# APPLIED SIGNAL PROCESSING + MACHINE LEARNING 1

CSE 599 Prototyping Interactive Systems | Lecture 11 | May 6

**Jon Froehlich** • Jasper O'Leary (TA)

DESIGN USE BUILD

MAKEABILITY LAB

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

UNIVERSITY of WASHINGTON

# Project Pitches

Published Edit

Please read the Final Project assignment first then loop back to read the project pitch assignment.

*While this assignment is individual, you have the choice to work on your course project individually or with a partner.*

## Project Pitches

For your pitches, you must submit:

- **A brainstorm sheet or sheets** (can be scanned from paper or born digitally) enumerating at least 10 different project ideas (at least a sentence or two per idea of explanation)

- You will also **downselect** to your top **two favorite ideas.** For those two ideas, we would like you to write 1-2 paragraphs explaining the idea (with sketches, if you'd like), why its interesting, how it fulfills the design prompt, and feasibility for completion in five weeks. Unlike the brainstorm sheet, we would like these paragraphs written digitally

Jasper and I will review all of the downselected ideas and help you reflect on possible pursuits. We will also spend in-class time to share ideas and form teams (as necessary).

# WHAT MIGHT ML HELP US DO IN HCI/UBICOMP SYSTEMS?

**Classify** low- or high-level activities (*e.g.*, gestures, playing baseball)

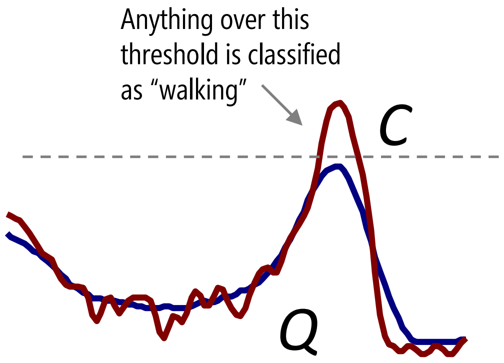
**Cluster** similar signals together (*e.g.*, how many categories of things exist in this dataset)

**Search**. (*e.g.*, I have signal A & I want to find all other signals like this in my data)

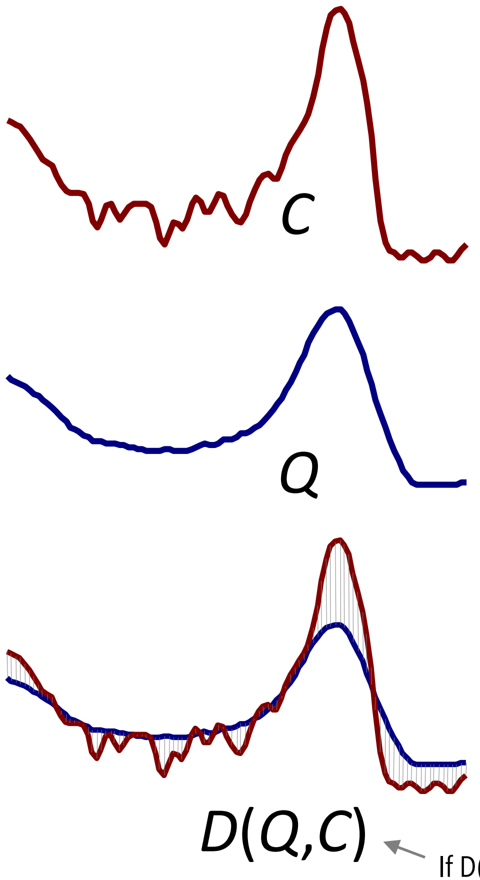
**Novelty detection** (*i.e.*, anomaly detection)

# 3 PREVAILING APPROACHES FOR SIGNAL CLASSIFICATION

## 1. Rule-Based

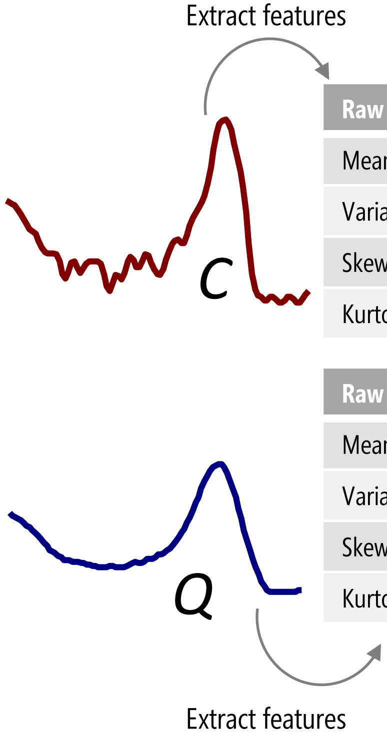



## 2. Shape-Matching




If D(Q,C) < threshold, we consider them the same signal

## 3. Feature-Based




| Raw Signal | 1st Derivative | FFT |
|---|---|---|
| Mean | Mean | Top N freqs |
| Variance | Variance | Fundamental |
| Skewness | Skewness | |
| Kurtosis | Kurtosis | |

| Raw Signal | 1st Derivative | FFT |
|---|---|---|
| Mean | Mean | Top N freqs |
| Variance | Variance | Fundamental |
| Skewness | Skewness | |
| Kurtosis | Kurtosis | |

Supervised Learning Model

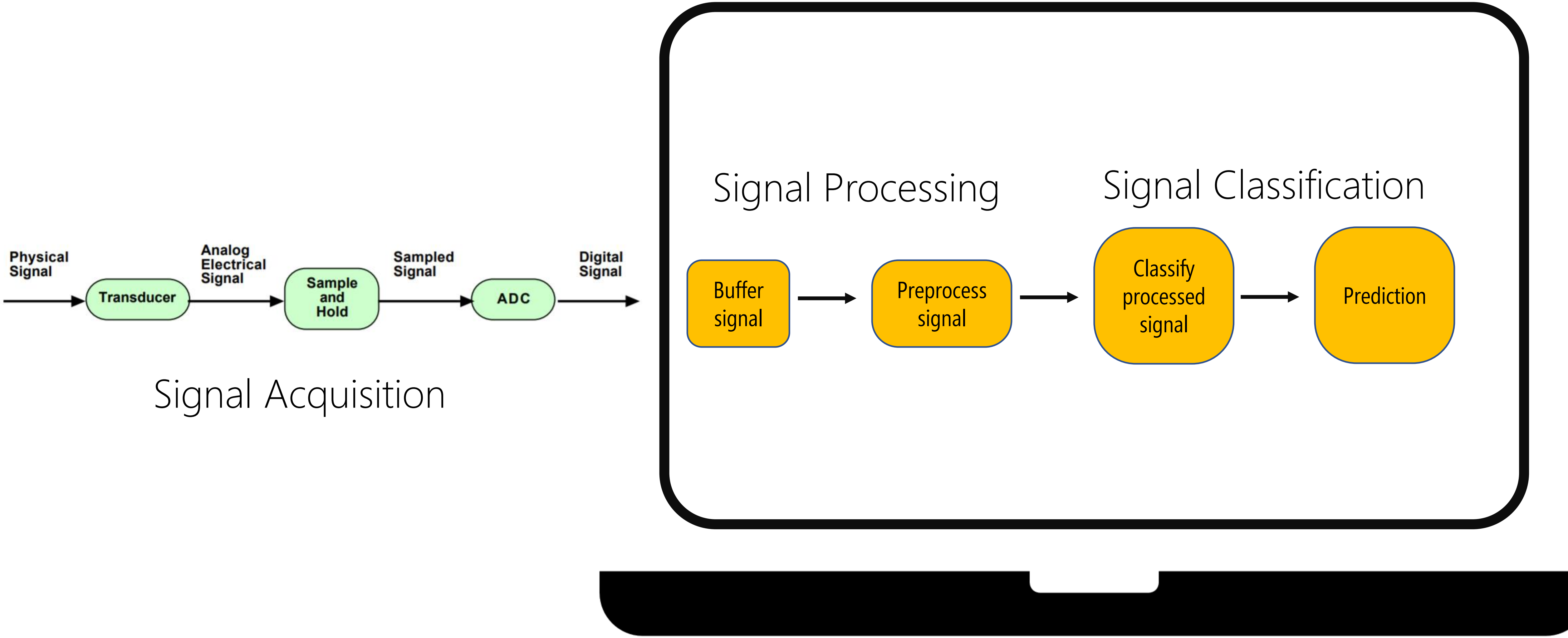
Output

# A3 GESTURE REC USING SHAPE MATCHING

```
Shape Matching Accuracy (All Accel Signals): 55/55 (100.0%)

Took 2.408s for 55 matches (avg=0.044s per match)
- Midair Zorro 'Z' 5/5 (100.0%)
- Baseball Throw 5/5 (100.0%)
- Midair 'S' 5/5 (100.0%)
- Bunny Hops 5/5 (100.0%)
- Forehand Tennis 5/5 (100.0%)
- Midair Counter-clockwise 'O' 5/5 (100.0%)
- At Rest 5/5 (100.0%)
- Midair Clockwise 'O' 5/5 (100.0%)
- Backhand Tennis 5/5 (100.0%)
- Shake 5/5 (100.0%)
- Underhand Bowling 5/5 (100.0%)
```

# DATA FLOW FROM PHYSICAL SIGNAL TO PREDICTION




Based on: Stiber, Stiber, & Larson, Signal Computing, 2016 ([link](link))

# LEARNING GOALS

How do we **acquire sensor data** (*i.e.*, signal acquisition)?

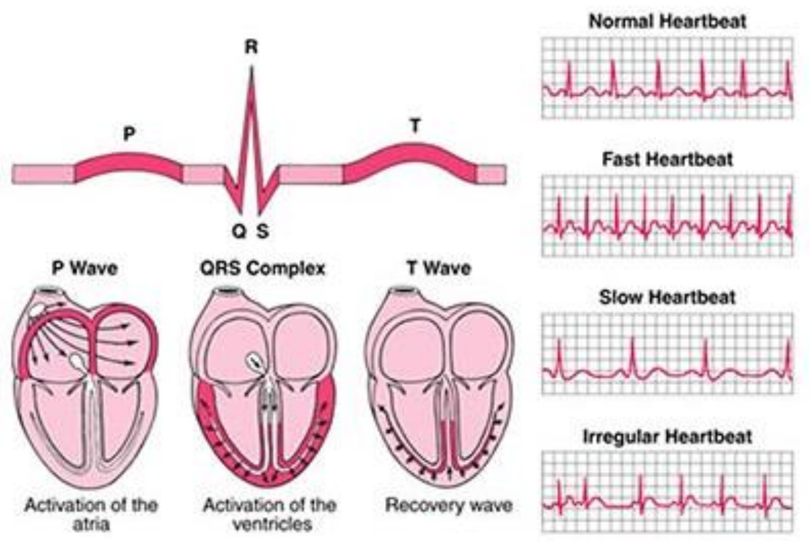
What is a time-series **signal** and how do we **represent them**?

Decomposing and synthesizing **signals**

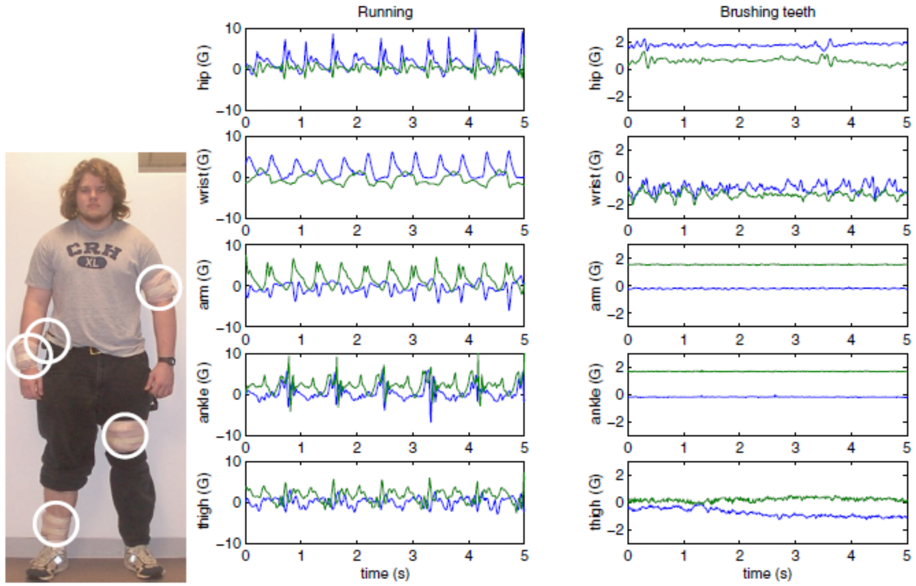
How to approach analyzing and processing **signals**

# TIME-SERIES SIGNALS

Sensors *sense* physical phenomena and convert them to electrical signals, which are digitized and represented as time-series arrays

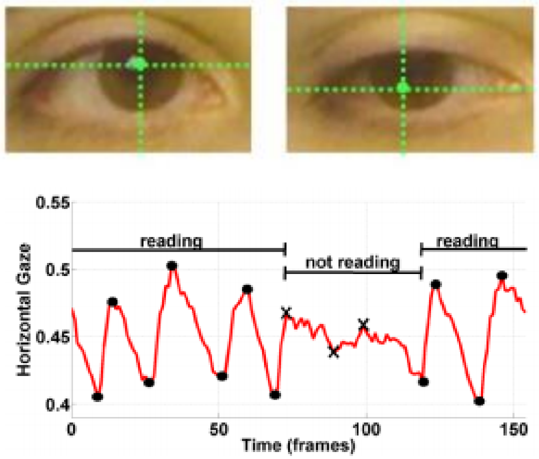



**Electrocardiograms for Heartbeat Tracking**




Source: Bao & Intille, Pervasive 2004

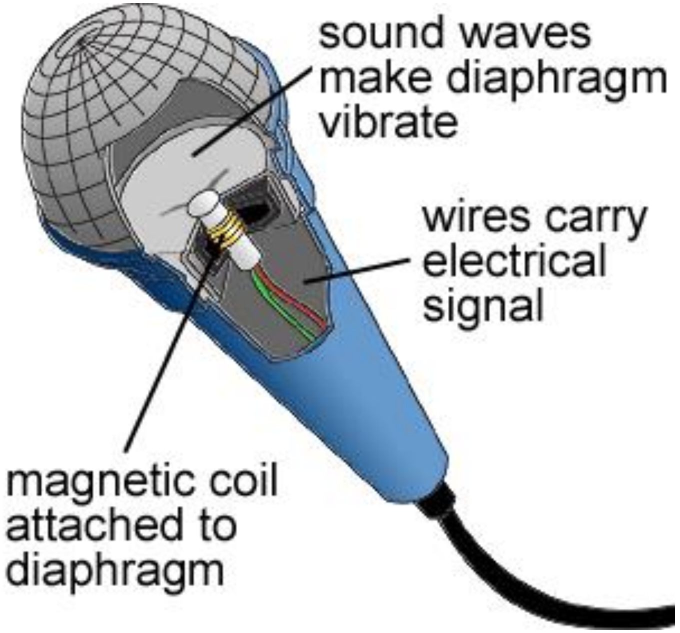
**Accelerometers for Activity Recognition**

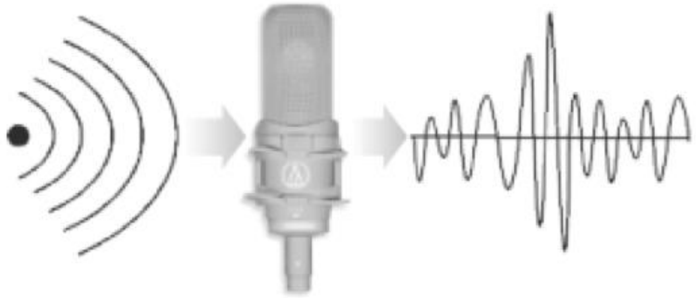



Source: Mariakakis *et a.,* CHI'15

**Smartphone Camera for Gaze Tracking**

# EXAMPLE: A MICROPHONE



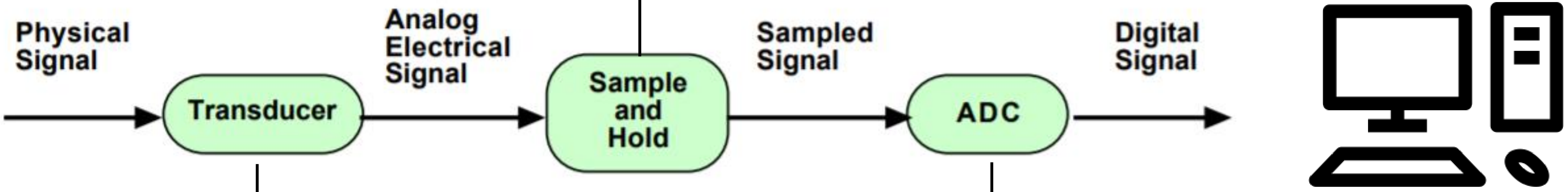



1. **Physical stimulus.** When you speak, play an instrument, *etc.*, it vibrates air in a sinusoidal pattern
2. **Diaphragm vibration.** Inside a microphone is a small diaphragm that vibrates in response to air movement
3. **Electricity generation.** A conductive coil, which is attached to the diaphragm moves back and forth across a magnet. This generates a current proportional to the sound wave.

# SIGNAL ACQUISITION

A sample and hold device **samples** the analog signal (as a voltage) and **holds** it at a fixed level.

Once it's on the computer (or smartphone), it's time to play using **digital signal processing** and **machine learning!**

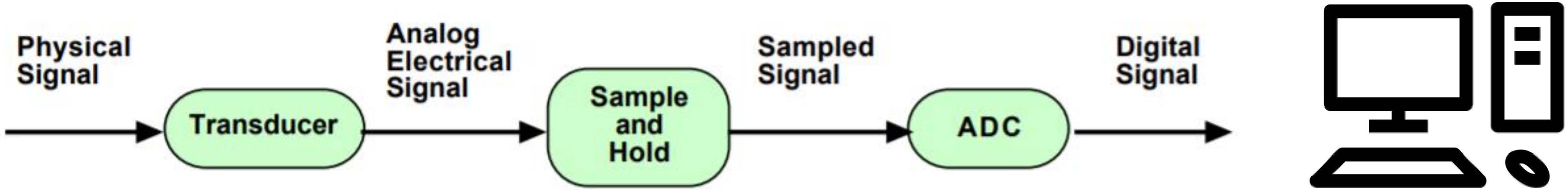



Converts physical signal into an analog electrical signal (*e.g.,* always has some noise due to environmental conditions like humidity, low light, etc.)

The **analog-to-digital** converter converts the fixed voltage (the sampled signal) to a binary number (called quantization)

Source: Stiber, Stiber, & Larson, Signal Computing, 2016 ([link](link))

# SIGNAL ACQUISITION: TWO KEY QUESTIONS

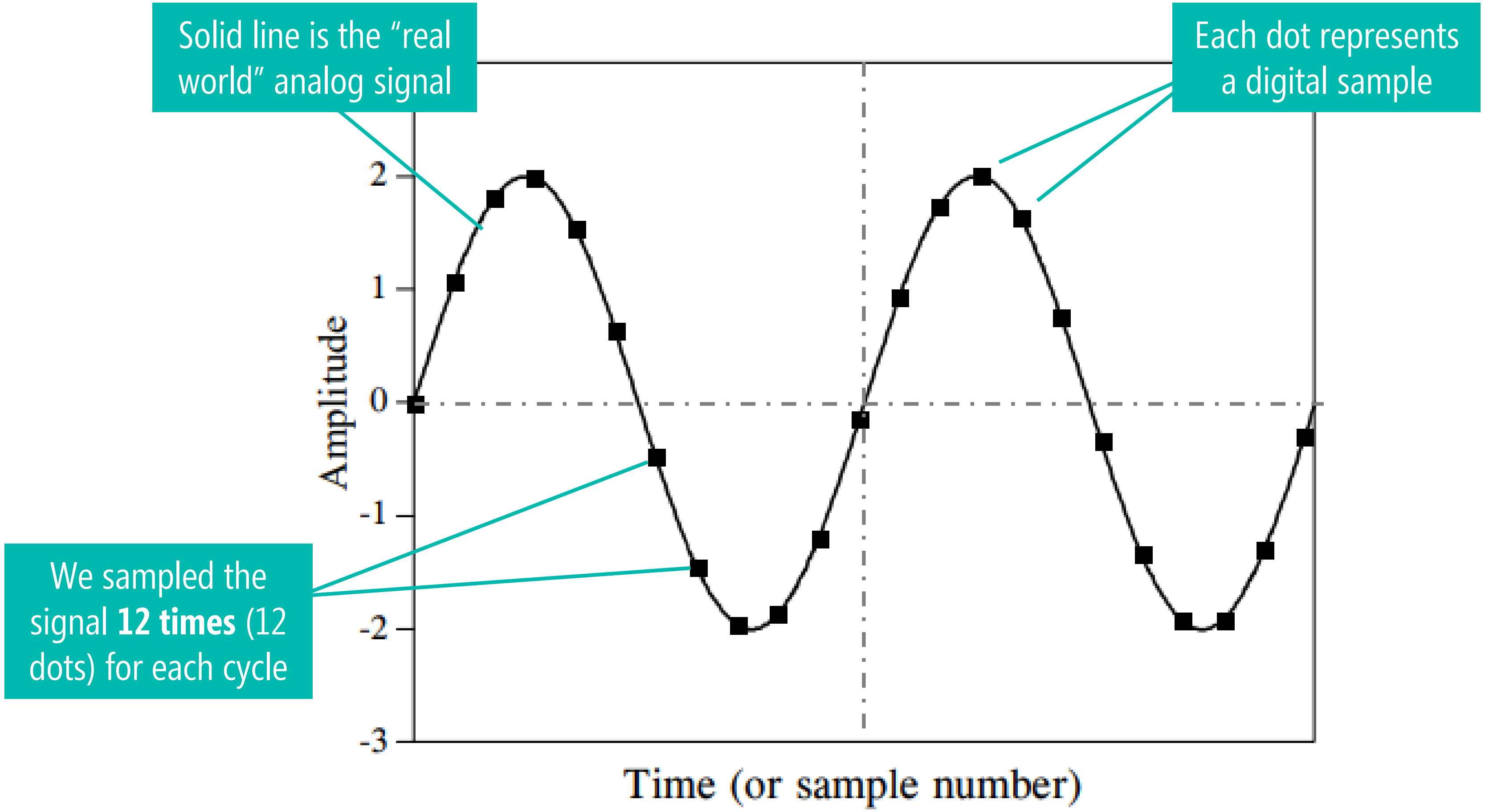



1. **Sampling rate:** How often do I need to sample the "real world?"
2. **Quantization**: How many bits should I use to represent each sample?

Source: Stiber, Stiber, & Larson, Signal Computing, 2016

# SAMPLING RATE

The rate that you sample the "real world." You should sample at a rate sufficient to accurately reconstruct the real analog signal.



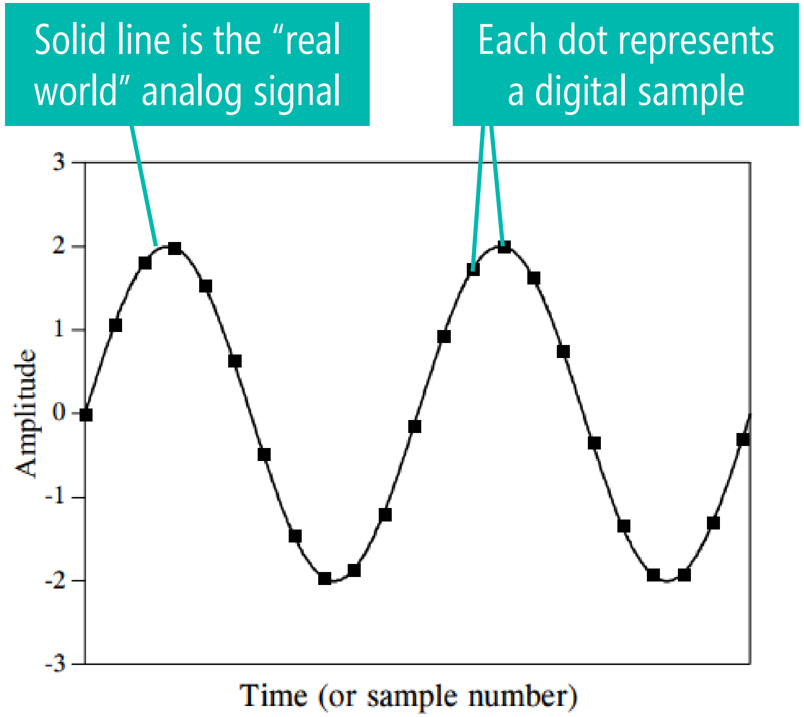

**Real-world signal:** 1 Hz
**Sampling frequency:** 12 Hz
**Ratio:** 12:1 ≈ 12

Source: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link)

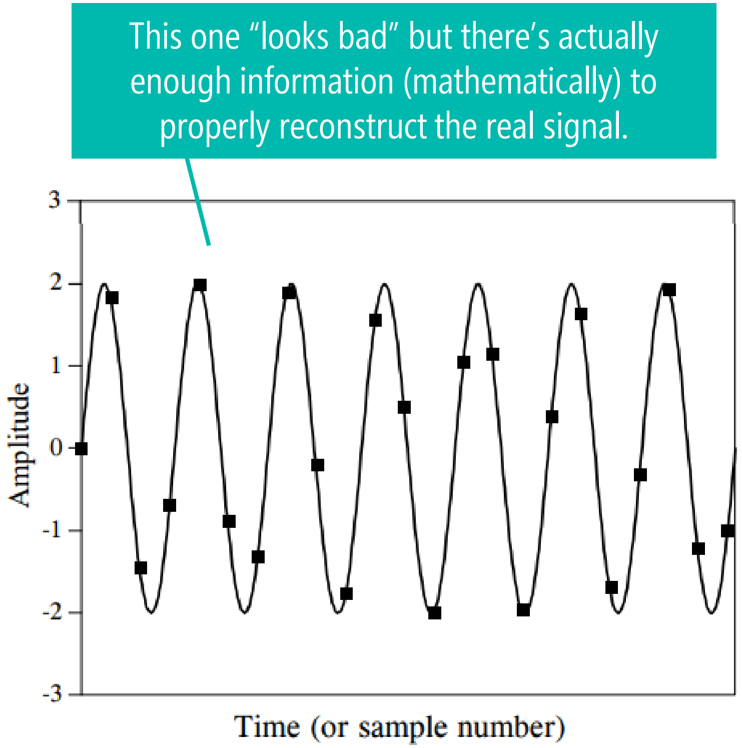
## SIGNAL ACQUISITION

# SAMPLING RATE

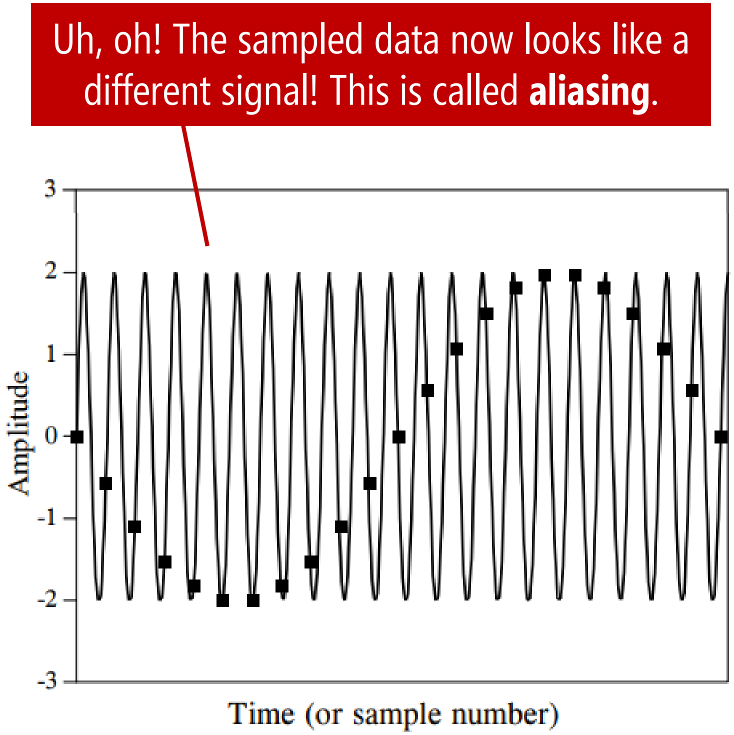
The rate that you sample the "real world." You should sample at a rate sufficient to accurately reconstruct the real analog signal.




**Real-world signal:** 1 Hz
**Sampling frequency:** 12 Hz
**Ratio:** 12:1 ≈ 12




**Real-world signal:** 3.72 Hz
**Sampling frequency:** 12 Hz
**Ratio:** 12:3.72 ≈ 3.22




**Real-world signal:** 11.4 Hz
**Sampling frequency:** 12 Hz
**Ratio:** 12:11.4 ≈ 1.05

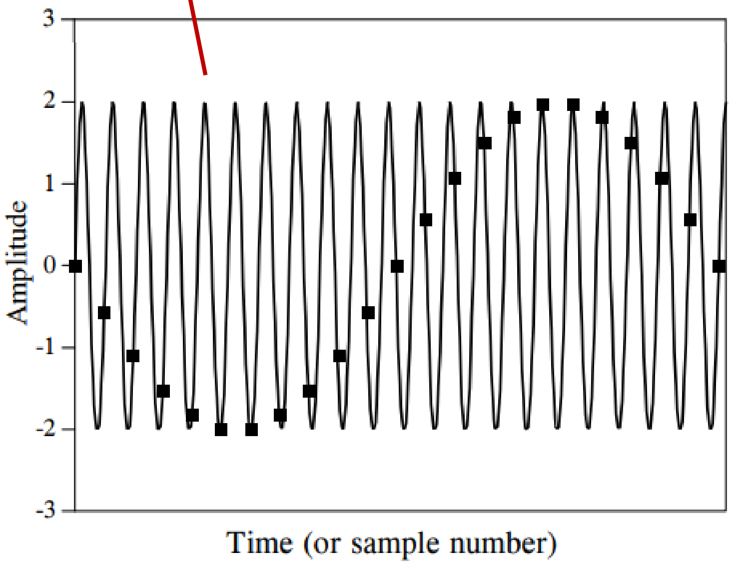
Source: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link)

SIGNAL ACQUISITION

# SAMPLING THEOREM

Frequently called the *Shannon* sampling theorem or the *Nyquist* sampling theorem

Uh, oh! The sampled data now looks like a different signal! This is called **aliasing**.




**Real-world signal:** 11.4 Hz
**Sampling frequency:** 12 Hz
**Ratio:** 12:11.4 ≈ 1.05

The sampling theorem states that a continuous signal can only be properly sampled *if it does not contain frequency components above one-half of the sampling rate.*

$$\underbrace{Sampling\ rate\ must\ be}_{} > 2 * \max(real\ world\ signal\ frequency)$$

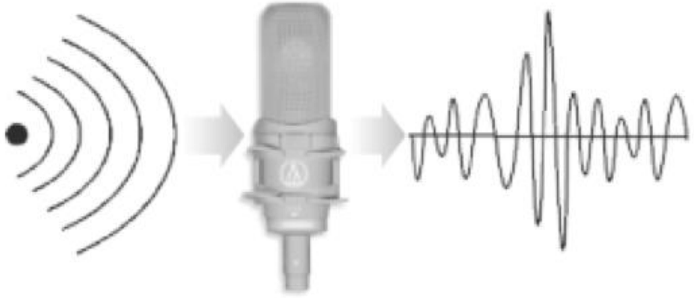
The **Nyquist frequency** is the minimum sampling rate you can in order to properly represent the signal

Source: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link)

# EXAMPLE: RECORDING SOUND

8,000 Hz

440 Hz



**Sensor (Microphone)**
**Sampling frequency:** 4,000 Hz

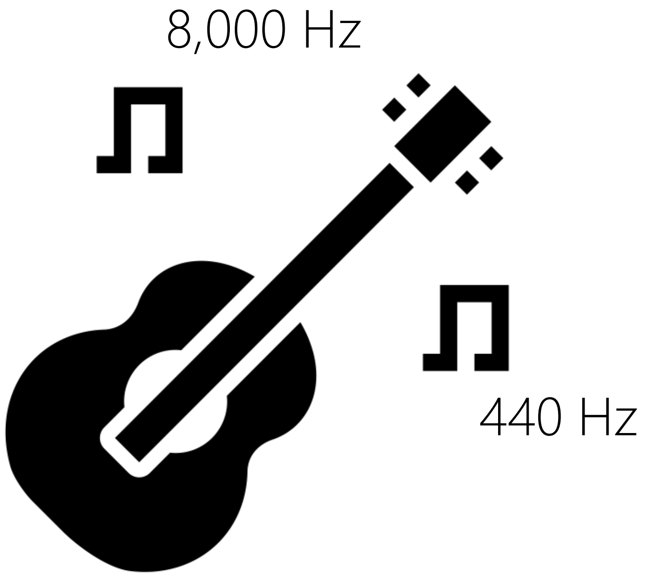
**What will happen here?**

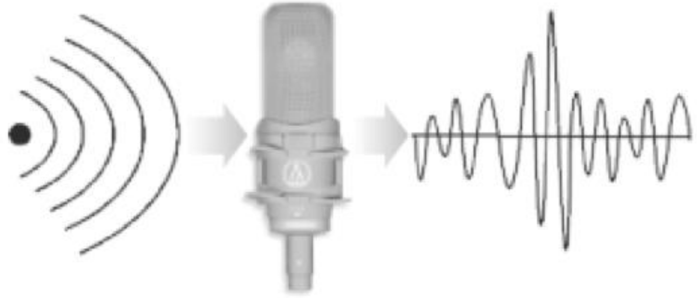
The guitar is producing sounds at a range of frequencies from 100Hz to above 8,000Hz; however, we are only recording our music at 4,000Hz.

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 ([link](link))

# EXAMPLE: RECORDING SOUND

When recording audio, if you play frequencies that are above ½ the sampling rate, you will get aliasing!



**Sensor (Microphone)**
**Sampling frequency:** 4,000 Hz

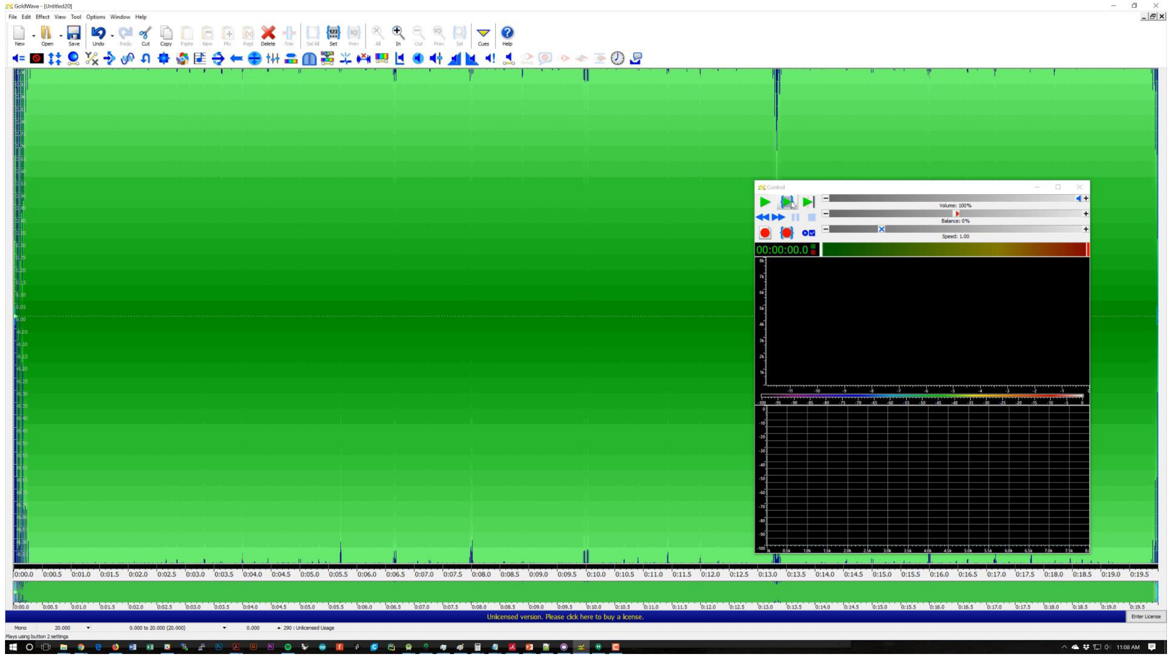
A **sampling rate of 4,000Hz** requires that any analog signal (in this case, sound) be composed of **frequencies 2,000** and below.

Any **frequencies** present in the signal **above this limit** will be **aliased** to frequencies between 0-2,000 Hz and will be combined with whatever information was legitimately there.

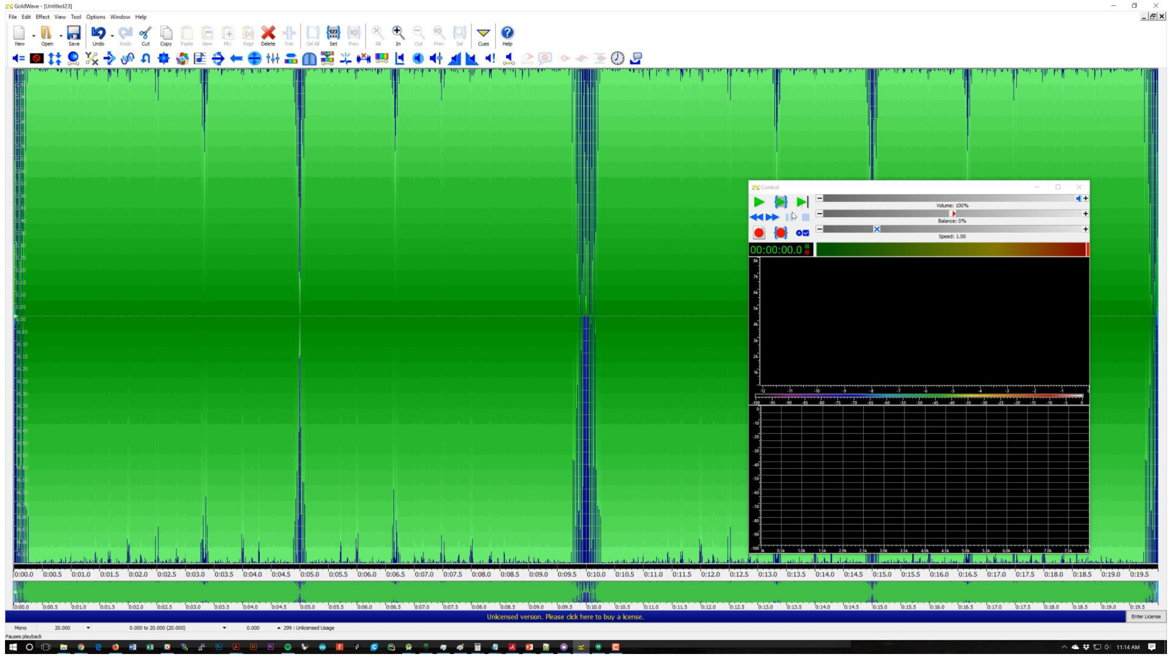
See: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link)

# ALIASING EXAMPLE: RECORDING SOUND

When recording audio, your sampling rate must be 2x the largest frequency in your “real-world” signal

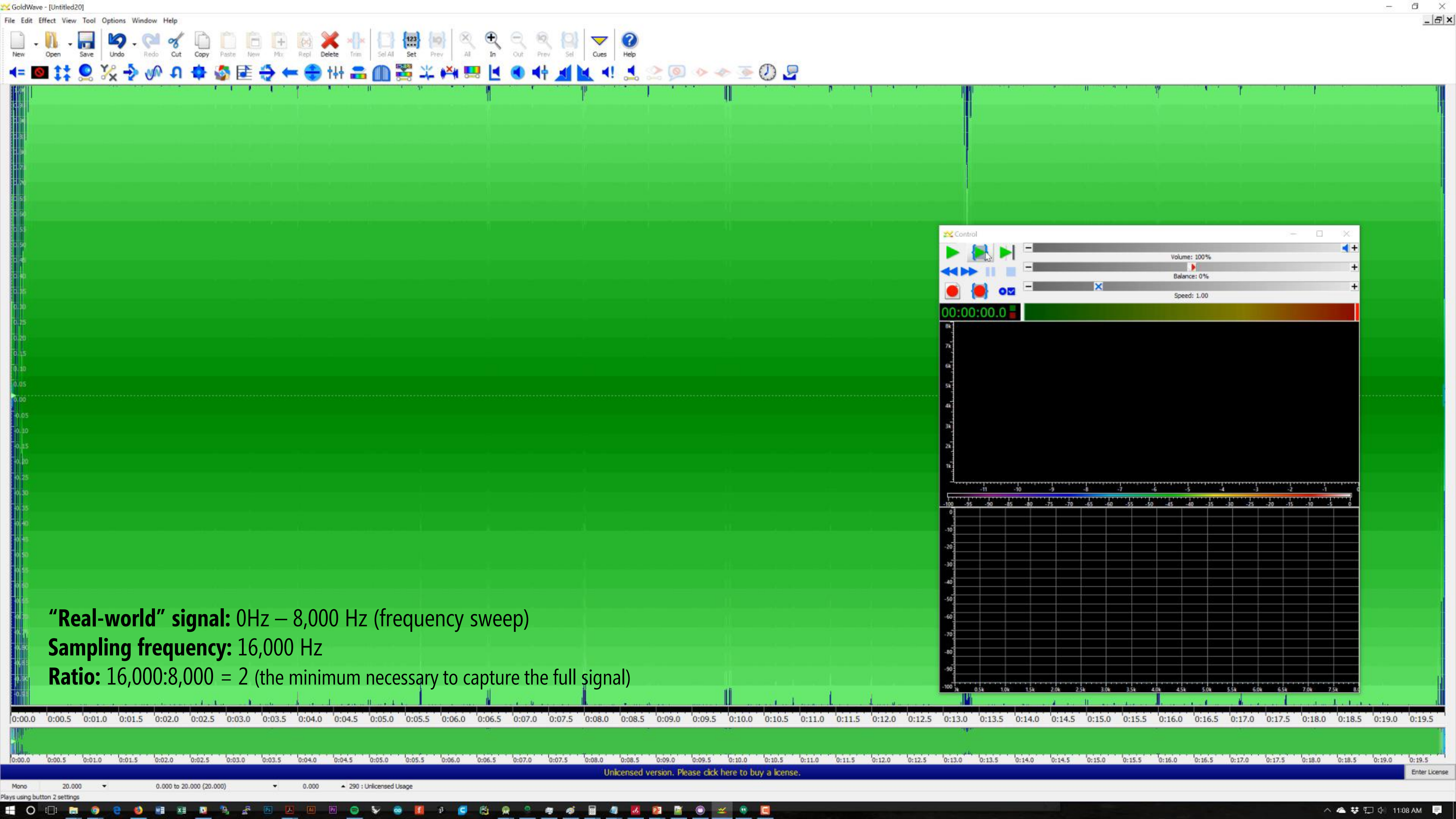


**“Real-world” signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 16,000 Hz
**Ratio:** 16,000:8,000 = 2 (the minimum necessary to capture the full signal)



**“Real-world” signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 4,000 Hz
**Ratio:** 4,000:8,000 = 0.5 (uh oh, ratio needs to 2x or more!)

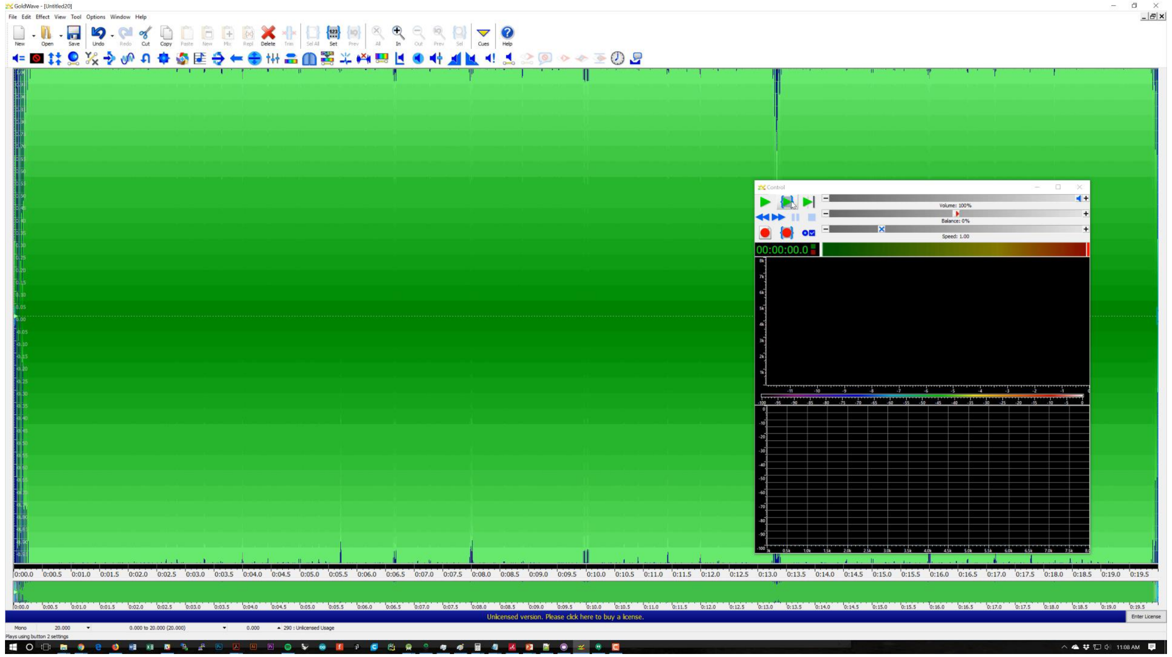
See: Smith, The Scientist and Engineer’s Guide to DSP, 1998 ([link](link))



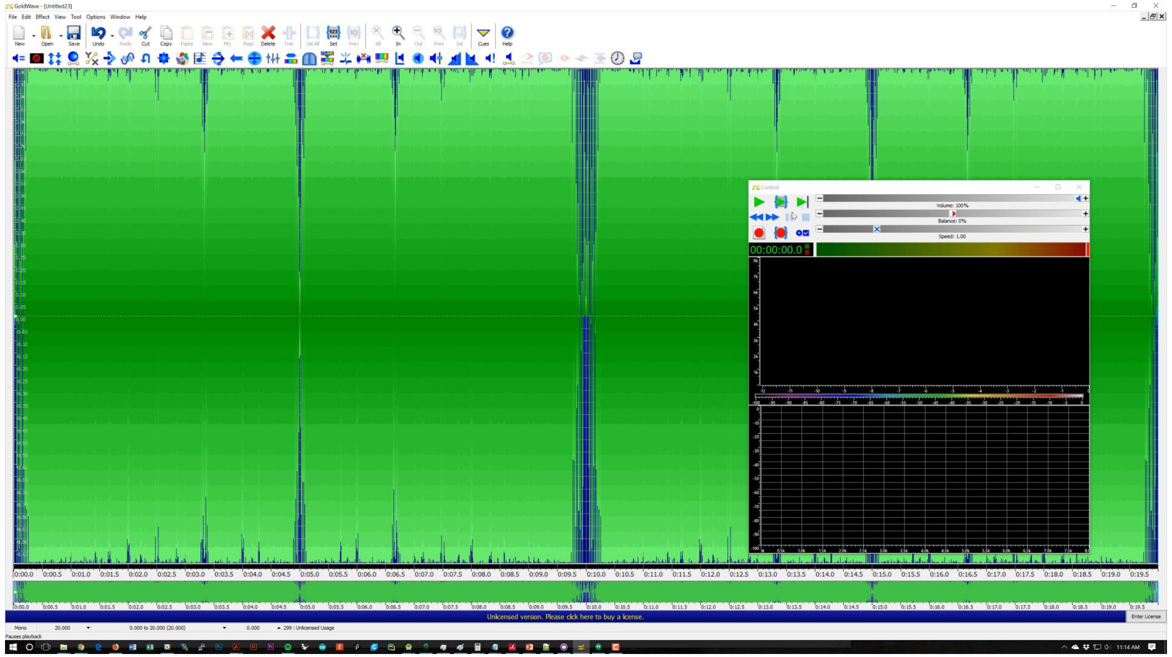

**"Real-world" signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 16,000 Hz
**Ratio:** 16,000:8,000 = 2 (the minimum necessary to capture the full signal)

# ALIASING EXAMPLE: RECORDING SOUND

When recording audio, your sampling rate must be 2x the largest frequency in your "real-world" signal

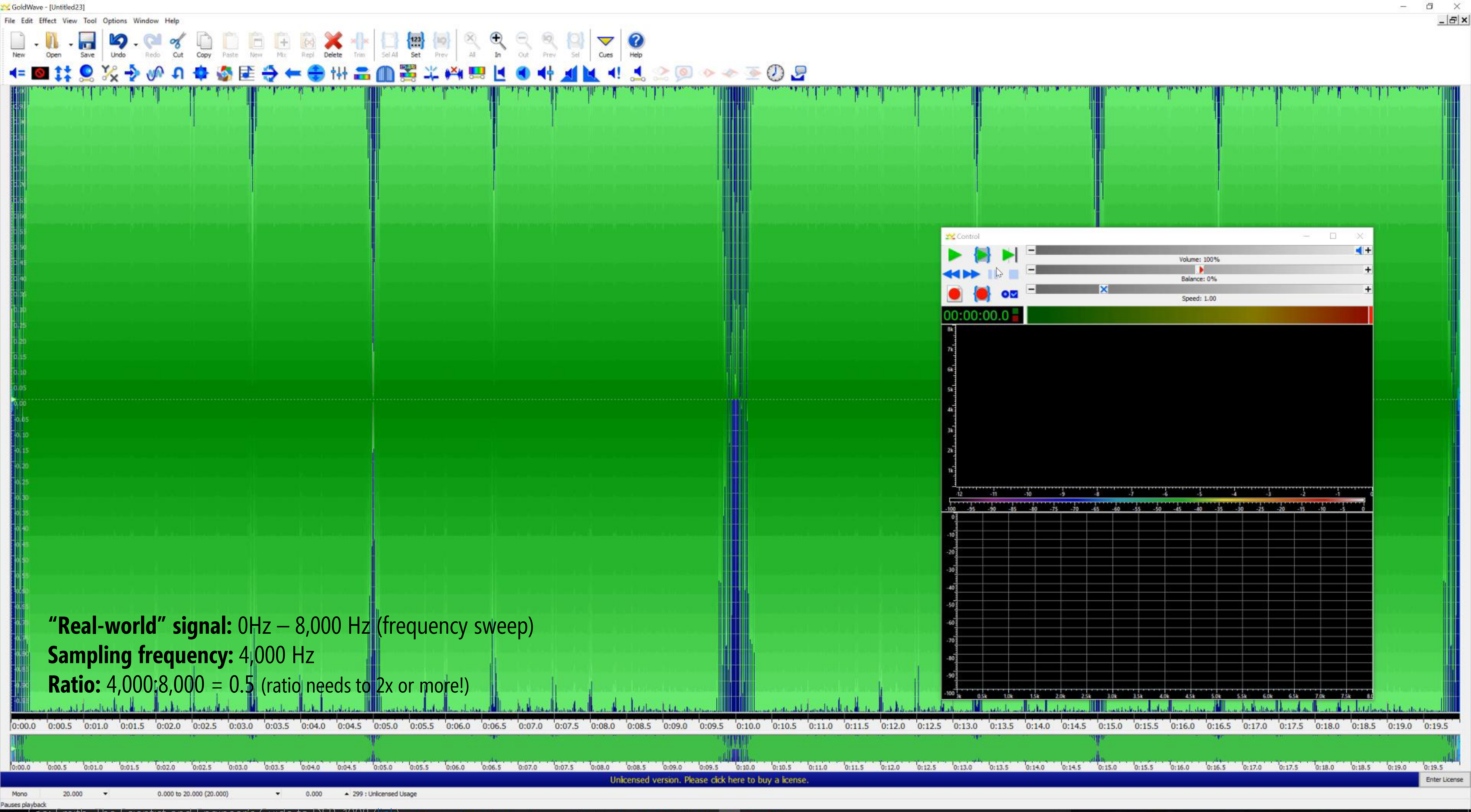


**"Real-world" signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 16,000 Hz
**Ratio:** 16,000:8,000 = 2 (the minimum necessary to capture the full signal)



**"Real-world" signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 4,000 Hz
**Ratio:** 4,000:8,000 = 0.5 (ratio needs to 2x or more!)

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 ([link](link))

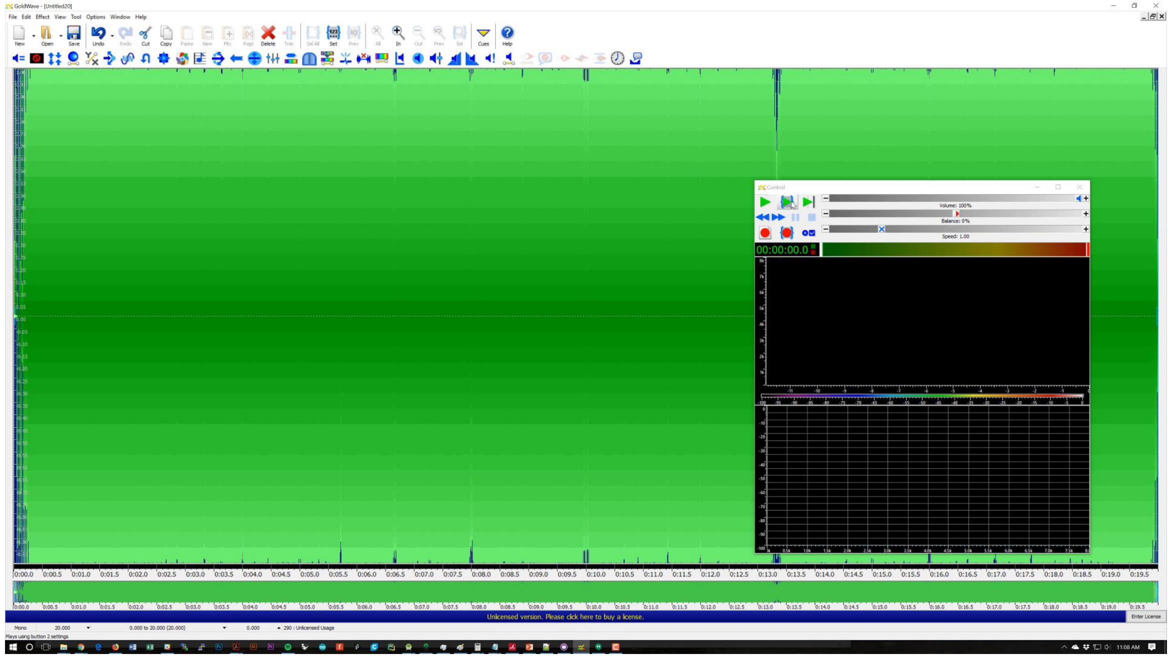



**"Real-world" signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 4,000 Hz
**Ratio:** 4,000:8,000 = 0.5 (ratio needs to 2x or more!)

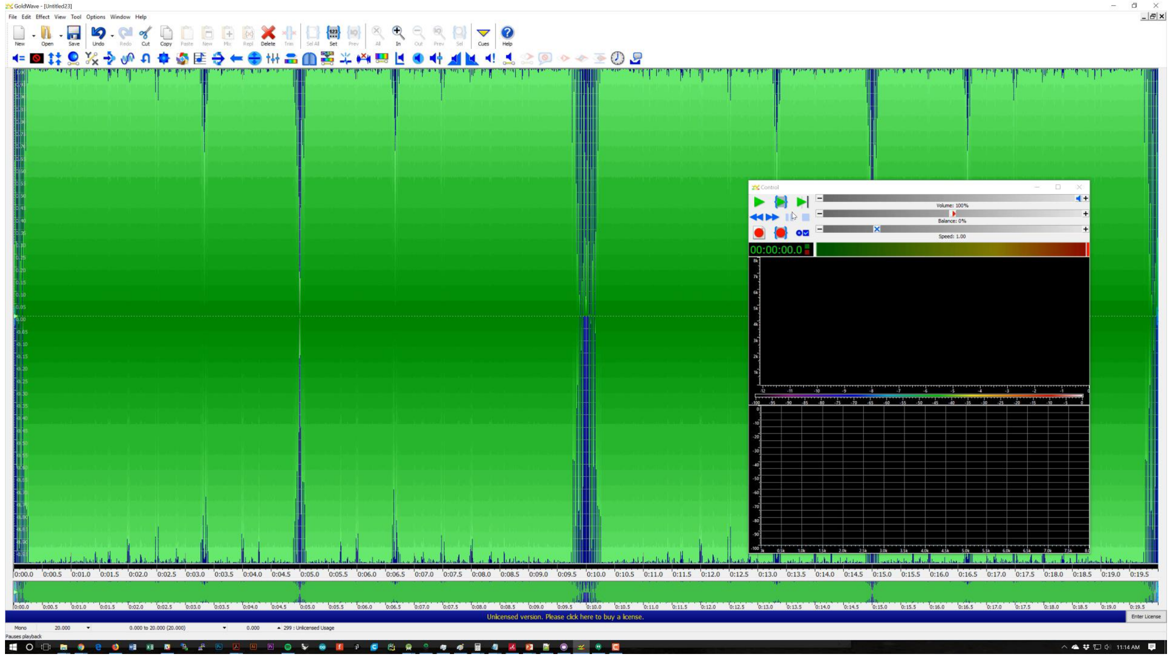
See: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link)

# ALIASING EXAMPLE: RECORDING SOUND

When recording audio, your sampling rate must be 2x the largest frequency in your "real-world" signal



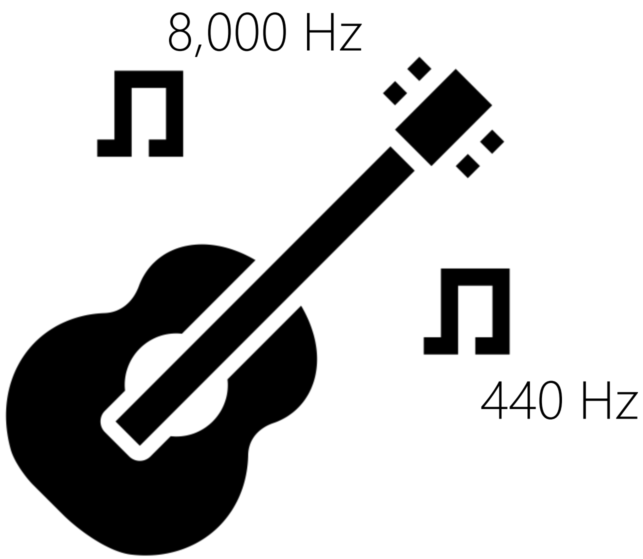
**"Real-world" signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 16,000 Hz
**Ratio:** 16,000:8,000 = 2 (the minimum necessary to capture the full signal)



**"Real-world" signal:** 0Hz – 8,000 Hz (frequency sweep)
**Sampling frequency:** 4,000 Hz
**Ratio:** 4,000:8,000 = 0.5 (ratio needs to 2x or more!)

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 ([link](link))

# ALIASING EXAMPLE: RECORDING SOUND




The range of human hearing is ~20Hz – 20kHz with far more sensitivity between 1-4kHz

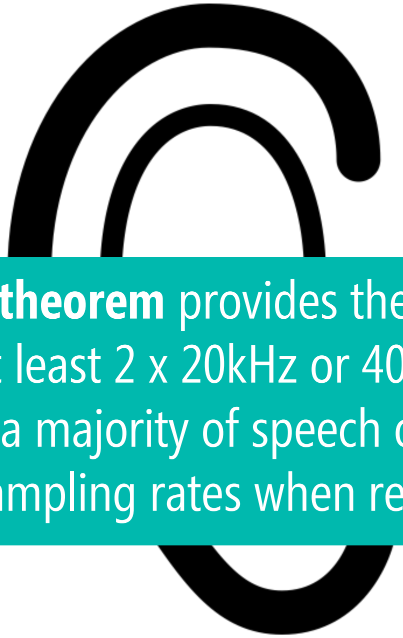

What sampling rate should we record music at then?

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link); http://www1.cs.columbia.edu/~hgs/audio/44.1.html

# ALIASING EXAMPLE: RECORDING SOUND

8,000 Hz

440




The **sampling theorem** provides the answer. The sampling rate needs to be at least 2 x 20kHz or 40kHz. CDs, for example, use 44.1kHz. Note: a majority of speech occurs < 8,000Hz so can use lower sampling rates when recording speech only.

**The range of human hearing is ~20Hz – 20kHz with far more sensitivity between 1-4kHz**

**What sampling rate should we record music at then?**

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link); http://www1.cs.columbia.edu/~hgs/audio/44.1.html

The G.711 telephony standard released in 1972 sampled at **8,000 Hz**!


See: G.711 telephony standard: https://en.wikipedia.org/wiki/G.711

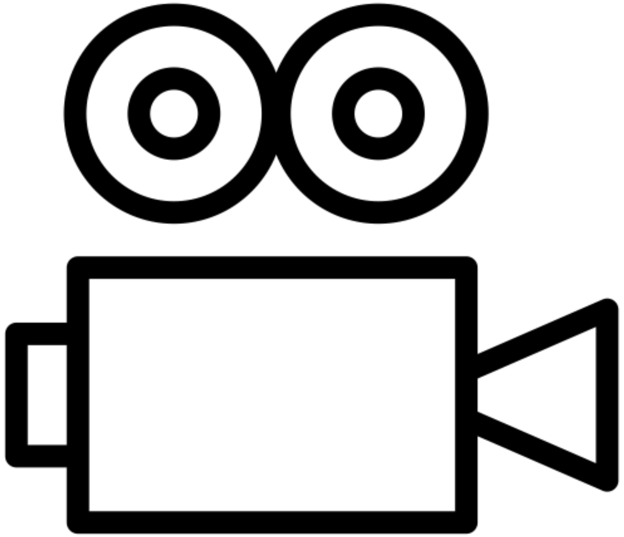
## SIGNAL ACQUISITION

# ALIASING EXAMPLE: VIDEO RECORDING MOVEMENT

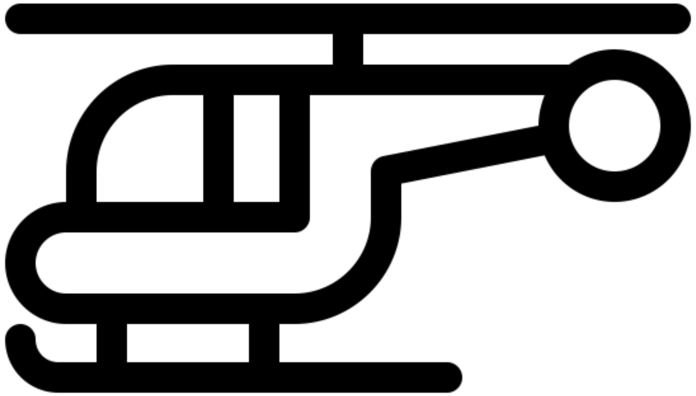
Source: https://youtu.be/yr3ngmRuGUc; See also: https://youtu.be/AYQAKwCxScc


The real-world rotational frequency of the blades
The video camera's sampling frequency, which causes us to see a much slower rotational frequency
3
2
1
0
-1
-2
-3
Amplitude
Time (or sample number)

# ALIASING EXAMPLE: CAMERAS



**Video cameras typically record at:**
24Hz – 60Hz (*i.e.,* 24 – 60 fps)



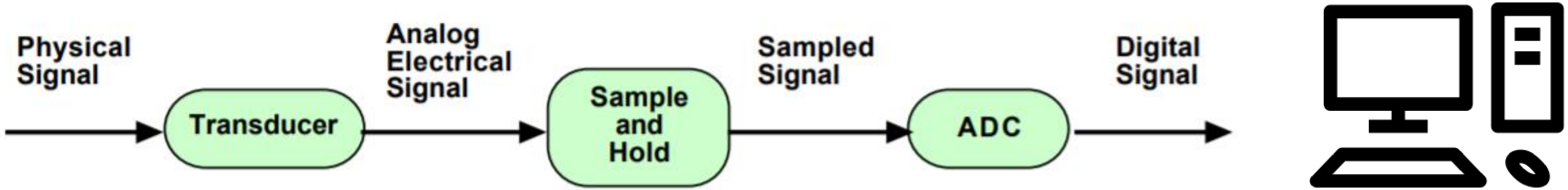
**How fast do helicopter blades spin?**
250 – 600 rpms

**How fast would we need to record video to capture 600 rpm blades?**

Minimum sampling rate
= 2 x 600 = 1,200 fps
(which would allow 2 captures per blade spin; more would be necessary for fluid video)

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link); http://www1.cs.columbia.edu/~hgs/audio/44.1.html

# SIGNAL ACQUISITION: TWO KEY QUESTIONS

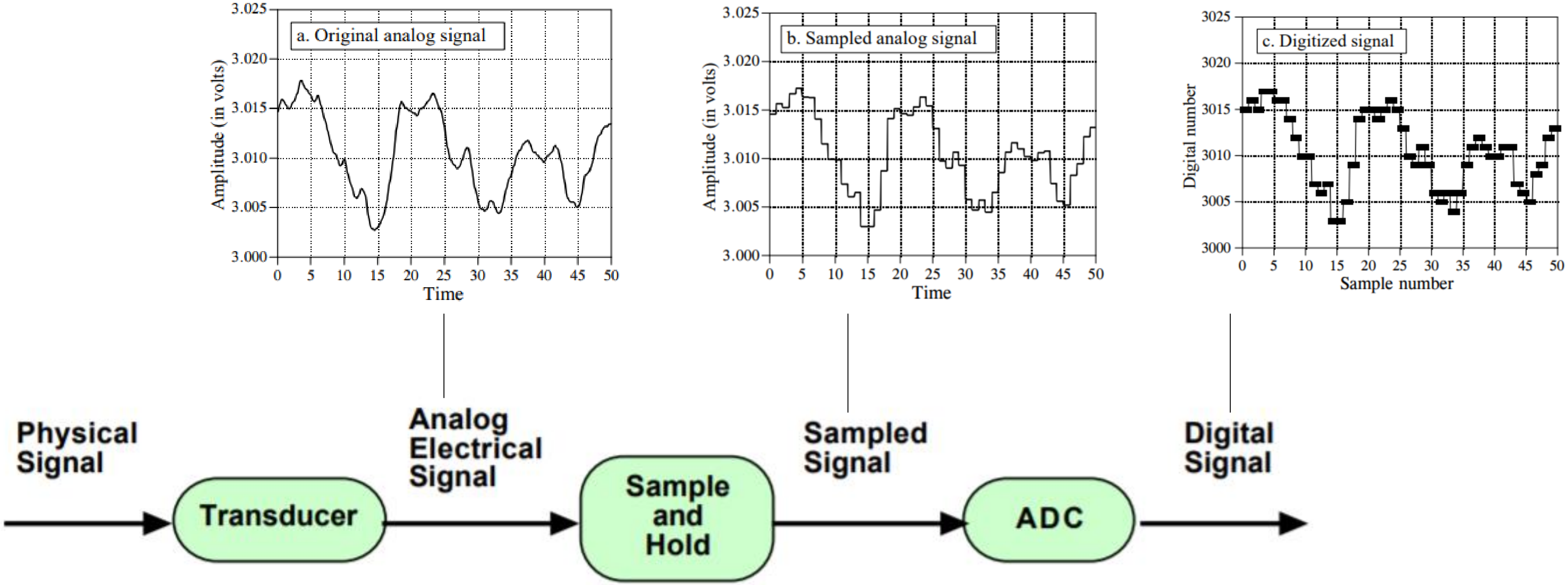



1. **Sampling rate:** How often am I sampling the "real world"?
2. **Quantization**: How many bits are used to represent each sample?

Source: Stiber, Stiber, & Larson, Signal Computing, 2016 (link)

# QUANTIZATION

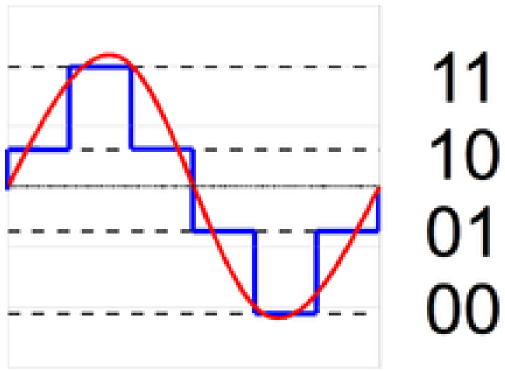
Quantization is the process of digitizing an analog signal. Quantization inherently involves rounding (and thus creates noise).



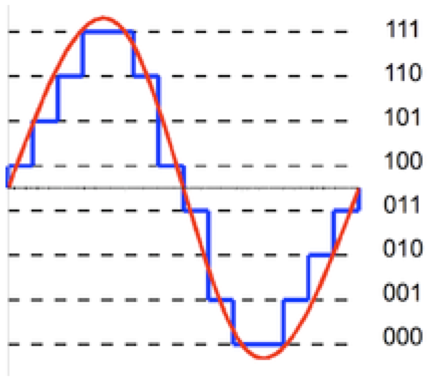

See: Smith, The Scientist and Engineer's Guide to DSP, 1998 (link); Stiber, Stiber, & Larson, Signal Computing, 2016 (link). Images from Wikipedia.

# QUANTIZATION

Quantization is the process of digitizing an analog signal. Quantization inherently involves rounding (and thus creates noise).




**2-Bit Quantization**
2-bit resolution allows for four discretization levels of the analog signal

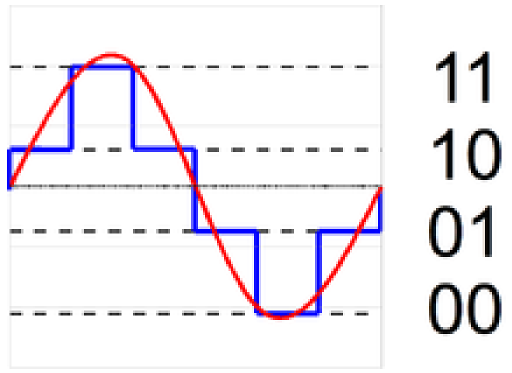



**3-Bit Quantization**
3-bit resolution allows for eight discretization levels of the analog signal

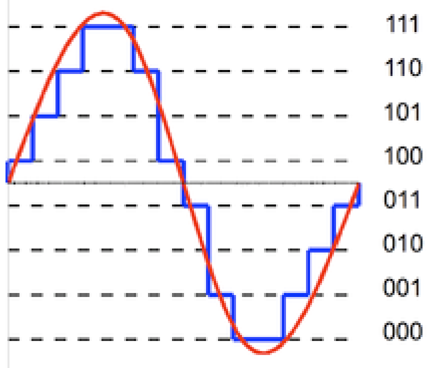
See: Stiber, Stiber, & Larson, Signal Computing, 2016 ([link]()). Images from Wikipedia.

# QUANTIZATION

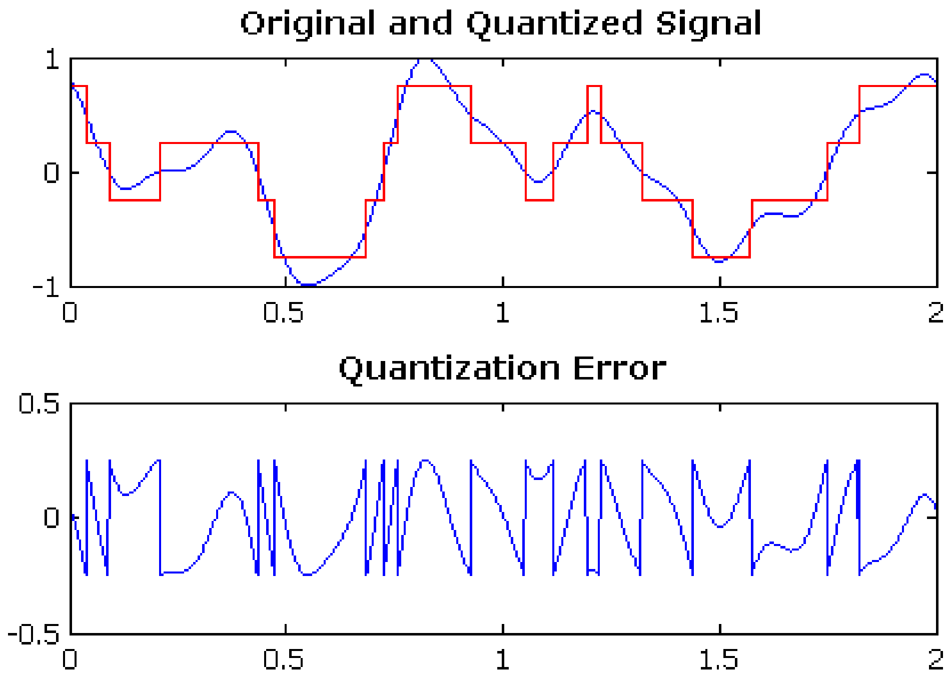
Quantization is the process of digitizing an analog signal. Quantization inherently involves rounding (and thus creates noise).




**2-Bit Quantization**
2-bit resolution allows for four discretization levels of the analog signal




**3-Bit Quantization**
3-bit resolution allows for eight discretization levels of the analog signal

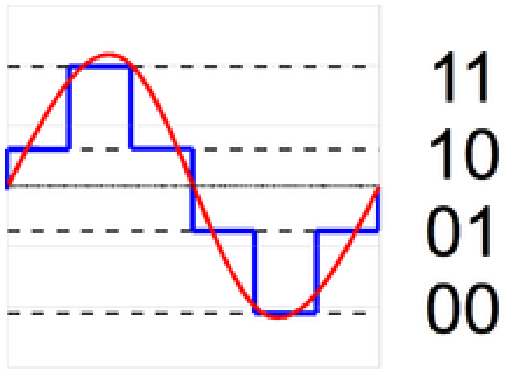



**Quantization Error**
The quantization error is the difference between an input value and its quantized value

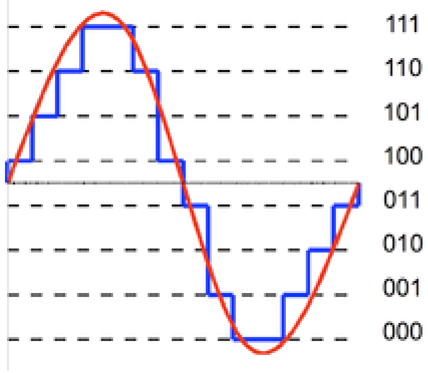
See: Stiber, Stiber, & Larson, Signal Computing, 2016 (link). Images from Wikipedia.

# QUANTIZATION

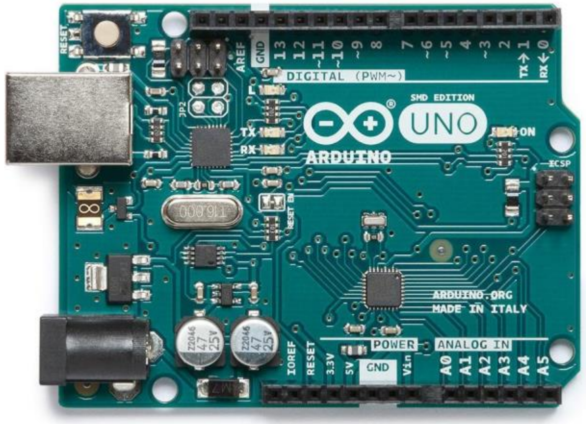
Quantization is the process of digitizing an analog signal. Quantization inherently involves rounding (and thus creates noise).




**2-Bit Quantization**
2-bit resolution allows for four discretization levels of the analog signal




**3-Bit Quantization**
3-bit resolution allows for eight discretization levels of the analog signal



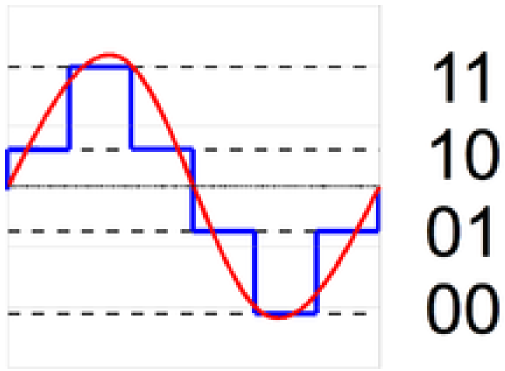

**What is the resolution of the Arduino ADC?**

See: Stiber, Stiber, & Larson, Signal Computing, 2016 ([link](link)). Images from Wikipedia.

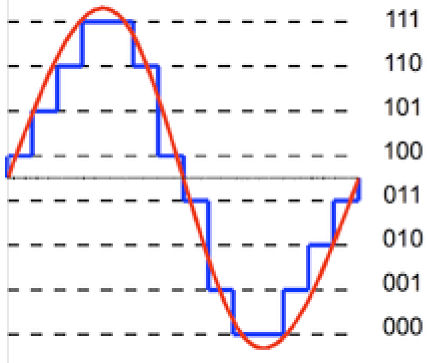
SIGNAL ACQUISITION

# QUANTIZATION

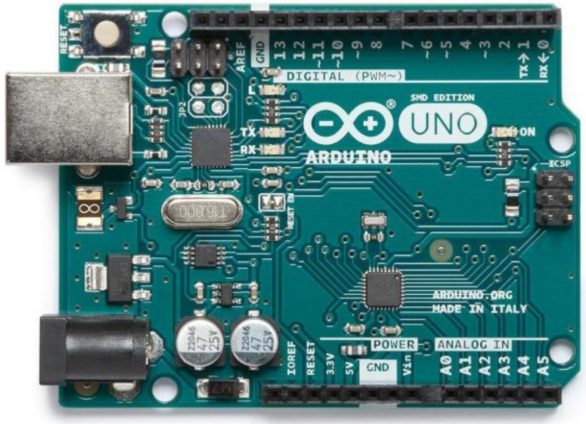
Quantization is the process of digitizing an analog signal. Quantization inherently involves rounding (and thus creates noise).




**2-Bit Quantization**
2-bit resolution allows for four discretization levels of the analog signal




**3-Bit Quantization**
3-bit resolution allows for eight discretization levels of the analog signal



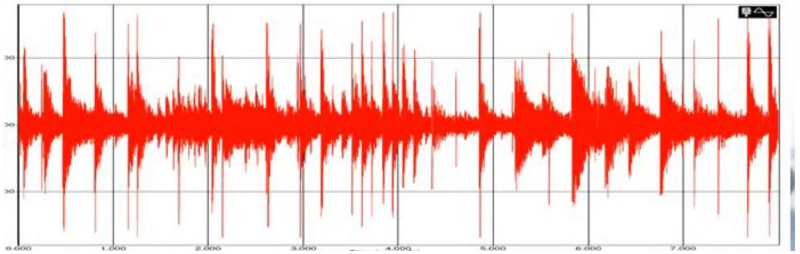

**What is the resolution of the Arduino ADC?**
The Arduino Uno and Leonardo use a 10-bit ADC. So, at 5V, the minimum detectable voltage change is 4.88mV (or 5/1024).

# QUANTIZATION EXAMPLE: AUDIO

I am going to play through a jazz recording at different quantization levels from the high-quality (24 bits) down to the lowest quality

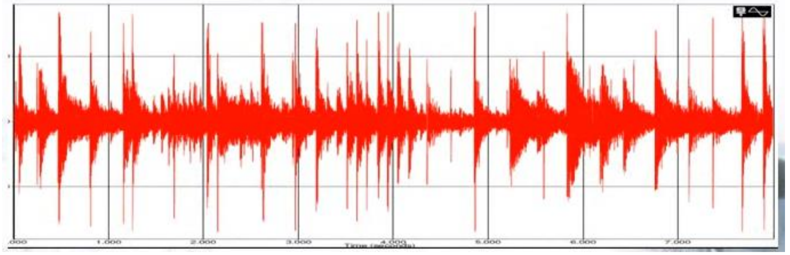
**24 BITS**

16+ million levels



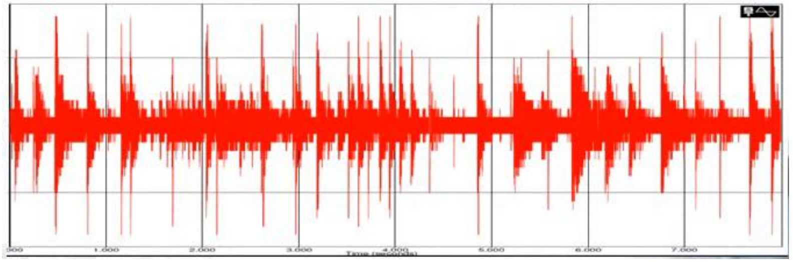
**8 BITS**

256 levels



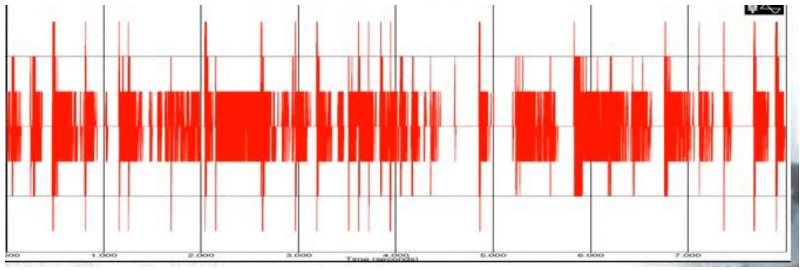
**5 BITS**

32 levels



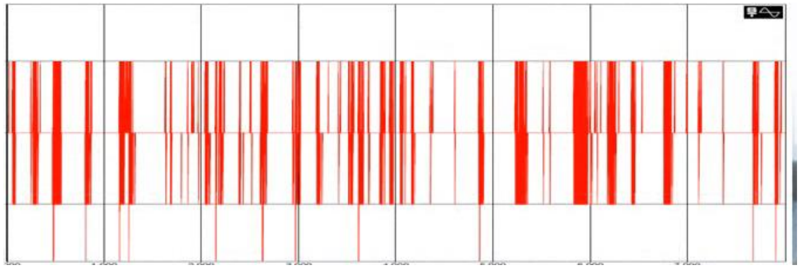
**3 BITS**

8 levels

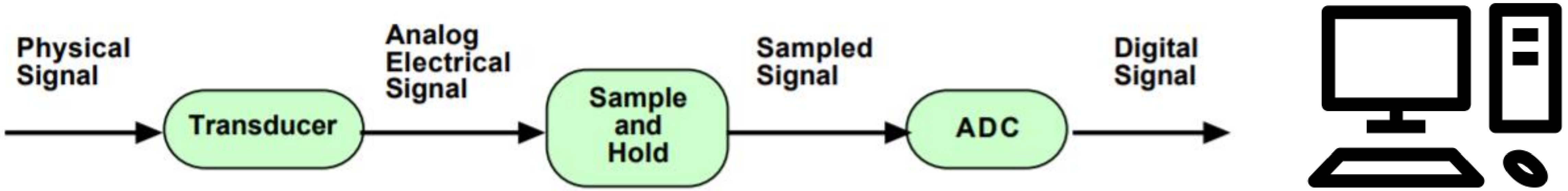


**2 BITS**

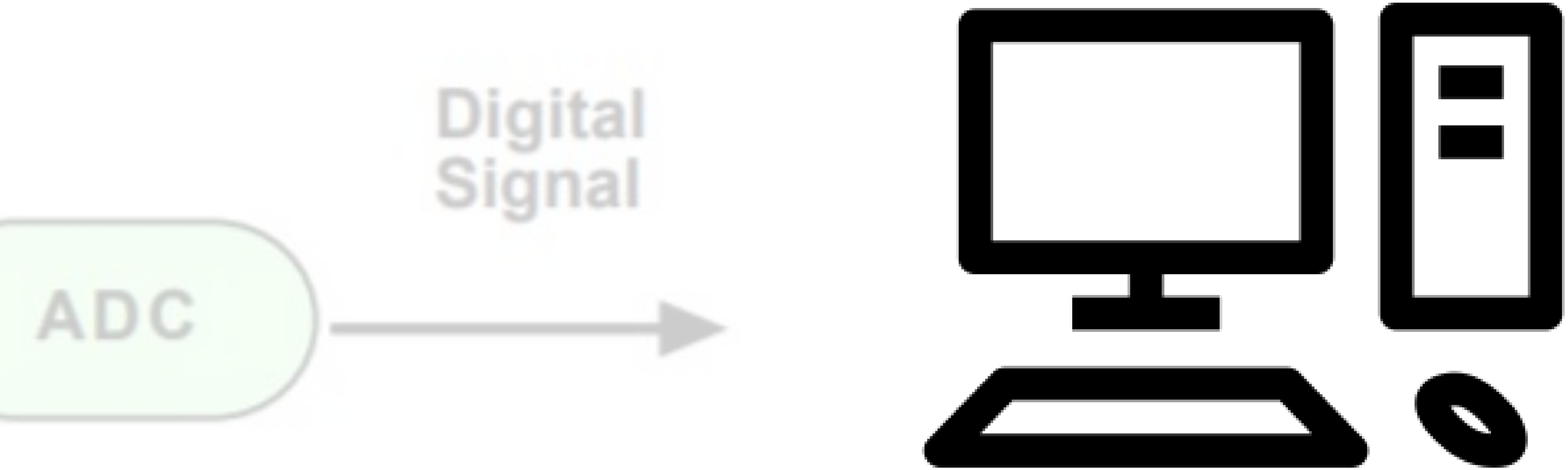
4 levels



Source: https://youtu.be/_cRFBBnUFug




**Processing the digital signal**
(aka digital signal processing or DSP)




**Processing the digital signal**

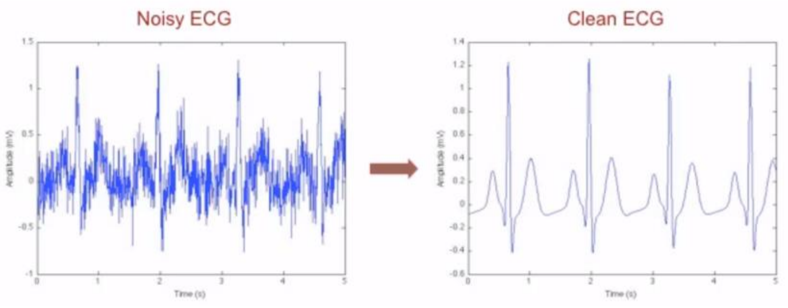
(aka digital signal processing or DSP)

# WHAT IS SIGNAL PROCESSING?

Manipulating/processing a signal to change its characteristics or to extract information (*e.g.*, filtering, transforming, correlating).

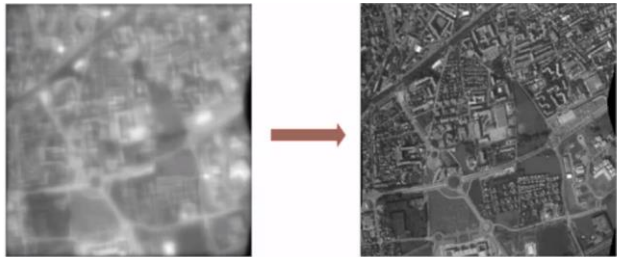
### ELIMINATING NOISE

(*e.g.*, smoothing ECG)



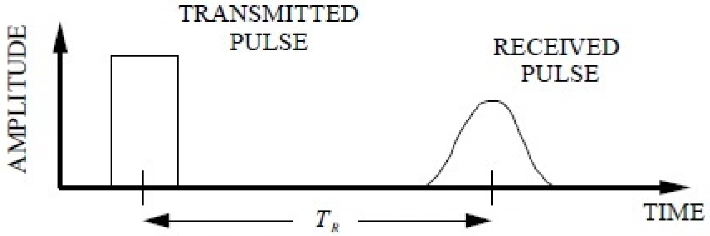

### CORRECTING DISTORTION

(*e.g.*, Hubble lens)



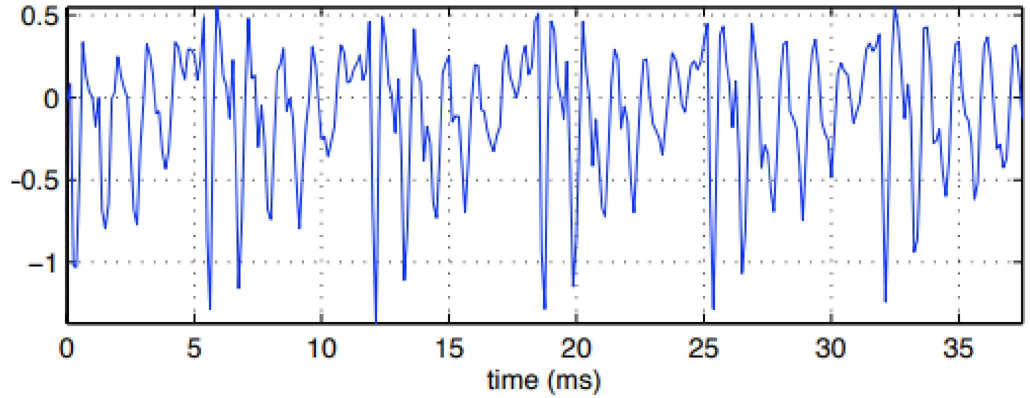
### EXTRACTING INFORMATION

(*e.g.*, distance & velocity from radar pulse)

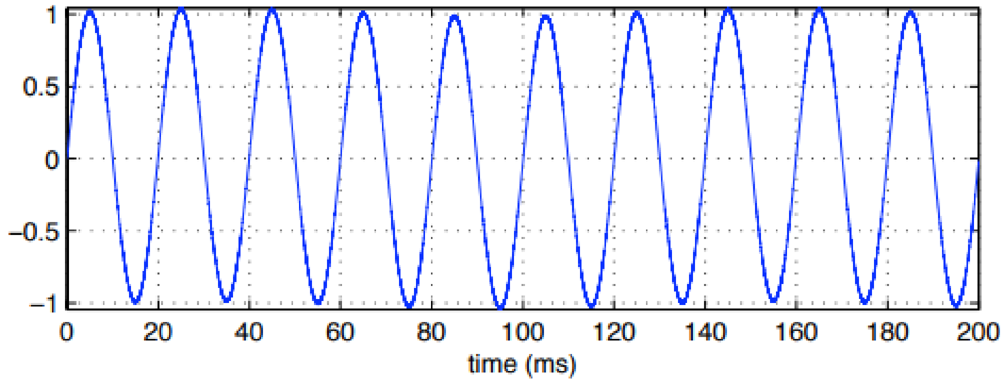



See: Barry Van Veen's website: http://allsignalprocessing.com; Images from: https://youtu.be/YmSvQe2FDKs

# BREAKING DOWN A SIGNAL: WHAT DO YOU SEE?




**The human speech of the vowel 'a' as in 'bat.'**

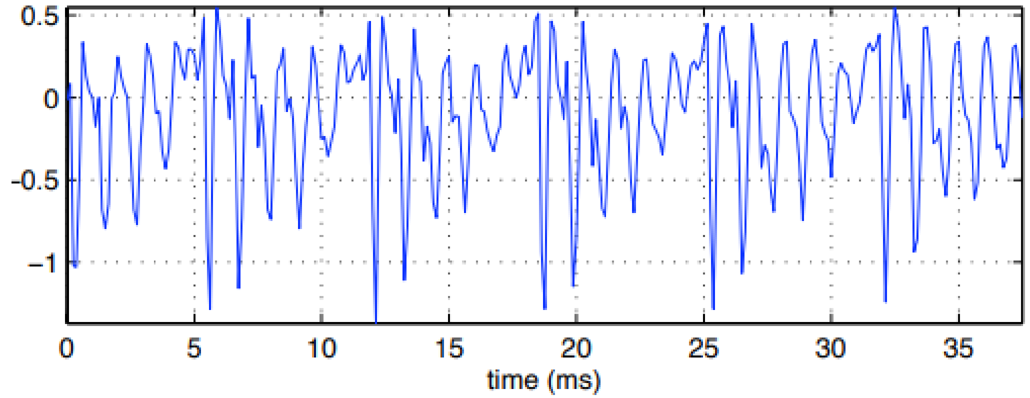



**The ringing of a tuning fork**

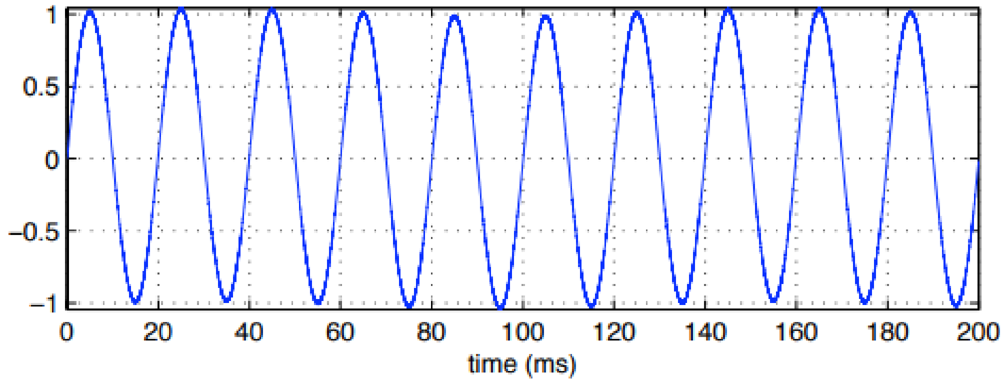
Source: Stiber, Stiber, & Larson, Signal Computing, 2016 ([link](link))

# BREAKING DOWN A SIGNAL: WHAT DO YOU SEE?

These time-series signals are called *periodic* signals because they repeat. It turns out that any periodic signal can be represented as a sum of related sines and cosines, which forms the basis of all signal processing.




**The human speech of the vowel 'a' as in 'bat.'**



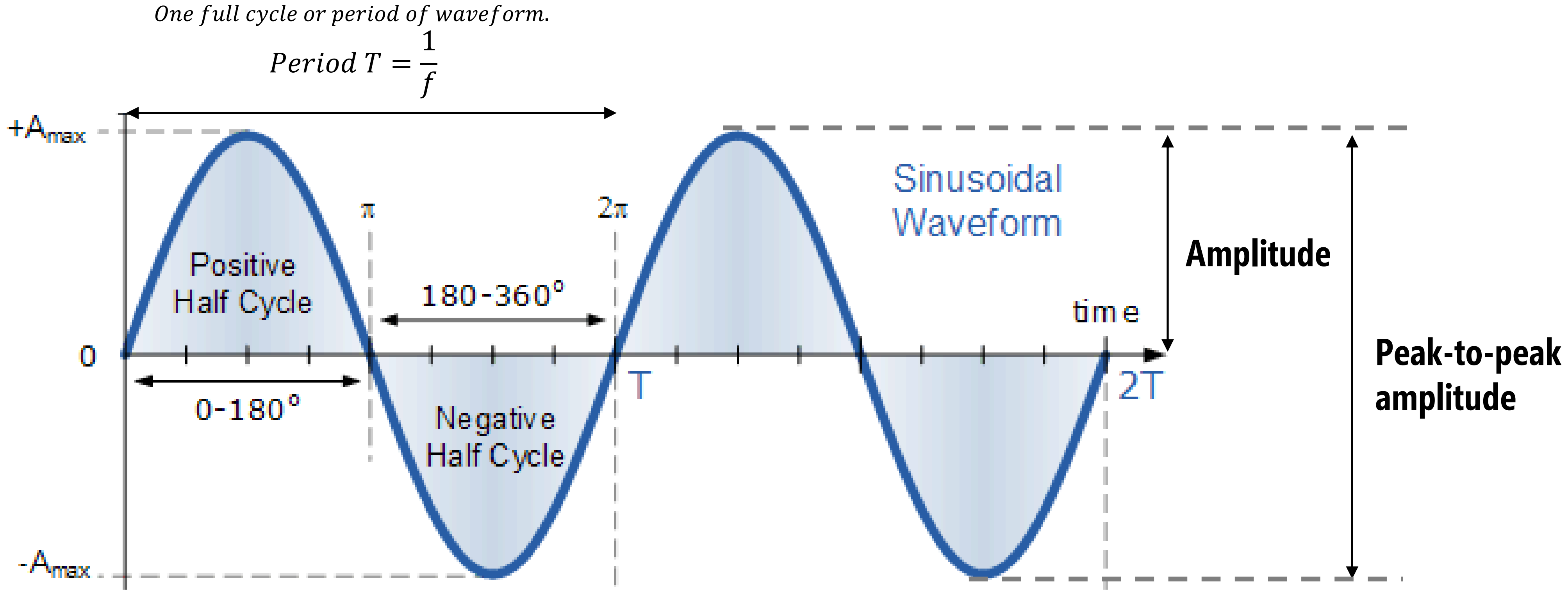

**The ringing of a tuning fork**

$$Periodic\ signals\ are\ represented\ by\ g(t) = \sin(\omega t + \theta) = \sin(2\pi f t + \theta)$$

$$Cosine\ is\ just\ sin\ phase - shifted\ by\ \frac{\pi}{2}:\ \cos(\omega t) = \sin(\omega t + \frac{\pi}{2})$$

Source: Stiber, Stiber, & Larson, Signal Computing, 2016 ([link](link))

# SINUSOIDS




$$g(t) = A\sin(\omega t + \theta) = A\sin(2\pi f t + \theta)$$

***A*** = wave amplitude

***ω*** = angular frequency (how many oscillations occur in a unit time interval)

***t*** = instantaneous time

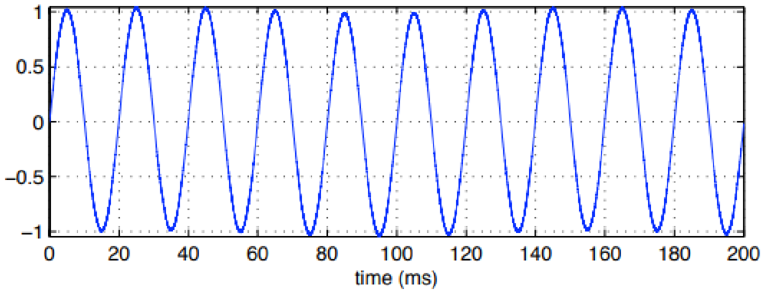
***θ*** = horizontal phase shift

Source: Stiber, Stiber, & Larson, Signal Computing, 2016 (link)

# REPRESENTATIONS OF SINUSOIDS

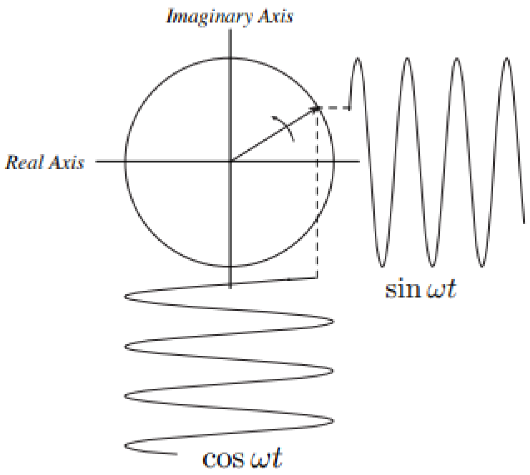
Rather than thinking of a sinusoid as a function which oscillates up and down, you can think of it as something that goes round and round. This is the phasor representation, which links to the complex exponential representation.

## SINUSOIDAL REPRESENTATION




$$g(t) = \sin(\omega t + \theta)$$

## PHASOR REPRESENTATION




## COMPLEX EXPONENTIAL

$$e^{j\theta(t)} = e^{j\omega t}$$

Any periodic function can be written as a sum of phasors represented by these complex exponentials. A super powerful idea used in lots signal processing (like Fourier transforms!).

Source: Stiber, Stiber, & Larson, Signal Computing, 2016 (link)

# SINUSOIDS

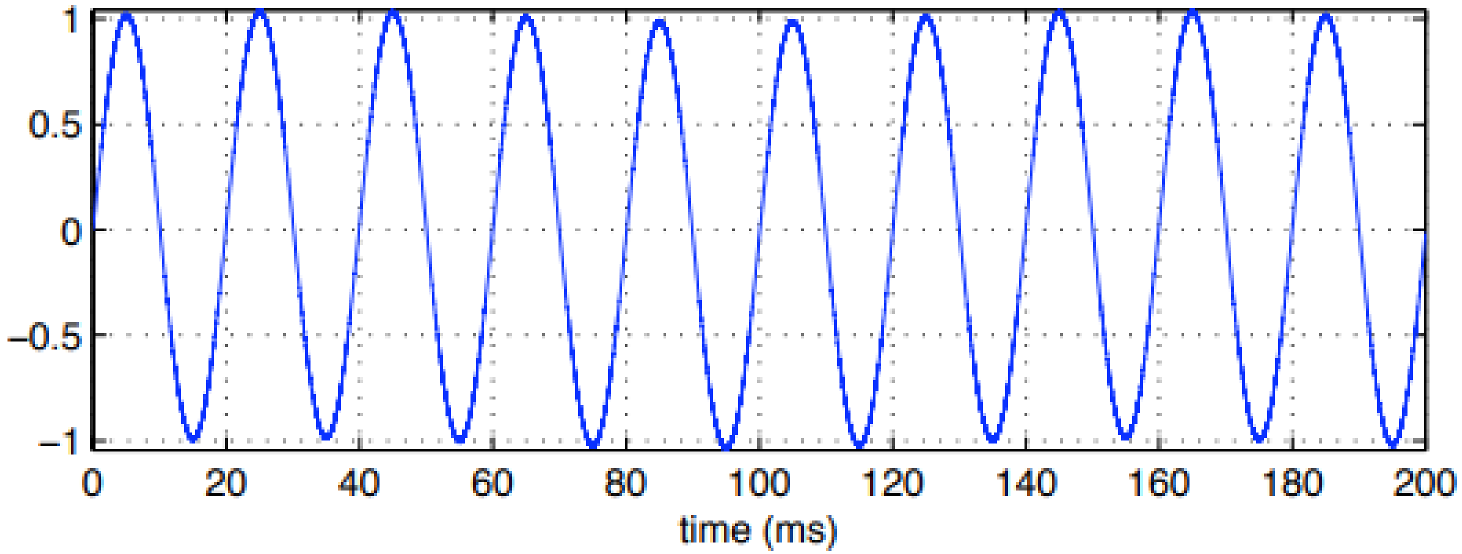
Do this **individually** first. I will then ask you to pair up with someone next to you to share and discuss your answer.

**Think, Pair, Share**

What is the amplitude of this signal?

What is the period (T) of the signal?

What is the frequency (*f*) of this signal?

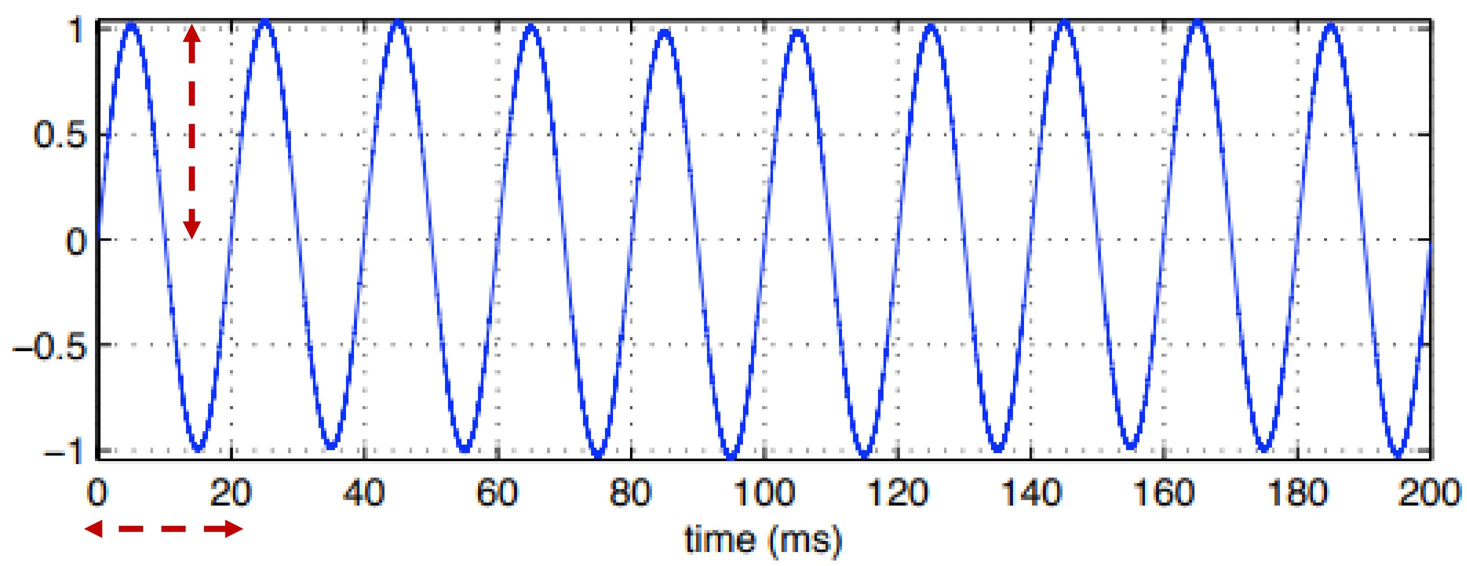



$$g(t) = \sin(\omega t + \theta) = \sin(2\pi f t + \theta)$$

Image source: Stiber, Stiber, & Larson, Signal Computing, 2016 (link)

# SINUSOIDS

Amplitude = 1




T = 0.02s

$$g(t) = \sin(\omega t + \theta) = \sin(2\pi f t + \theta)$$

**Think, Pair, Share**

What is the amplitude of this signal?

What is the period (T) of this signal?

What is the frequency (*f*) of this signal?

*One full cycle or period of waveform.*

$$Period\ T = \frac{1}{f}$$

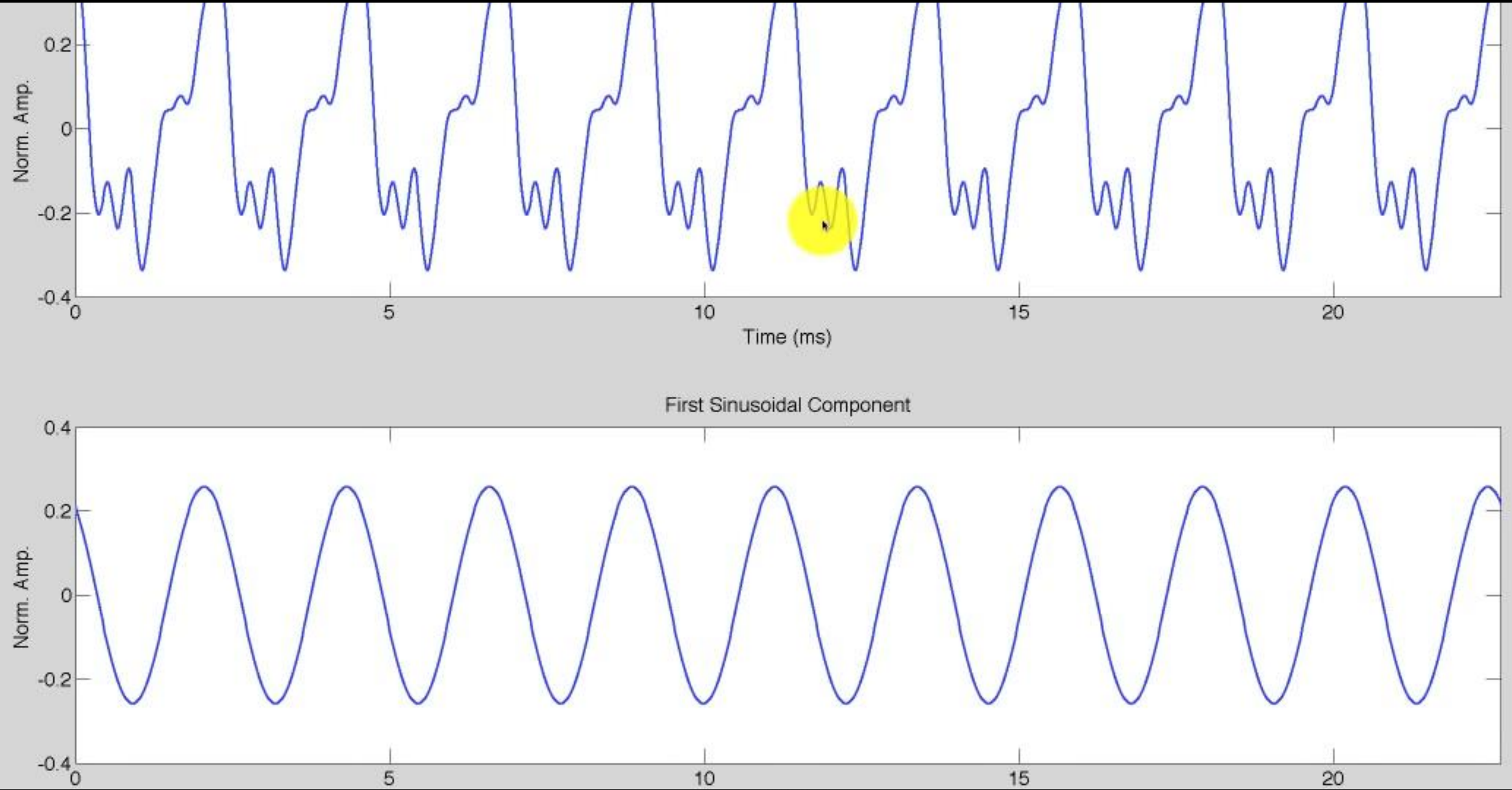
*Thus*

$$Frequency\ f = \frac{1}{T} = \frac{1}{0.02} = 50\ Hz$$

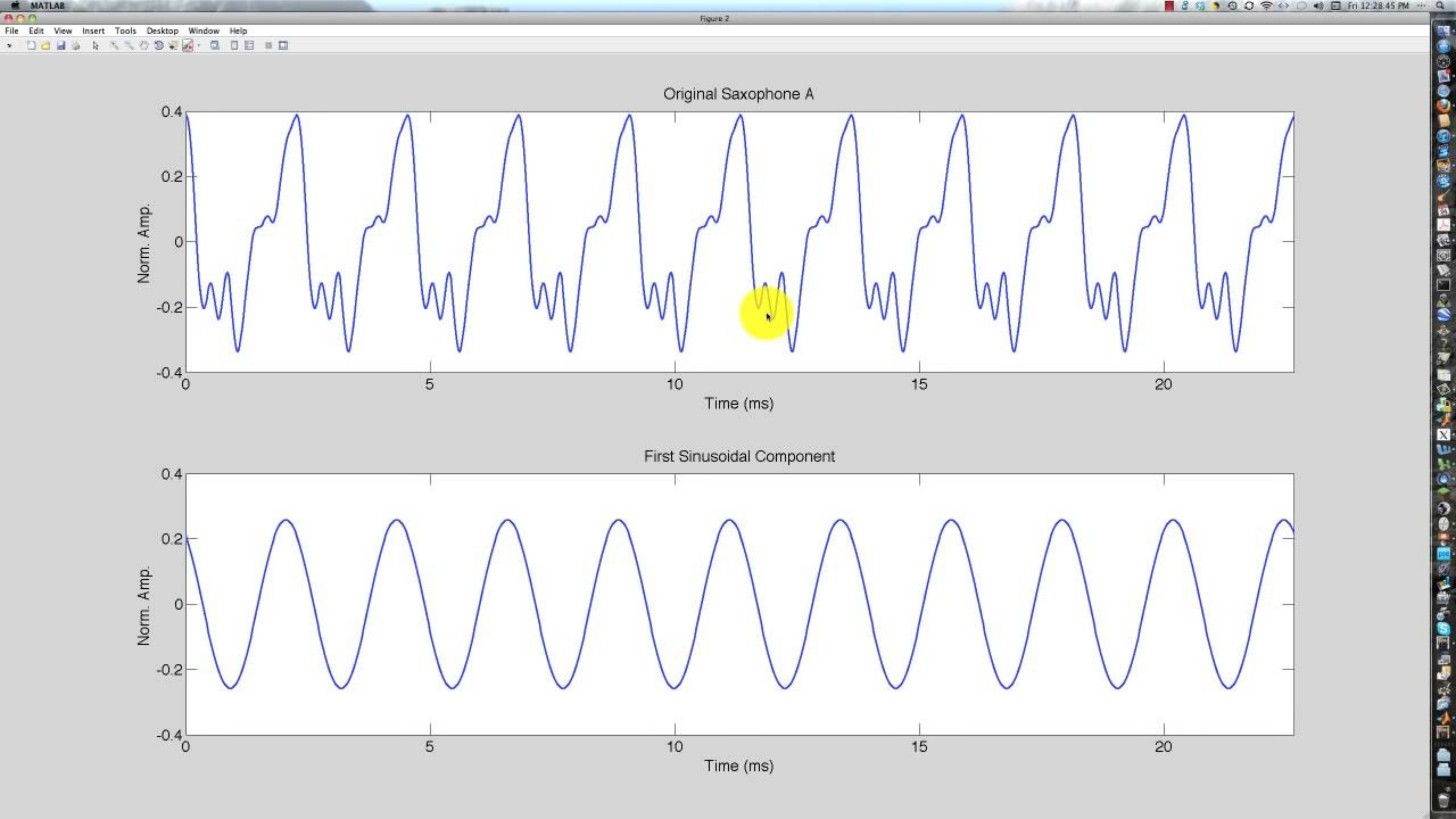
Image source: Stiber, Stiber, & Larson, Signal Computing, 2016 (link)

INTRODUCTION TO SIGNALS & DSP

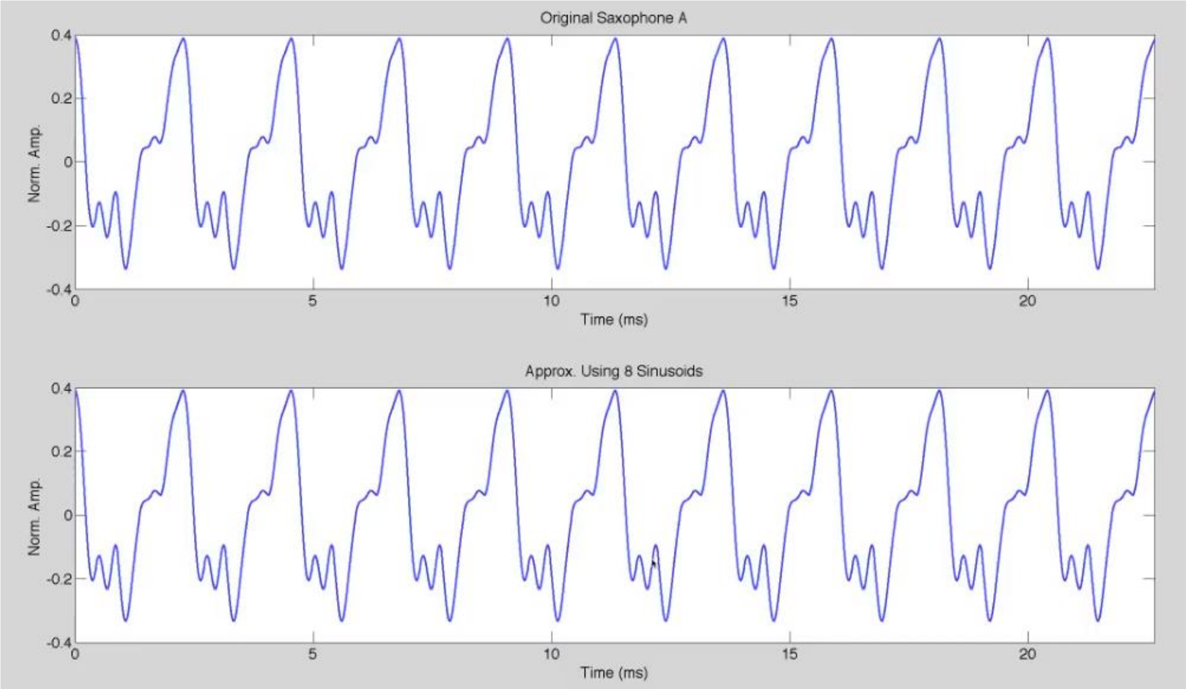
# REPRESENTING COMPLEX PERIODIC SIGNALS BY SUMMING SINUSOIDS




Source: Barry Van Veen's http://AllSignalProcessing.com. Direct video link: https://youtu.be/fCAZ7jcO-vc?t=10m43s


MATLAB
Figure 2
File Edit View Insert Tools Desktop Window Help
Original Saxophone A
0.4
0.2
0
-0.2
-0.4
Norm. Amp.
0 5 10 15 20
Time (ms)
First Sinusoidal Component
0.4
0.2
0
-0.2
-0.4
Norm. Amp.
0 5 10 15 20
Time (ms)

# HOW'D HE DO THAT?




$$
\begin{aligned}
&A_1 * \sin(2\pi f_1 t + \theta) \\
&+ A_2 * \sin(2\pi f_2 t + \theta) \\
&+ A_3 * \sin(2\pi f_3 t + \theta) \\
&+ A_4 * \sin(2\pi f_4 t + \theta) \\
&+ A_5 * \sin(2\pi f_5 t + \theta) \\
&+ A_6 * \sin(2\pi f_6 t + \theta) \\
&+ A_7 * \sin(2\pi f_7 t + \theta) \\
&+ A_8 * \sin(2\pi f_8 t + \theta)
\end{aligned}
$$

Where $f_n$ is an integer multiple of $f_1$ and $A_n$ gets decreasingly small

# SYNTHESIS AND DECOMPOSITION

This is an example of synthesis—combining signals by *scaling* (multiplications of signals by constants like A) and *addition*

**Synthesis**
Two or more signals are added together to form another signal.

**Decomposition**
Opposite from synthesis. Break one signal into two or more additive component signals.

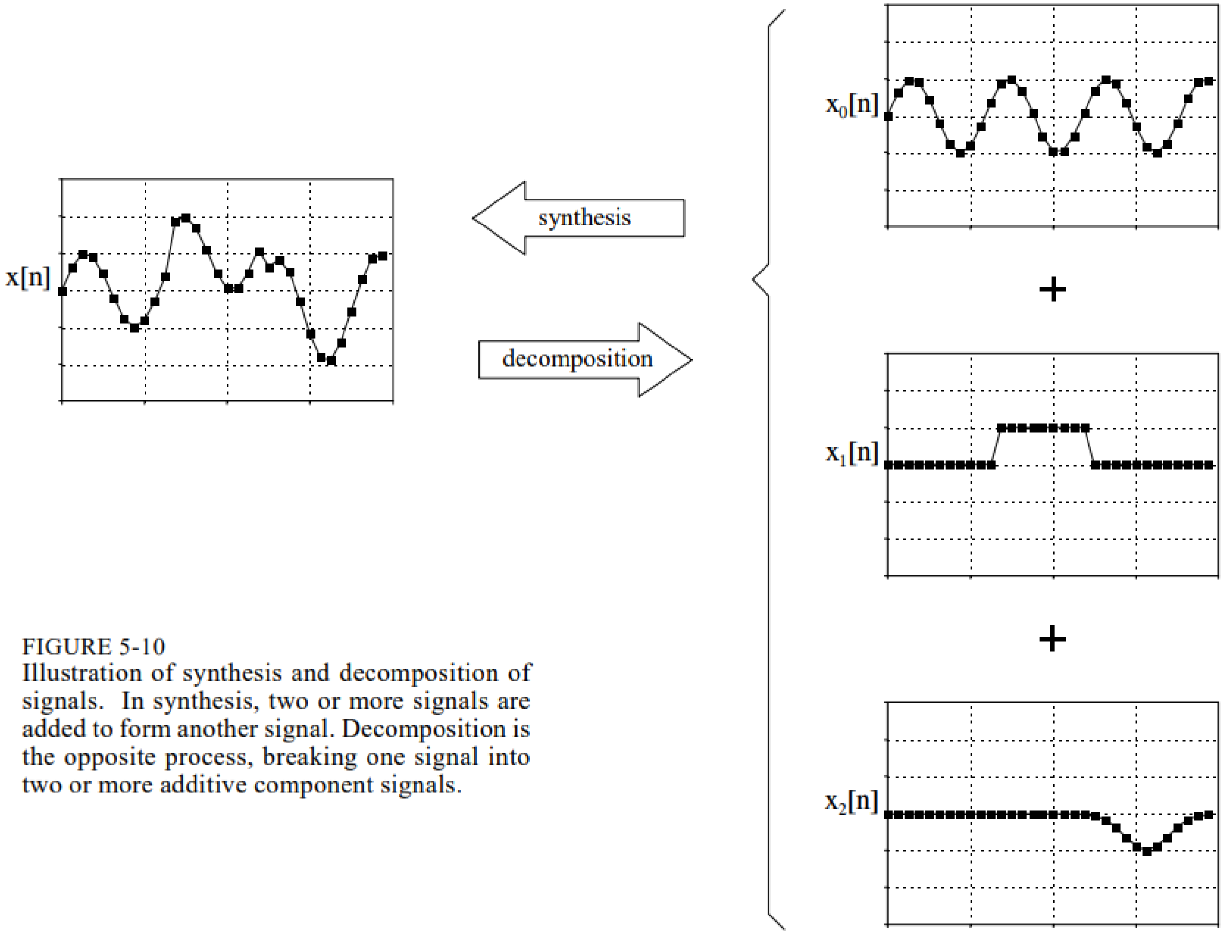
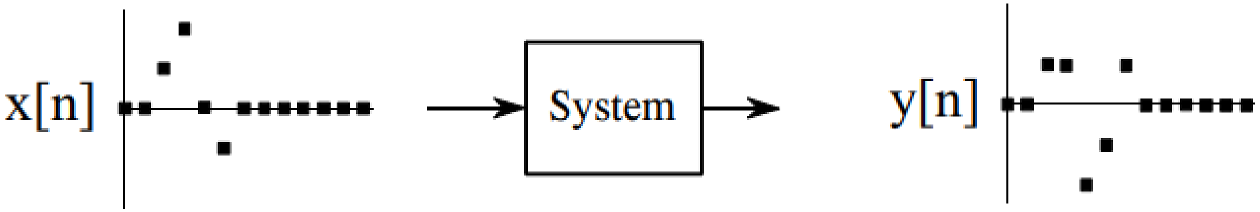


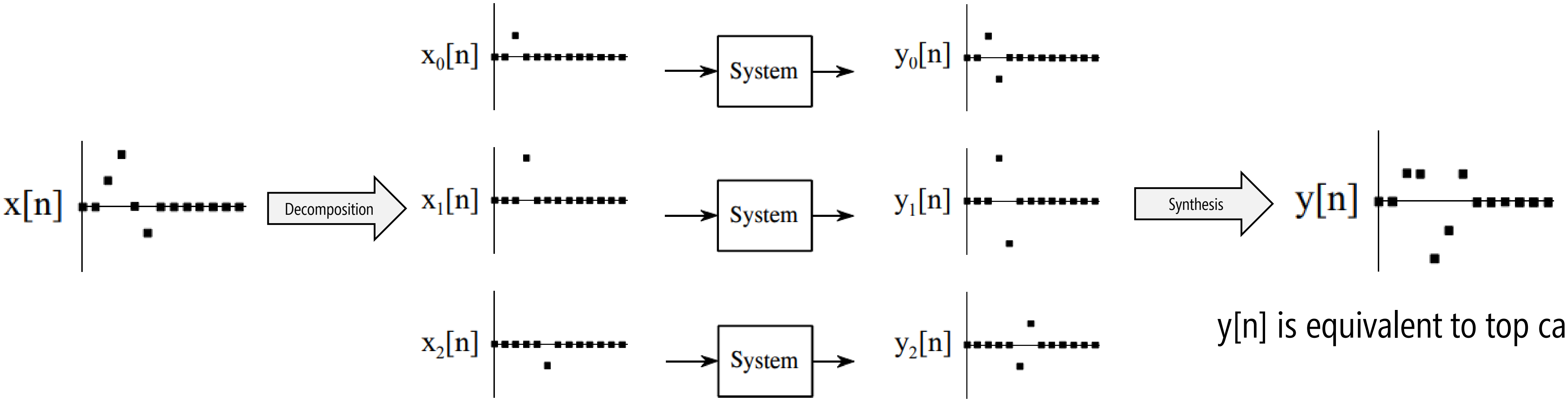

FIGURE 5-10
Illustration of synthesis and decomposition of signals. In synthesis, two or more signals are added to form another signal. Decomposition is the opposite process, breaking one signal into two or more additive component signals.

See: Chapter 5 of Smith, The Scientist and Engineer's Guide to DSP, 1998 ([link]()); note this only works for linear systems

# SUPERPOSITION!

Perhaps the most fundamental concept: passing all individual components of x[n] through a linear system (*e.g.*, a filter) produces output signals that, when synthesized (via addition), these output signals form the same signal produced when x[n] is passed through the system. This is the basis of nearly all signal processing techniques.




y[n] is equivalent to top case

See: Chapter 5 of Smith, The Scientist and Engineer's Guide to DSP, 1998 (link); Superposition only holds for linear systems