound a human can hear, roughly the sound of a mosquito flying 3 meters away.

- **dbFS** – relative to digital full-scale. 0 dbFS is the maximum allowable signal. Values typically negative.
- **dBV** – relative to 1 Volt RMS. 0dBV = 1V.
- **dBu** – relative to 0.775 Volts RMS with an unloaded, open circuit.
- **dBmV** – relative to 1 millivolt across 75 Ω. Widely used in cable television networks.
- ……

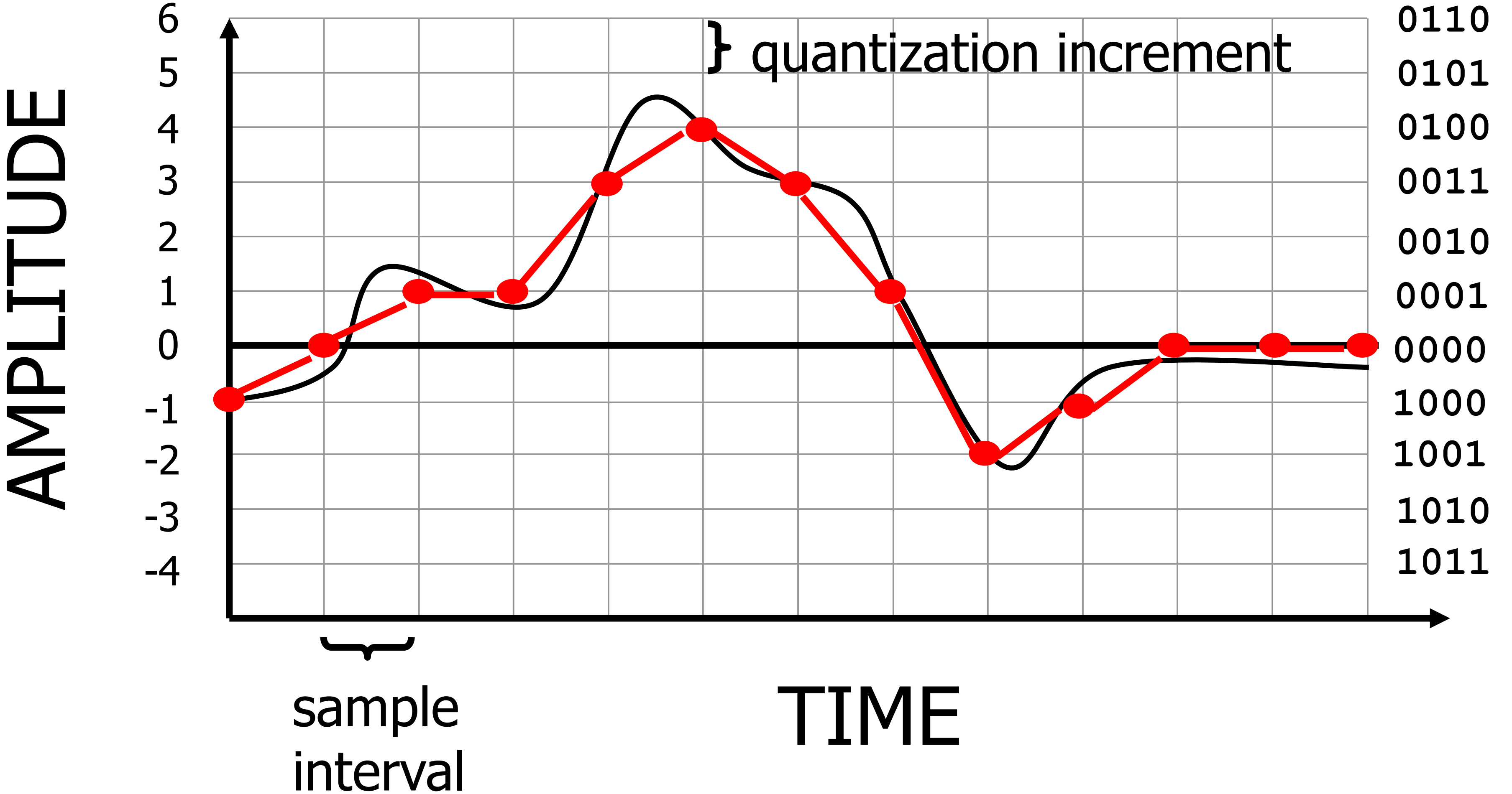
Typical Values

• Jet engine at 3m	140 db-SPL
• Pain threshold	130 db-SPL
• Loud motorcycle, 5m	110 db-SPL
• Vacuum cleaner	80 db-SPL
• Quiet restaurant	50 db-SPL
• Rustling leaves	20 db-SPL
• Human breathing, 3m	10 db-SPL
• Hearing threshold	0 db-SPL

Digital Sampling



More quantization levels = more dynamic range



Bit Depth and Dynamics

- More bits = more quantization levels = better sound

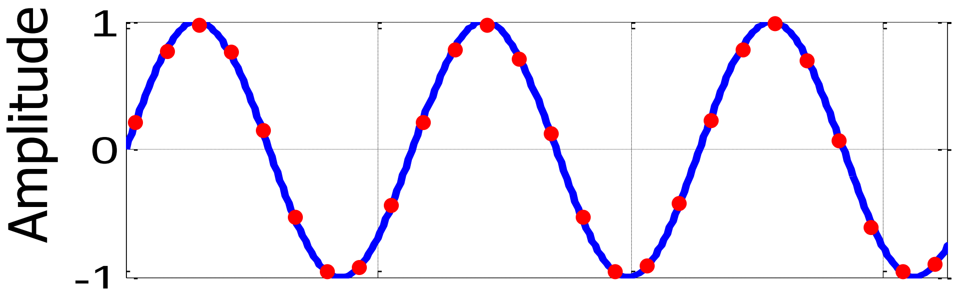
- Compact Disc: 16 bits = 65,536 levels
- POTS (plain old telephone service): 8 bits = 256 levels

- Signal-to-quantization-noise ratio (SQNR), if the signal is uniformly distributed in the whole range

$$\mathrm{SQNR} = 20 \log_{10} 2^N \approx 6.02N \text{ dB}$$

  – E.g., $N = 16$ bits depth gives about 96dB SQNR.

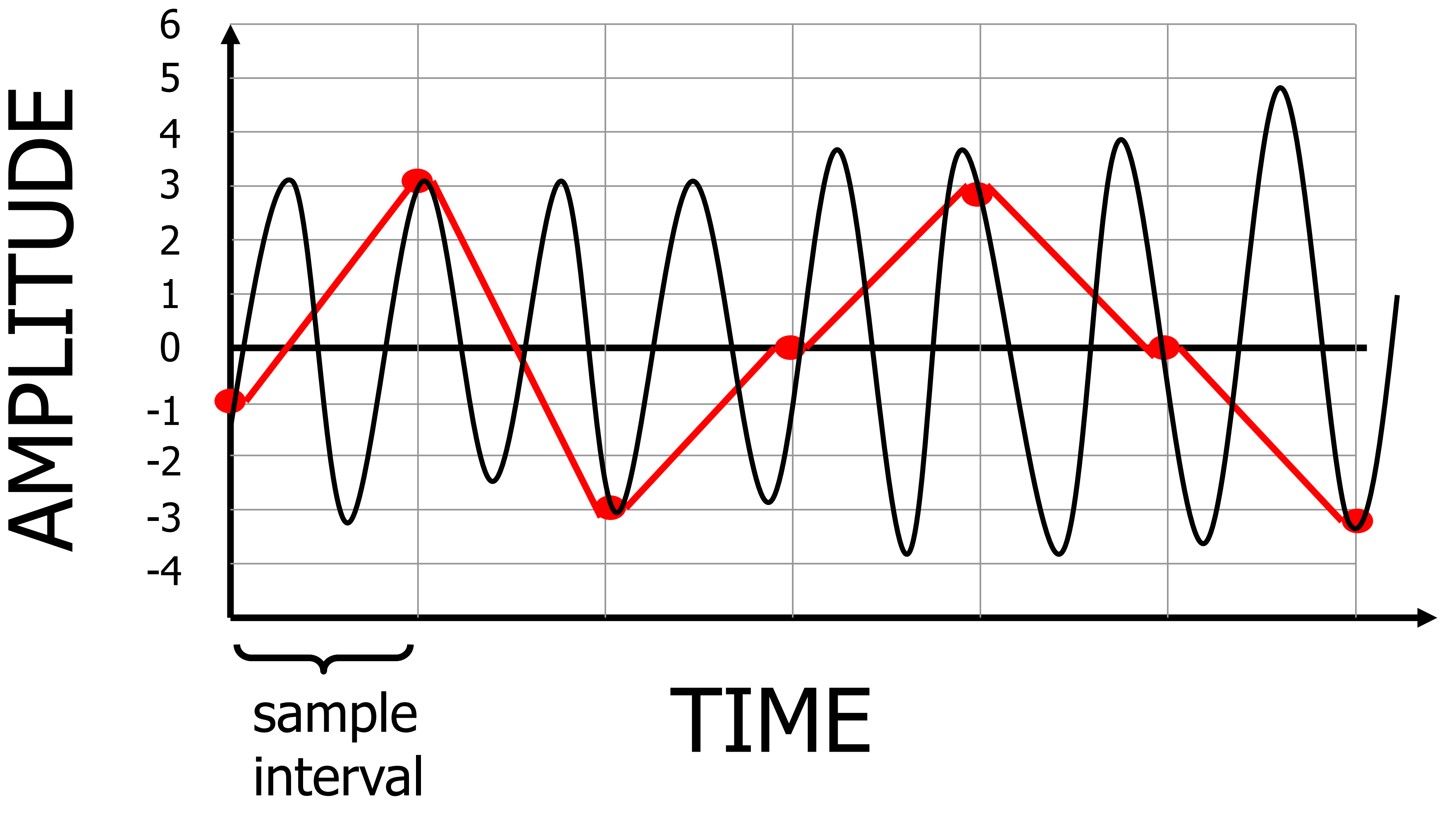
RMS



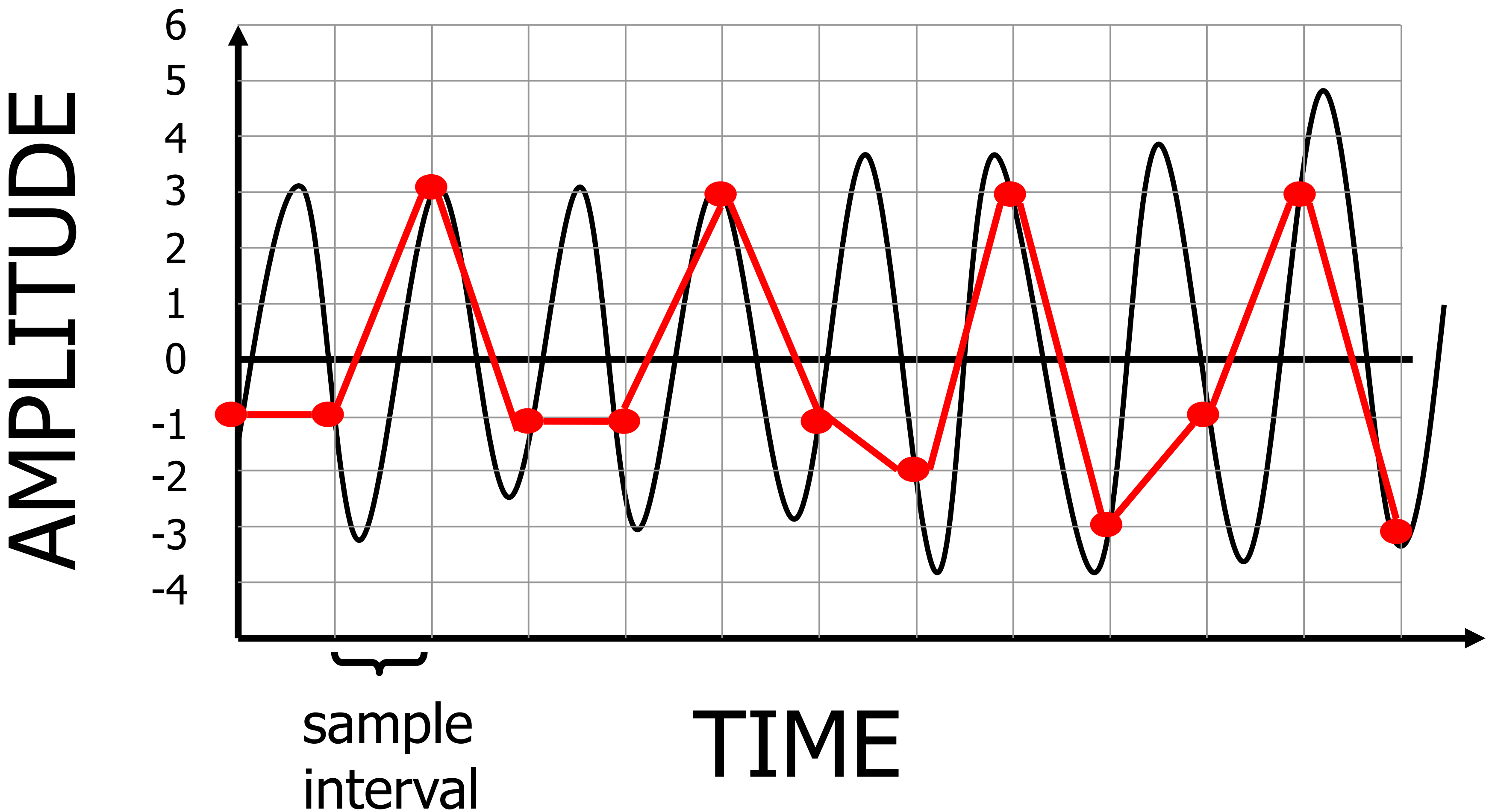
The red dots form the discrete signal $x[n]$

$$x_{RMS} = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} x^2[n]}$$

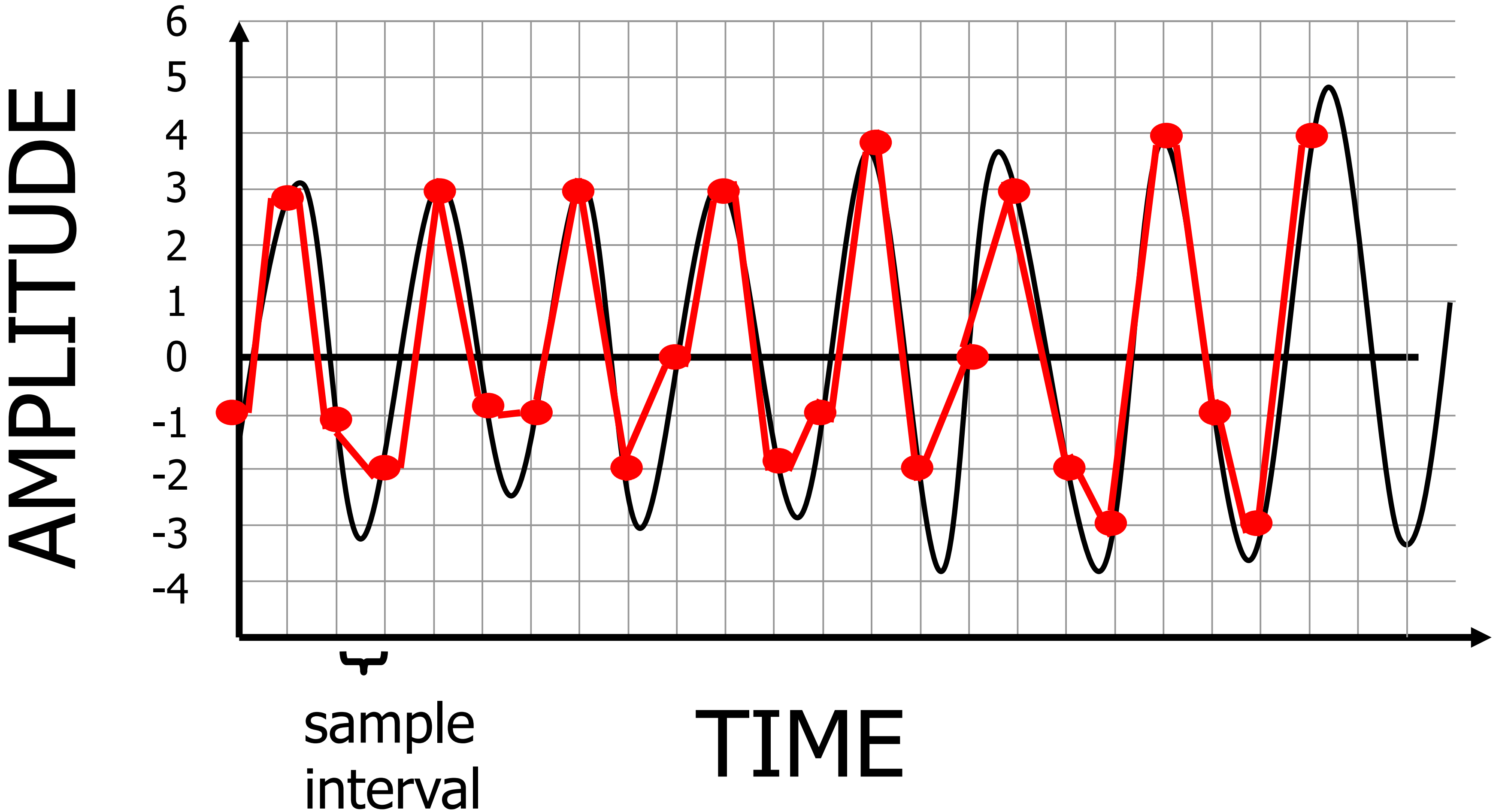
Aliasing and Nyquist



Aliasing and Nyquist



Aliasing and Nyquist

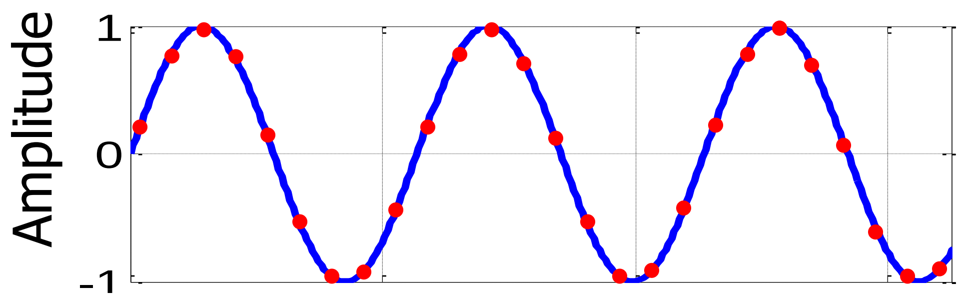


Nyquist-Shannon Sampling Theorem

- You can’t reproduce the signal if your sample rate isn’t faster than twice the highest frequency in the signal.

- Nyquist rate: twice the frequency of the highest frequency in the signal.
  - A property of the continuous-time signal.
- Nyquist frequency: half of the sampling rate
  - A property of the discrete-time system.

Discrete-Time Fourier Transform (DTFT)

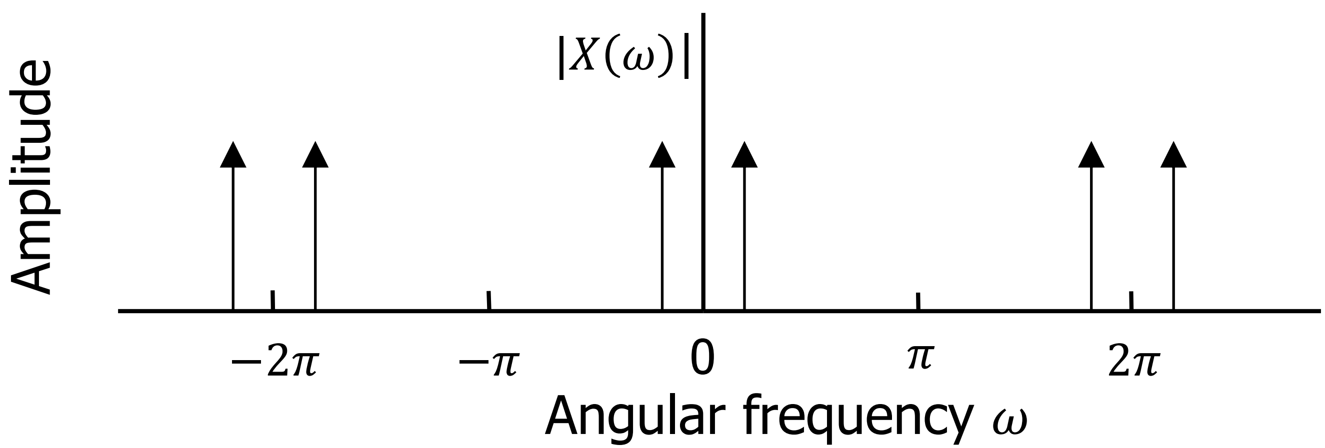


The red dots form the discrete signal $x[n]$, where $n = 0, \pm 1, \pm 2, \dots$

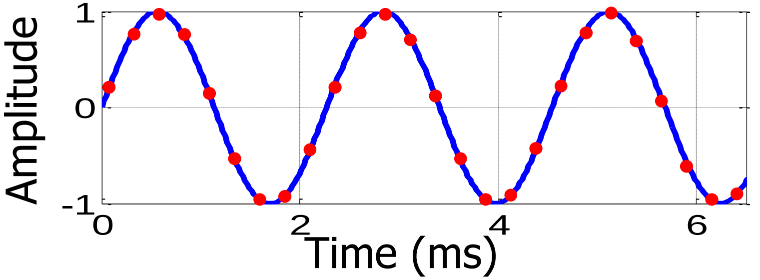
$X(\omega)$ is Periodic.
We often only show $[-\pi, \pi]$

ω is a continuous variable

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$



Relation between FT and DTFT



Sampling: $x[n] = x_c(nT)$

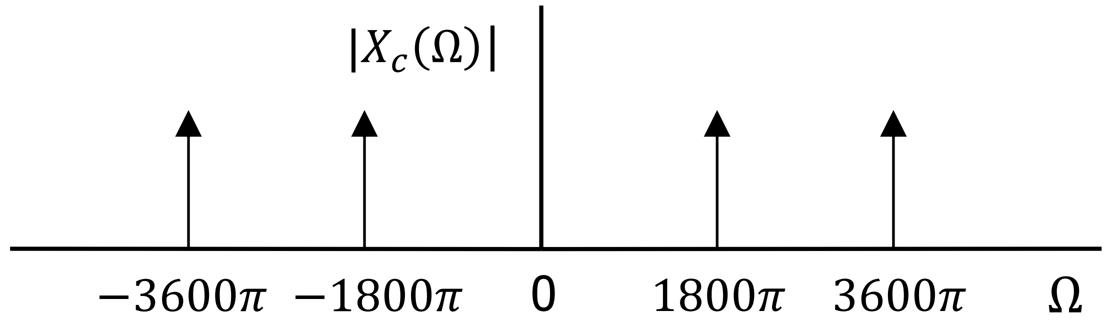
FT: $X_c(\Omega) = \int_{-\infty}^{\infty} x_c(t) e^{-j\Omega t} dt$

DTFT: $X(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$

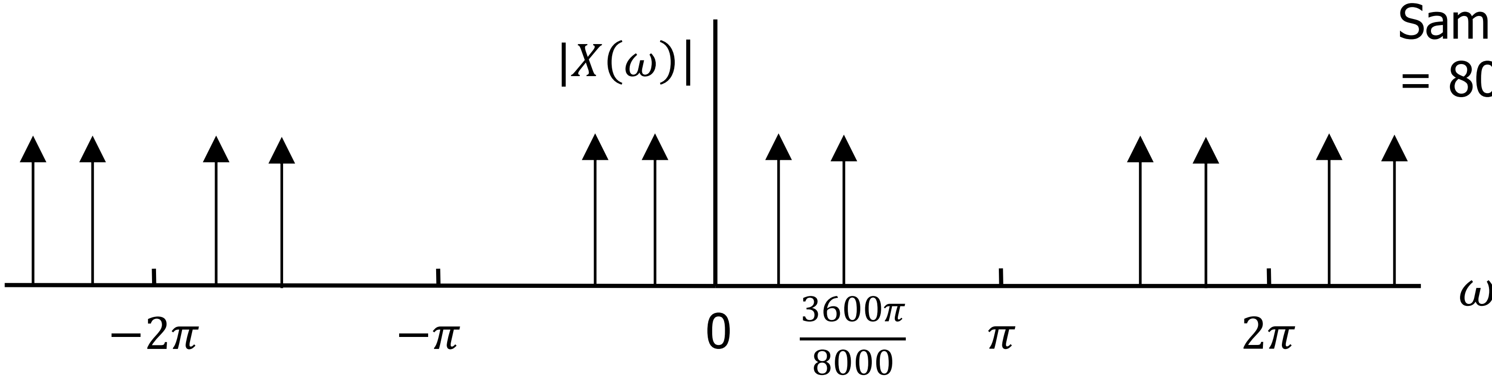
$$X(\omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_c\left(\frac{\omega}{T} + \frac{2\pi k}{T}\right)$$

- Scaling: $\omega = \Omega T$, i.e., $\omega = 2\pi$ corresponds to $\Omega = \frac{2\pi}{T} = 2\pi f_s$, which corresponds to $f = f_s$.
- Repetition: $X(\omega)$ contains infinite copies of X_c, spaced by 2π.

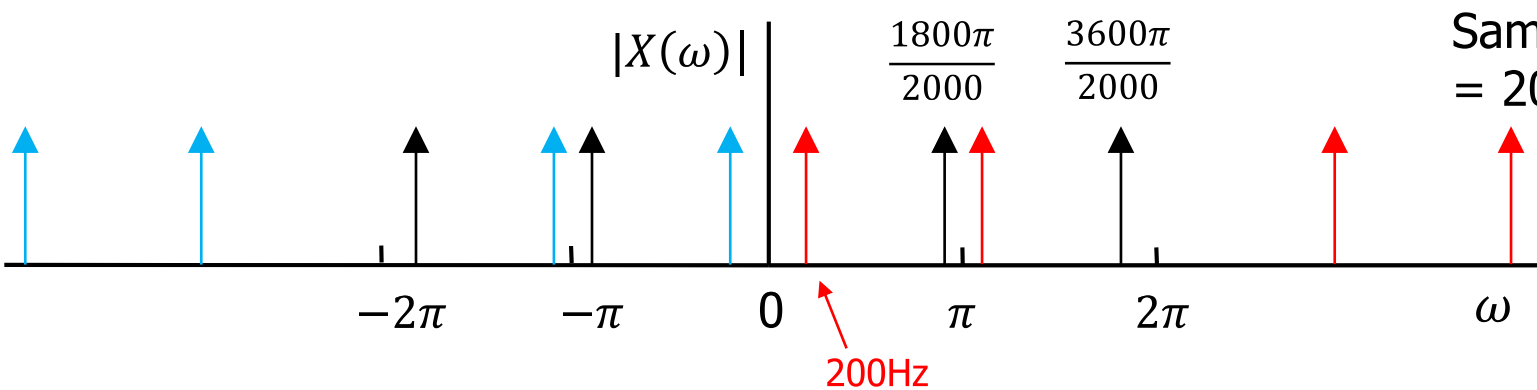
Aliasing



Complex tone
900Hz + 1800Hz



Sampling rate
= 8000Hz



Sampling rate
= 2000Hz

Fourier Series

- FT and DTFT do not require the signal to be periodic, i.e., the signal may contain arbitrary frequencies, which is why the frequency domain is continuous.

- Now, if the signal is periodic:

$$x(t+mT) = x(t) \quad \forall m \in \mathrm{Z}$$

- It can be reproduced by a series of sine and cosine functions:

$$x(t) = A_0 + \sum_{n=1}^{\infty} [A_n \cos(\Omega_n t) + B_n \sin(\Omega_n t)]$$

- In other words, the frequency domain is discrete.

Discrete Fourier Transform (DFT)

- FT and DTFT are great, but the infinite integral or summations are hard to deal with.
- In digital computers, everything is discrete, including both the signal and its spectrum.

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}$$

frequency domain index

time domain index

Length of the signal, i.e. length of DFT

DFT and IDFT

DFT: $$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}$$

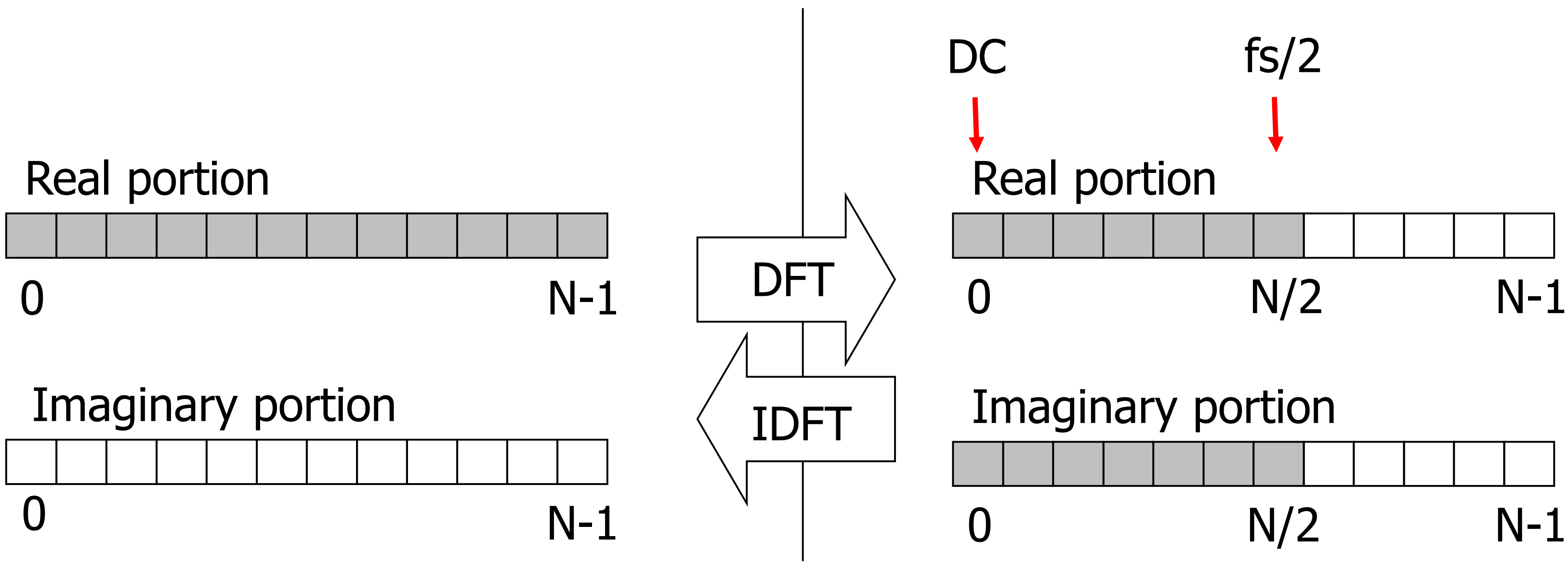
IDFT: $$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N}$$

- Both $x[n]$ and $X[k]$ are discrete and of length N.
- Treats $x[n]$ as if it were infinite and periodic.
- Treats $X[k]$ as if it were infinite and periodic.
- Only one period is involved in calculation.

Discrete Fourier Transform

- If the time-domain signal has no imaginary part (like an audio signal) then the frequency-domain signal is conjugate symmetric around N/2.

Time domain $x[n]$ Frequency domain $X[k]$



Kinds of Fourier Transforms

Type of Transform	Example Signal
Fourier Transform Signals: continuous, aperiodic Spectrum: aperiodic, continuous	
Fourier Series Signals: continuous, periodic Spectrum: aperiodic, discrete	
Discrete Time Fourier Transform Signals: discrete, aperiodic Spectrum: periodic, continuous	
Discrete Fourier Transform Signals: discrete, periodic Spectrum: periodic, discrete	

Duality

Time domain

		continuous	discrete		
Frequency domain	continuous	**Fourier Transform**	**DTFT**	aperiodic	**Time domain**
	discrete	**Fourier Series**	**DFT**	periodic	
		aperiodic	periodic		

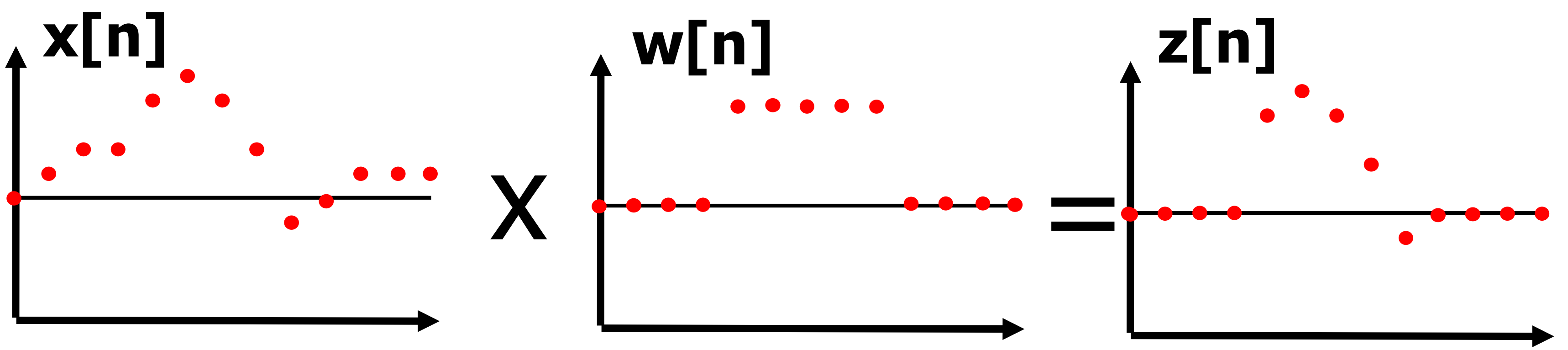
Frequency domain

The FFT

- Fast Fourier Transform
  - A much, much faster way to do the DFT
  - Introduced by Carl F. Gauss in 1805
  - Rediscovered by J.W. Cooley and John Tukey in 1965
  - The Cooley-Tukey algorithm is the one we use today (mostly)
  - Big O notation for this is **O(N *log* N)**
  - Matlab functions **fft** and **ifft** are standard.

Windowing

- A function that is zero-valued outside of some chosen interval.
  - When a signal (data) is multiplied by a window function, the product is zero-valued outside the interval: all that is left is the "view" through the window.

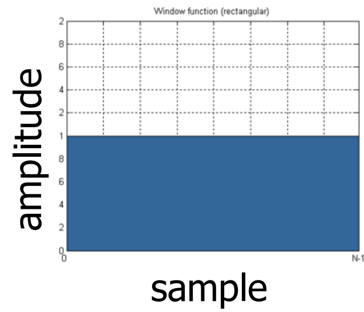


Example: windowing x[n] with a rectangular window

Some famous windows

- Rectangular

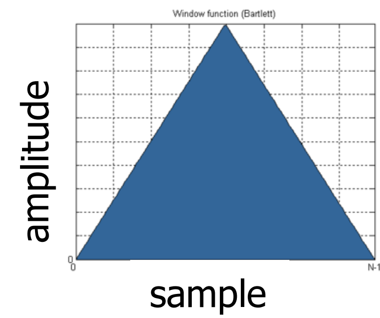
$$w[n] = 1$$



Note: we assume w[n] = 0 outside some range [0,N]

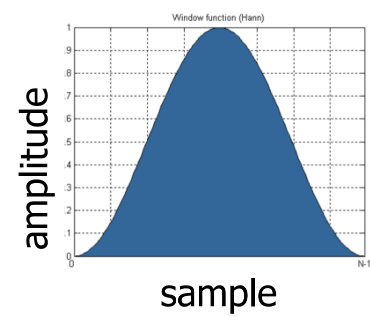
- Triangular (Bartlett)

$$w[n] = \frac{2}{N-1}\left(\frac{N-1}{2} - \left|n - \frac{N-1}{2}\right|\right)$$



- Hann

$$w[n] = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right)$$



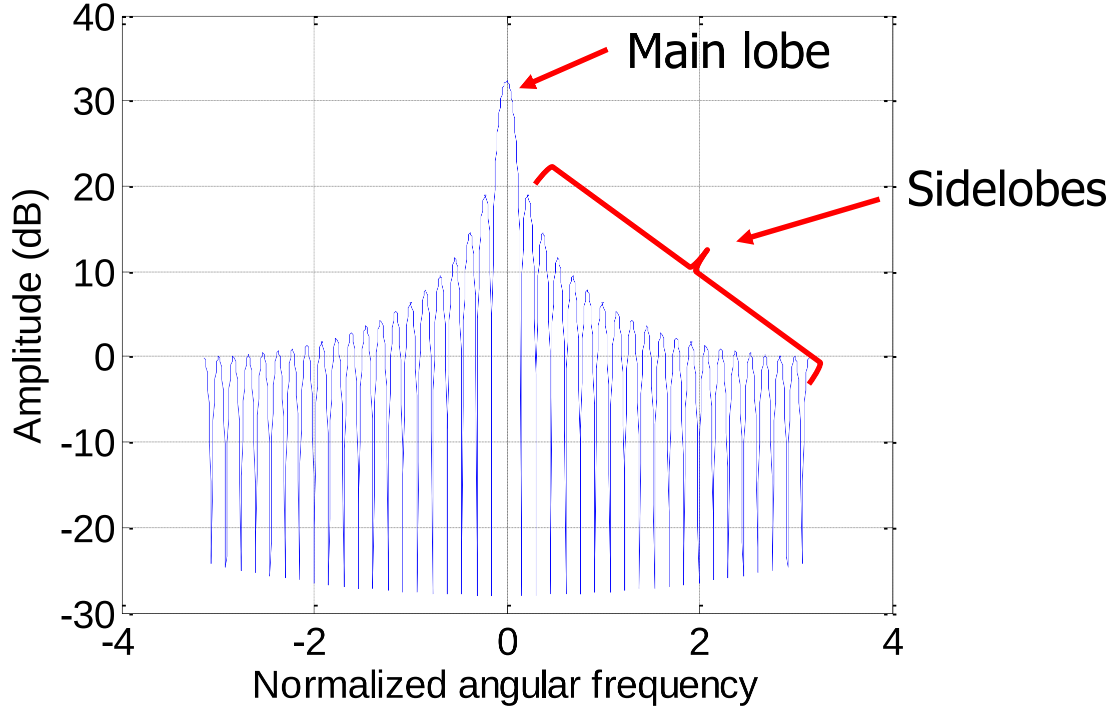
Why window shape matters

- Don’t forget that a DFT assumes the signal in the window is periodic
- The boundary conditions mess things up…unless you manage to have a window whose length is exactly 1 period of your signal
- Making the edges of the window less prominent helps suppress undesirable artifacts

Fourier Transform of Windows

```
L = 41;                     % window length
fftLen = 1024;              % fft length
w_rc = ones(L,1);           % rectangular window
wf_rc = 20*log10(abs(fft(w_rc, fftLen)));
figure; h = axes('FontSize', 16);
% frequency indices, make the positive and negative frequencies symmetric
fbins = [(-(fftLen-1)/2 : -1), (0 : fftLen/2)] * 2*pi/fftLen;
plot(h, fbins, [wf_rc(fftLen/2+2:end); wf_rc(1:fftLen/2+1)]);
grid on;
xlabel('Normalized angular frequency');
ylabel('Amplitude (dB)');
```

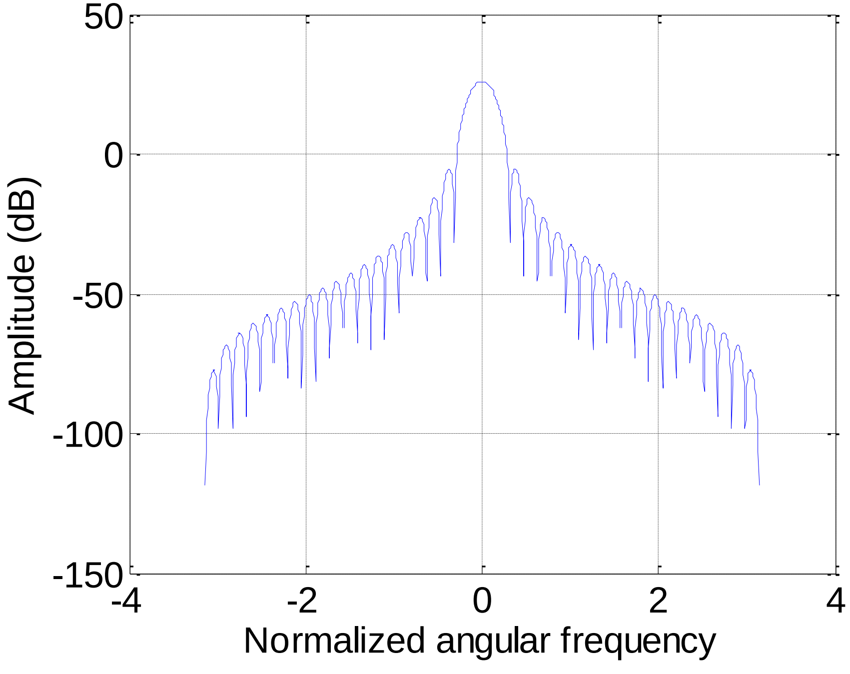
We want
- Narrow main lobe
- Low sidelobes



Which window is better?

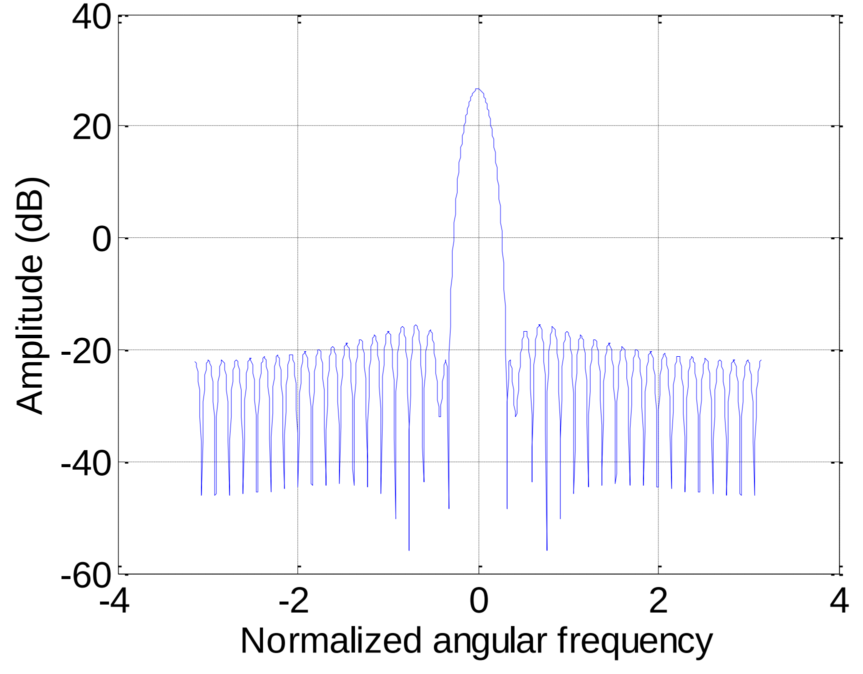
Hann window

$$w[n] = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right)$$



Hamming window

$$w[n] = 0.54 \; - 0.46 \times \cos\left(\frac{2\pi n}{N-1}\right)$$



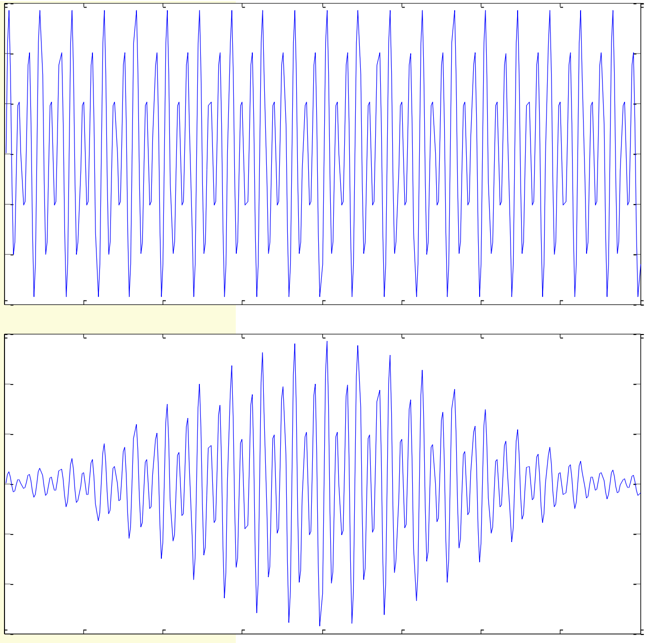
Multiplication v.s. Convolution

Time domain	Frequency Domain
$x[n] \cdot y[n]$	$\frac{1}{N} X[k] * Y[k]$
$x[n] * y[n]$	$X[k] \cdot Y[k]$

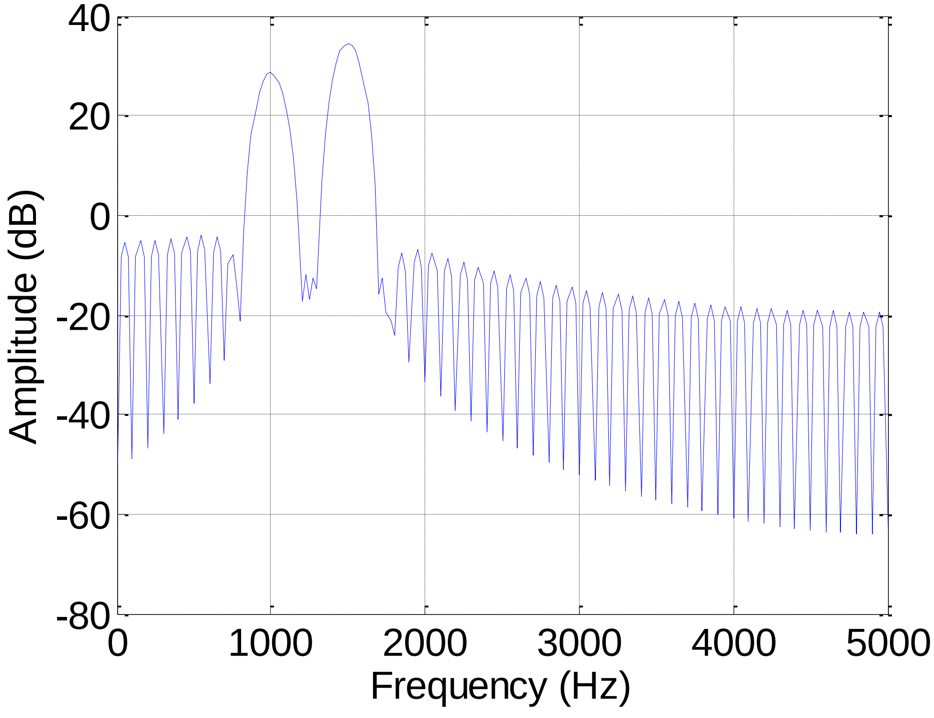
- Windowing is multiplication in time domain, so the spectrum will be a convolution between the signal's spectrum and the window's spectrum
- Convolution in time domain takes $O(N^2)$, but if we perform in the frequency domain…
  - FFT takes $O(N \log N)$
  - Multiplication takes $O(N)$
  - IFFT takes $O(N \log N)$

Windowed Signal

```
fs = 10000;      % sampling rate
f1 = 1000;       % fisrt sinusoid 1000Hz
f2 = 1500;       % second sinusoid 1500Hz
t = 0:1/fs:3;    % 3 seconds long
x1 = sin(2*pi*f1*t);     % first signal
x2 = 2*sin(2*pi*f2*t);   % second signal
x = x1+x2;               % mixture signal
L = 100;                 % window length
fftLen = L*4;            % fft length
w = hamming(L);          % window
wx = w'.*x(101:100+L);   % windowed signal
% magnitude spectrum of windowed signal
wxf = 20*log10(abs(fft(wx, fftLen)));
% show spectrum (only the positive frequencies)
figure; h = axes('FontSize', 16);
plot(h, (0:fftLen/2)*fs/fftLen, wxf(1:fftLen/2+1));
grid on;
xlabel('Frequency (Hz)');
ylabel('Amplitude (dB)');
```



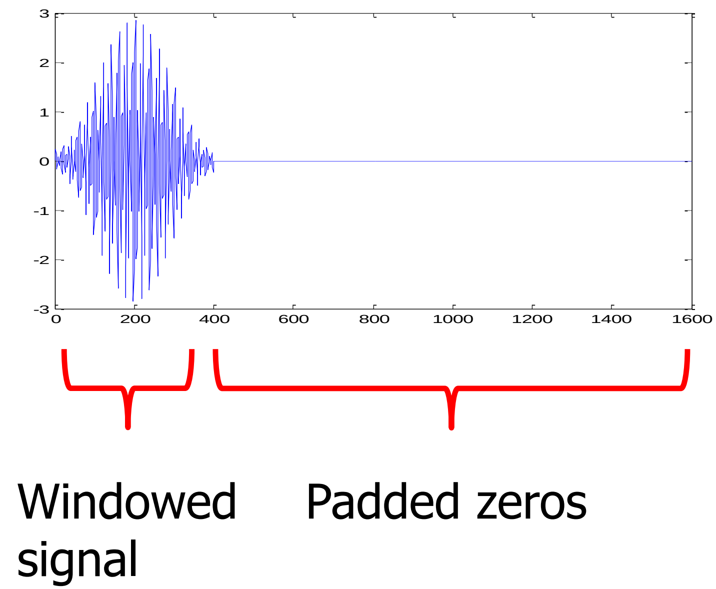
Spectrum of Windowed Signal



- Two sinusoids: 1000Hz + 1500Hz
- Sampling rate: 10KHz
- Window length: 100 (i.e. 100/10K = 0.01s)
- FFT length: 400 (i.e. 4 times zero padding)

Zero Padding

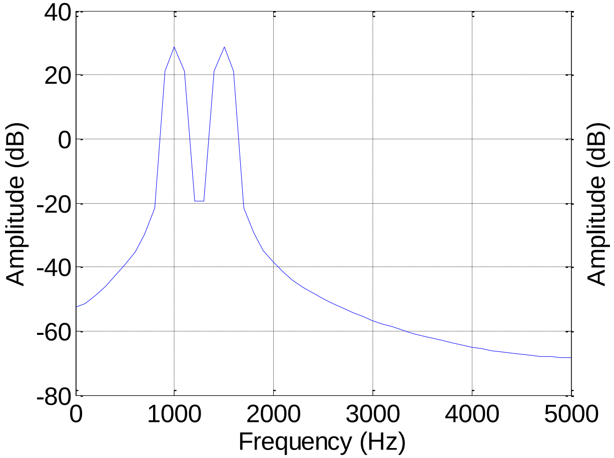
- Add zeros after (or before) the signal to make it longer
- Perform DFT on the padded signal



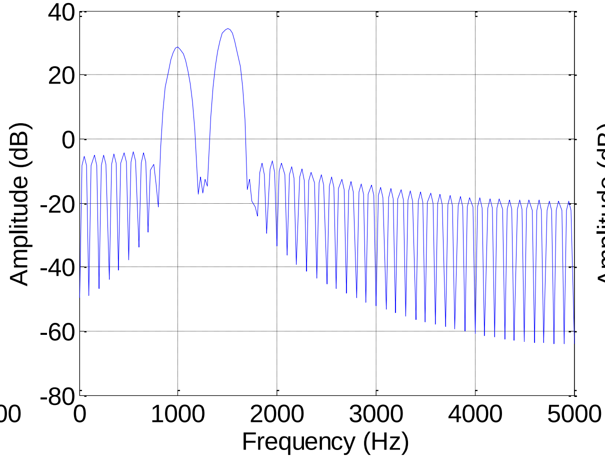
Why Zero Padding?

- Zero padding in time domain gives the ideal interpolation in the frequency domain.
- It doesn't increase (the real) frequency resolution!
  - 4 times is generally enough
  - Here the resolution is always fs/L=100Hz

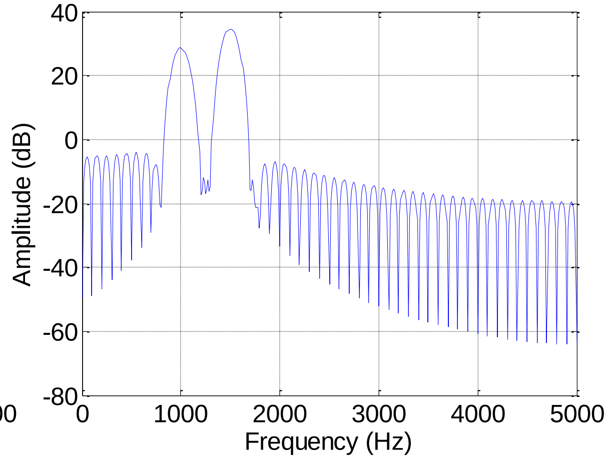
No zero padding



4 times zero padding



8 times zero padding



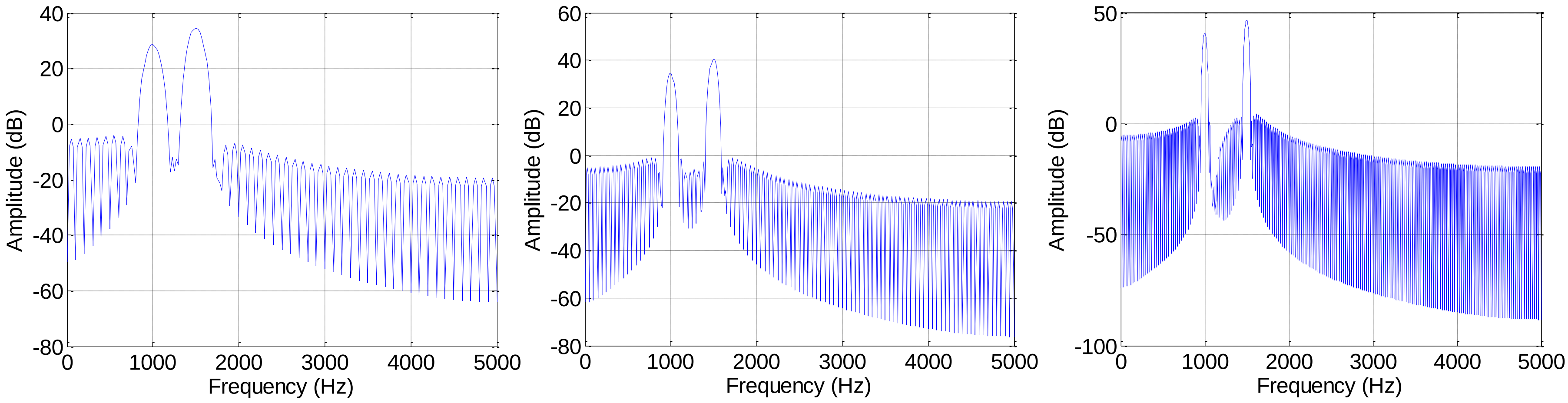
How to increase frequency resolution?

- Time-frequency resolution tradeoff

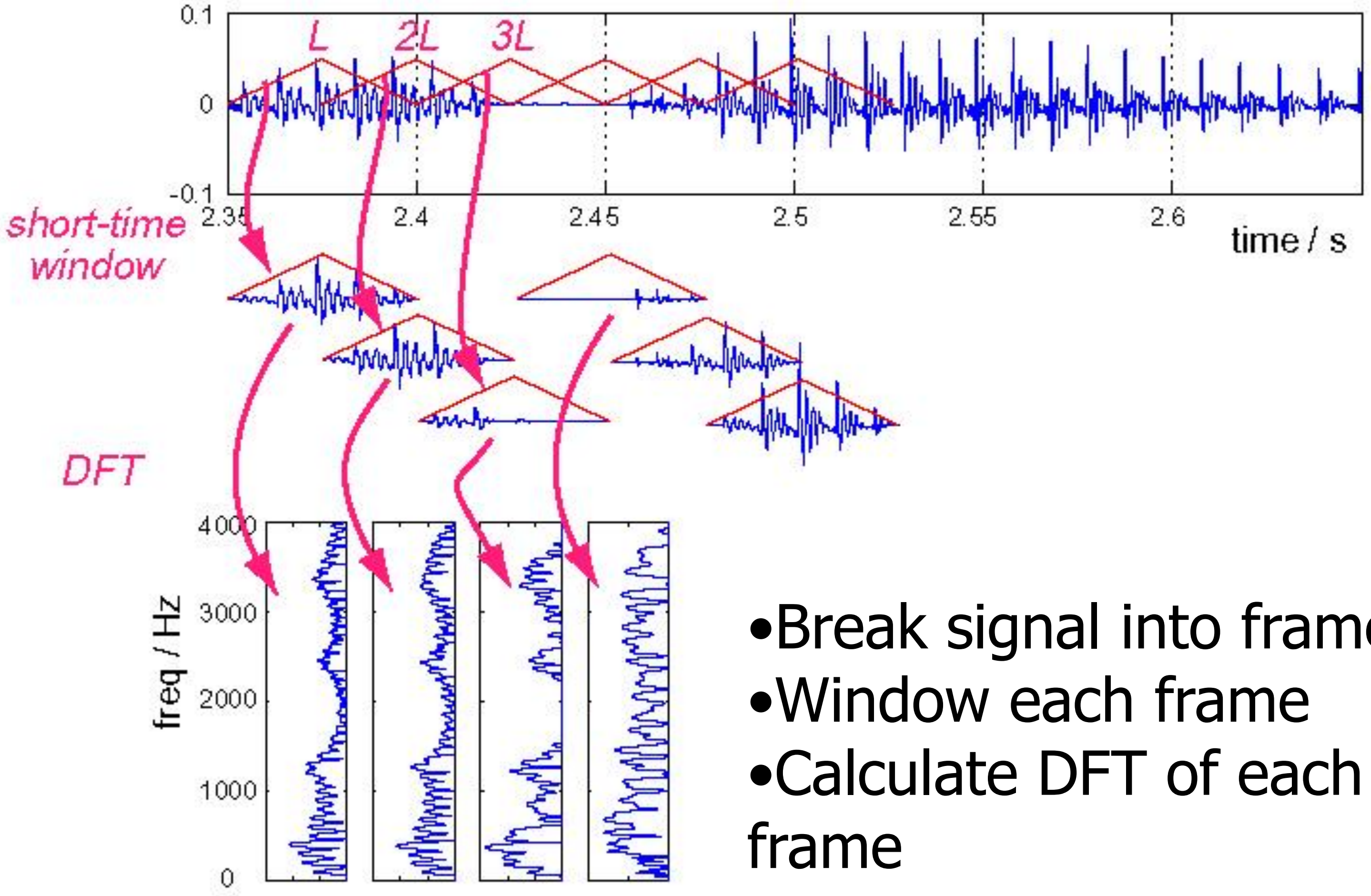
$$\Delta t \cdot \Delta f = 1$$

(second) (Hz)

Window length: 10ms Window length: 20ms Window length: 40ms

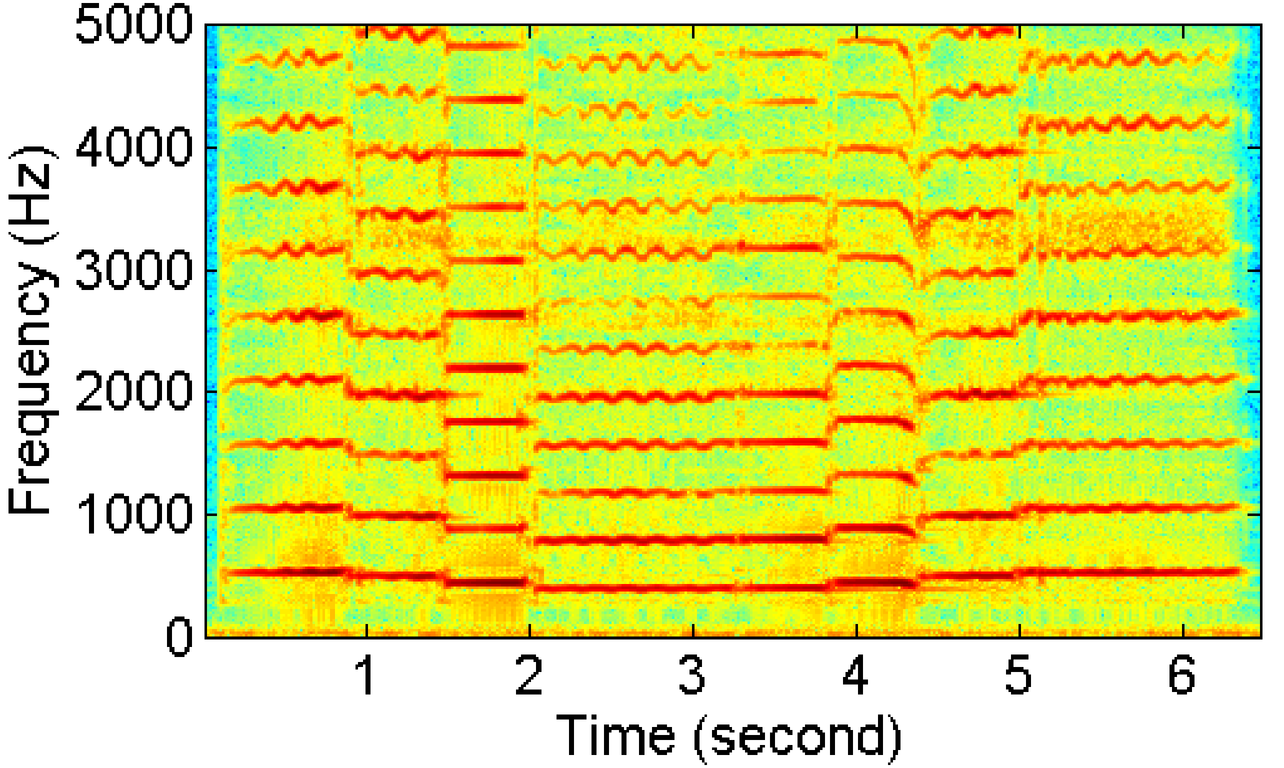


Short time Fourier Transform



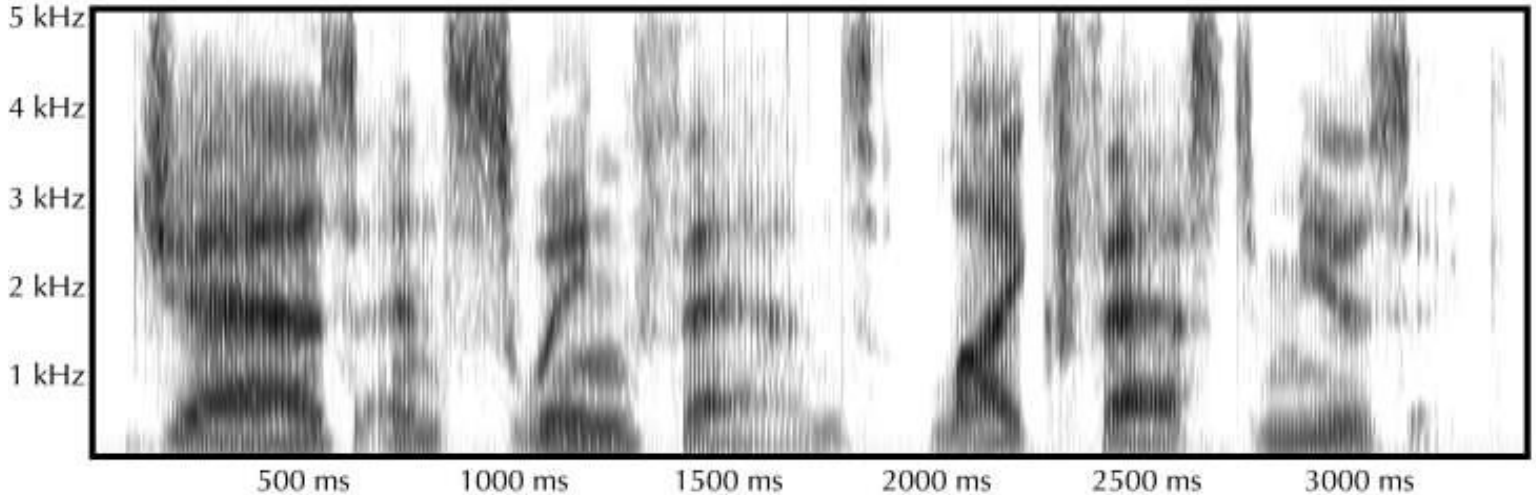
- Break signal into frames
- Window each frame
- Calculate DFT of each windowed frame

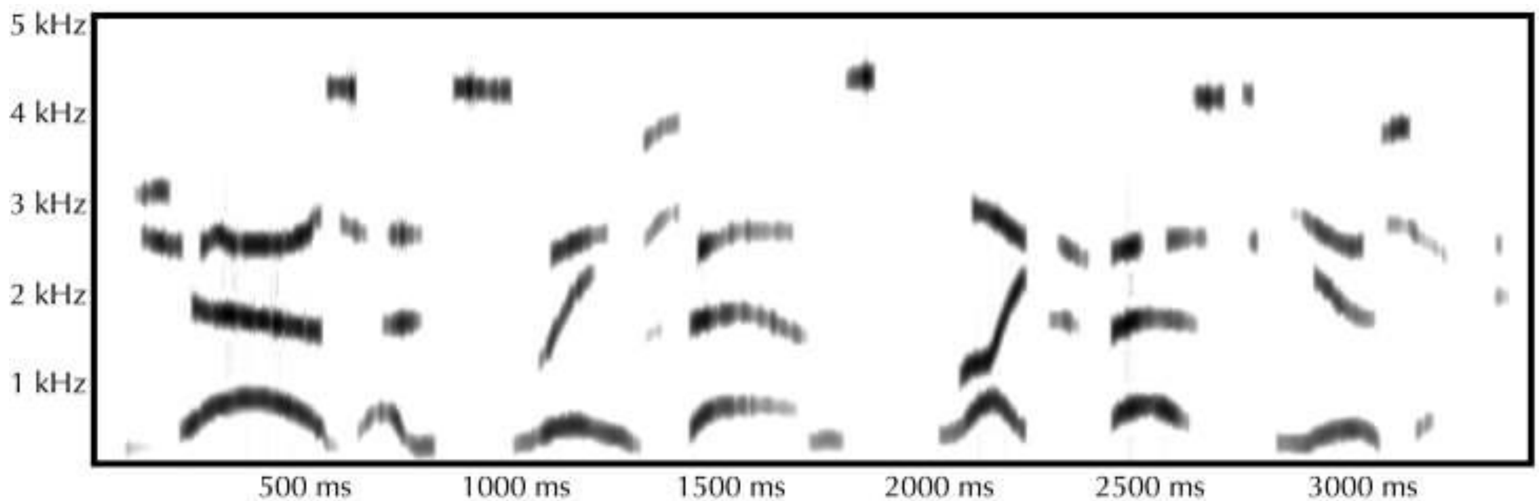
The Spectrogram



- There is a "spectrogram" function in matlab.

A Fun Example





(Thanks to Robert Remez)

Overlap-Add Synthesis

- IDFT on each spectrum.
  - Use the complex, full spectrum.
  - Don’t forget the phase (often using the original phase).
  - If you do it right, the time signal you get is real.

- (optional) Multiply with a synthesis window (e.g., Hamming) to suppress signals at edges.
  - Not dividing the analysis window

- Overlap and add different frames together.

Constant Overlap Add (COLA)

- Windows of all frames add up to a constant function. Perfect reconstruction!

$$\sum_{m} w[n - mR] = const$$

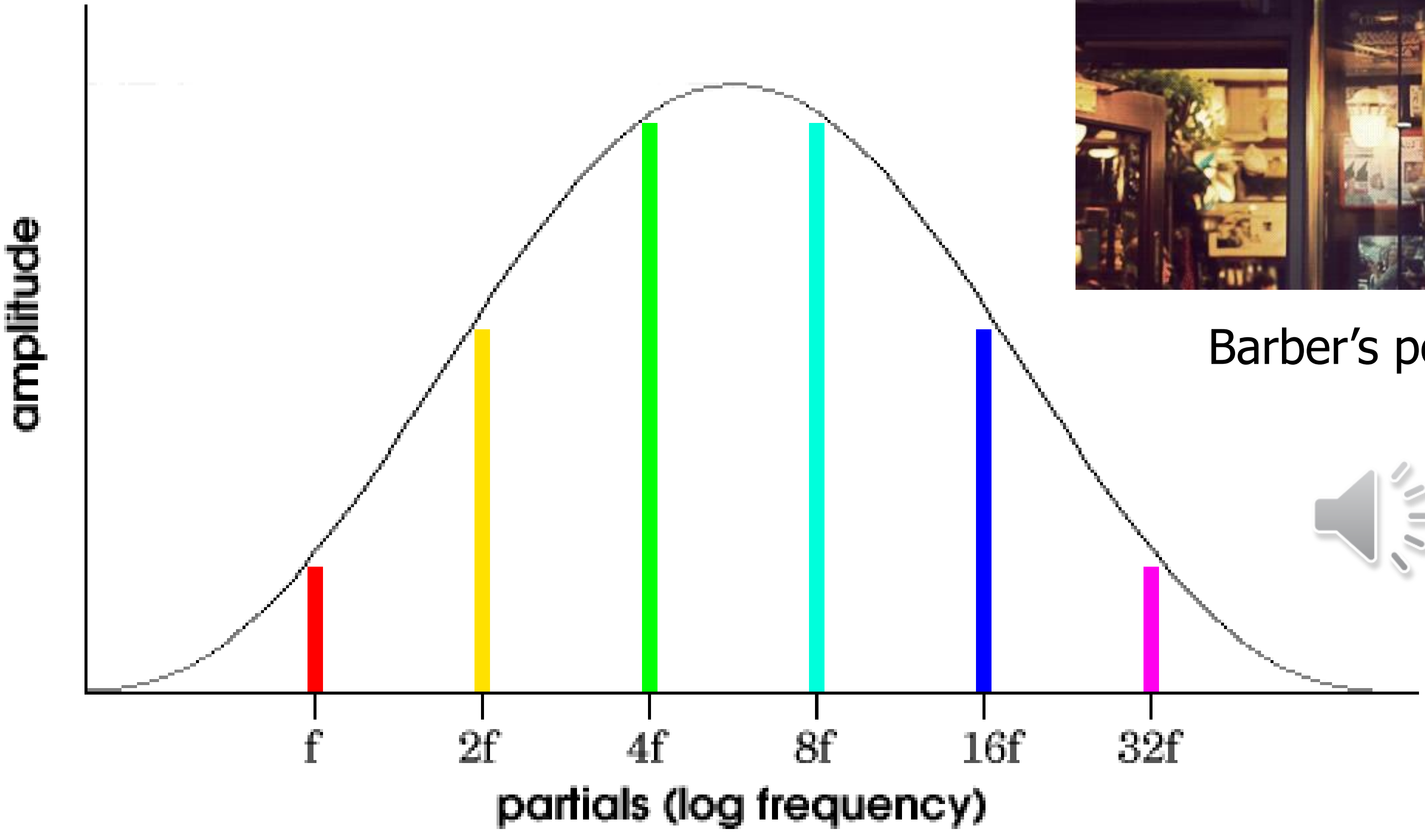
Frame index

Window function

Frame hop size

- Requires special design of w and R
  - Rectangular window: $R \leq L$ ← Window size
  - Triangular window: $R = \frac{L}{k}, k \geq 2, k \in \mathbb{N}$
  - Hamming/hann window: $R = \frac{L}{2k}, \mathrm{k} \in \mathbb{N}$

Shepard Tones



Barber's pole

Continuous Risset scale

Shepard Tones

- Make a sound composed of sine waves spaced at octave intervals.
- Control their amplitudes by imposing a Gaussian (or something like it) filter in the log-frequency dimension.
- Move all the sine waves up a musical ½ step.
- Wrap around in frequency.