Speech Processing:
Digital Speech Signals

Module 3

Catherine Lai

5 October 2021

Today

- Where are we now? (Interpreting spectrograms)
- Time Domain and Frequency Domain
- Digital speech signals
- Discrete Fourier Transform → What are spectrograms, really?

So far

- Module 1: Phonetics and visual representations of speech
- Module 2: Acoustics of Consonants and vowels

How can we characterise speech from articulatory and acoustic perspectives?

THE INTERNATIONAL PHONETIC ALPHABET (revised to 2015)

CONSONANTS (PULMONIC)

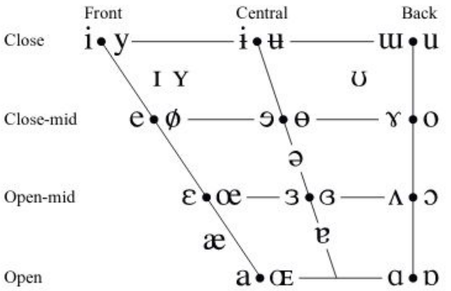
	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k ɡ	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Symbols to the right in a cell are voiced, to the left are voiceless. Shaded areas denote articulations judged impossible.

CONSONANTS (NON-PULMONIC)

Clicks		Voiced implosives		Ejectives	
ʘ	Bilabial	ɓ	Bilabial	ʼ	Examples:
ǀ	Dental	ɗ	Dental/alveolar	pʼ	Bilabial
ǃ	(Post)alveolar	ʄ	Palatal	tʼ	Dental/alveolar
ǂ	Palatoalveolar	ɠ	Velar	kʼ	Velar
ǁ	Alveolar lateral	ʛ	Uvular	sʼ	Alveolar fricative

VOWELS



Where symbols appear in pairs, the one to the right represents a rounded vowel.

OTHER SYMBOLS

ʍ Voiceless labial-velar fricative

w Voiced labial-velar approximant

ɥ Voiced labial-palatal approximant

ʜ Voiceless epiglottal fricative

ʢ Voiced epiglottal fricative

ɕ ʑ Alveolo-palatal fricatives

ɺ Voiced alveolar lateral flap

ɧ Simultaneous ʃ and x

Affricates and double articulations can be represented by two symbols joined by a tie bar if necessary. t͡s k͡p

SUPRASEGMENTALS

ˈ Primary stress

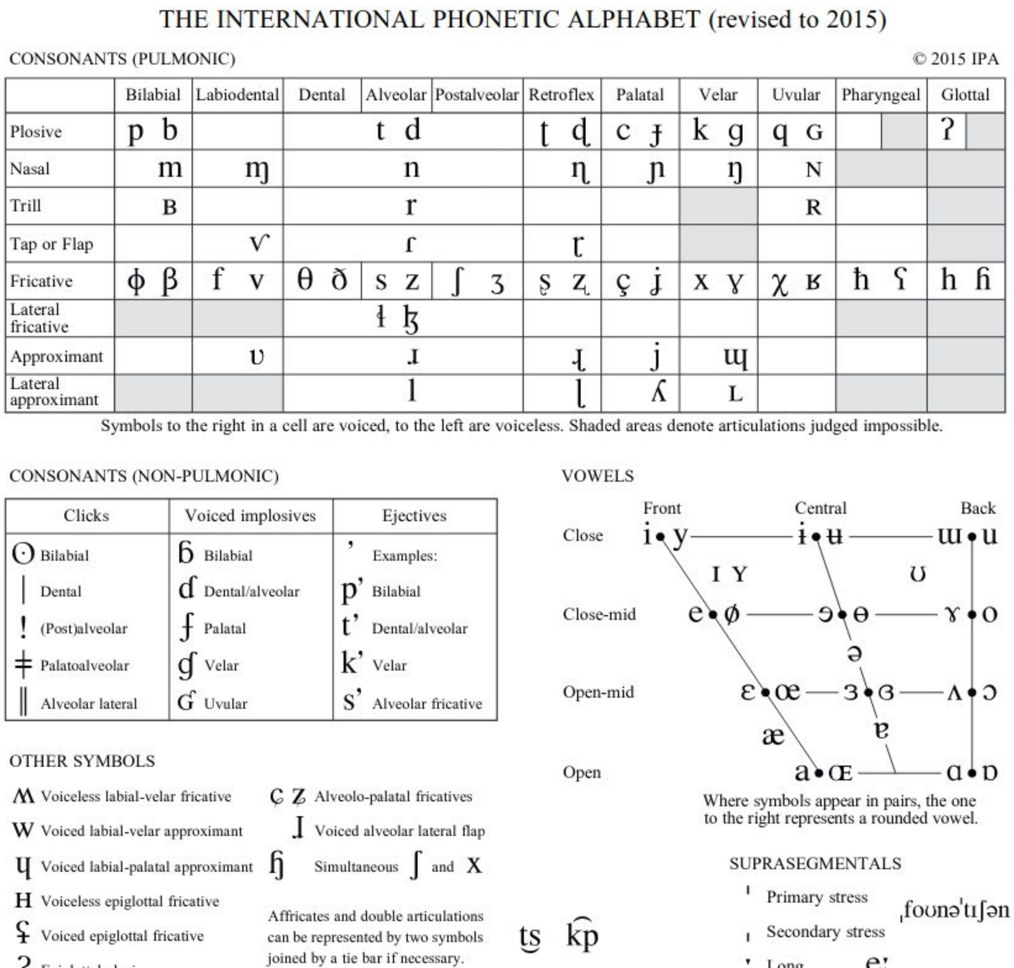
ˌ Secondary stress

ˌfoʊnəˈtɪʃən

ː Long eː

From modules 1 & 2, you should be able to:

- Describe how speech sounds in terms of **manner** and **place** of articulation
- Know enough vocal anatomy/phonetics terminology to **read and interpret the IPA chart**
- You **don't** need to memorize all the symbols or to make all the sounds!



- If you don't have a phon background it may take some time to absorb. That's ok!
- Try to build on and consolidate the concepts from module 1 & 2 through the semester

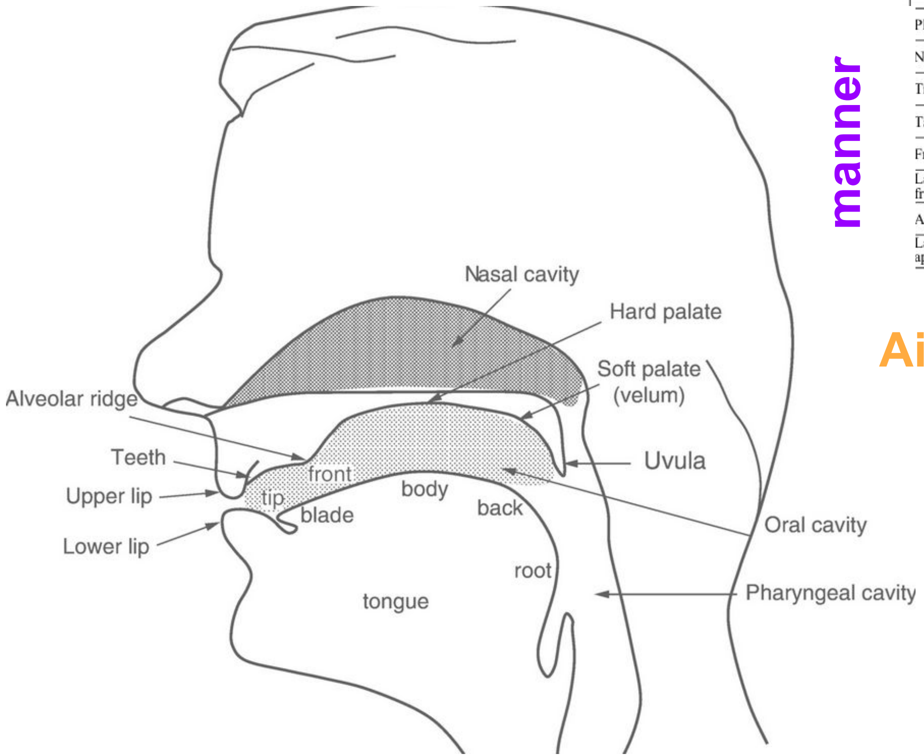
Assessment:

- ONLINE TEST WEEK 5

  Phon/Signals (15%): Open on Learn Mon 12pm 16/10/23 - Wed 12pm 18/10/23

- Use these concepts in the assignments to help give your analyses more depth

Consonants



Place of articulation

manner

CONSONANTS (PULMONIC) © 2015 IPA

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k ɡ	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Symbols to the right in a cell are voiced, to the left are voiceless. Shaded areas denote articulations judged impossible.

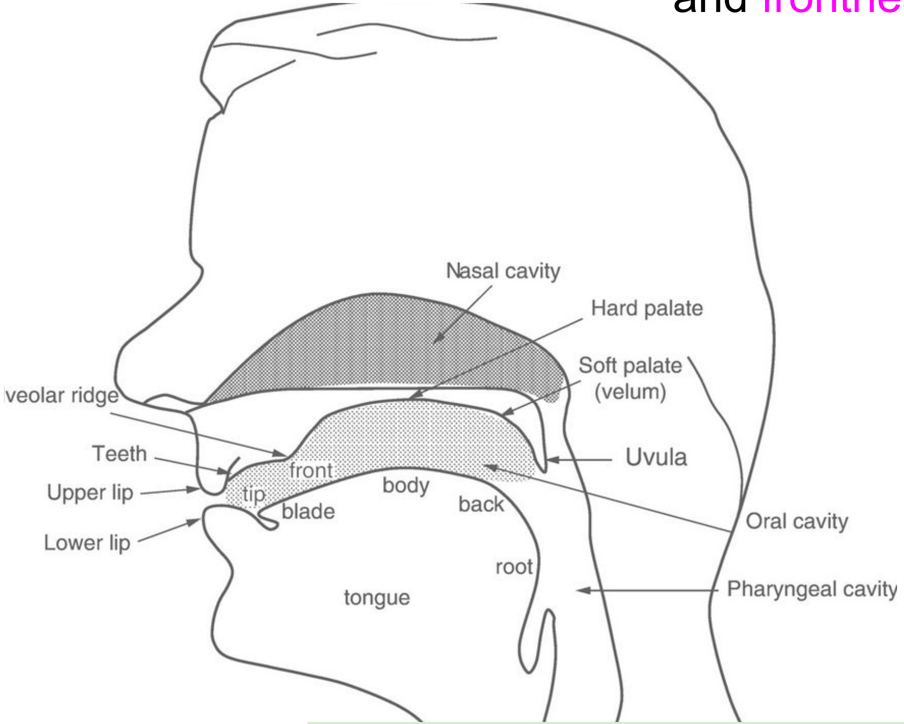
Air stream

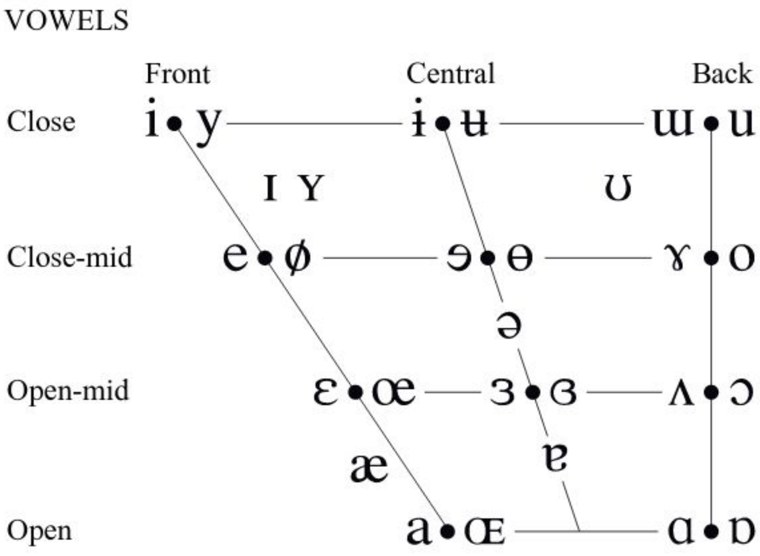
CONSONANTS (NON-PULMONIC)

Clicks	Voiced implosives	Ejectives
ʘ Bilabial	ɓ Bilabial	ʼ Examples:
ǀ Dental	ɗ Dental/alveolar	pʼ Bilabial
ǃ (Post)alveolar	ʄ Palatal	tʼ Dental/alveolar
ǂ Palatoalveolar	ɠ Velar	kʼ Velar
ǁ Alveolar lateral	ʛ Uvular	sʼ Alveolar fricative

Vowels

Vowels are mainly characterised by their height and frontness (tongue position), and lip roundness



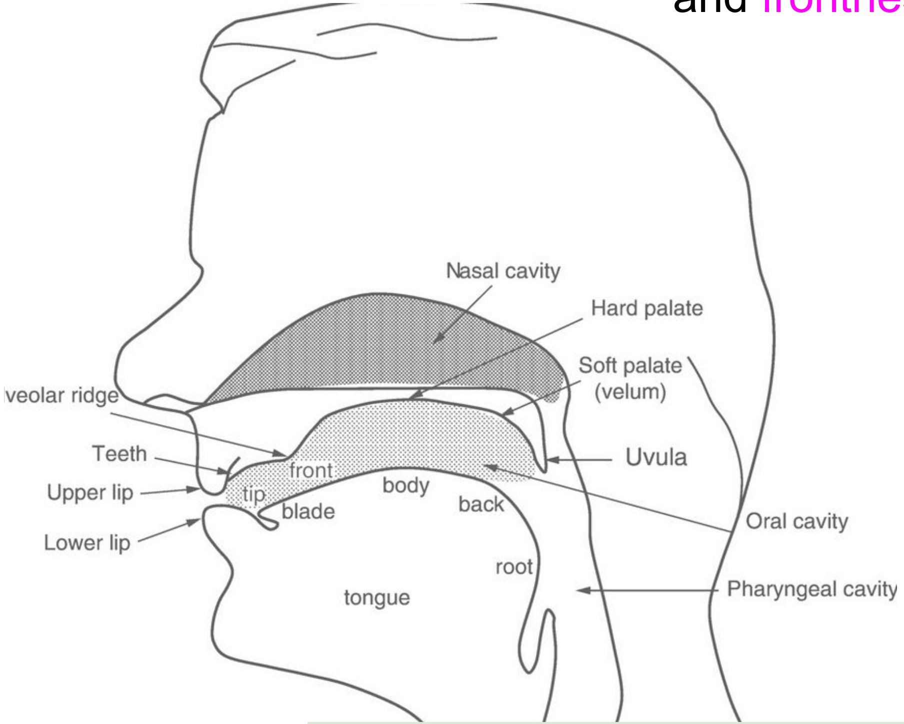


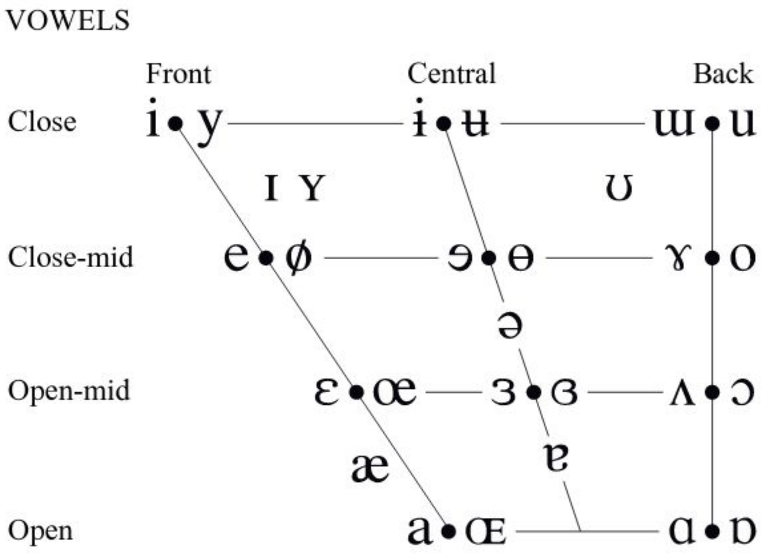
Where symbols appear in pairs, the one to the right represents a rounded vowel.

Computers can't see into our mouths (generally speaking!) so we want to derive features from the speech signal that we can use to identify different speech sounds → **articulatory/acoustic mapping**

Vowels

Vowels are mainly characterised by their height and frontness (tongue position), and lip roundness





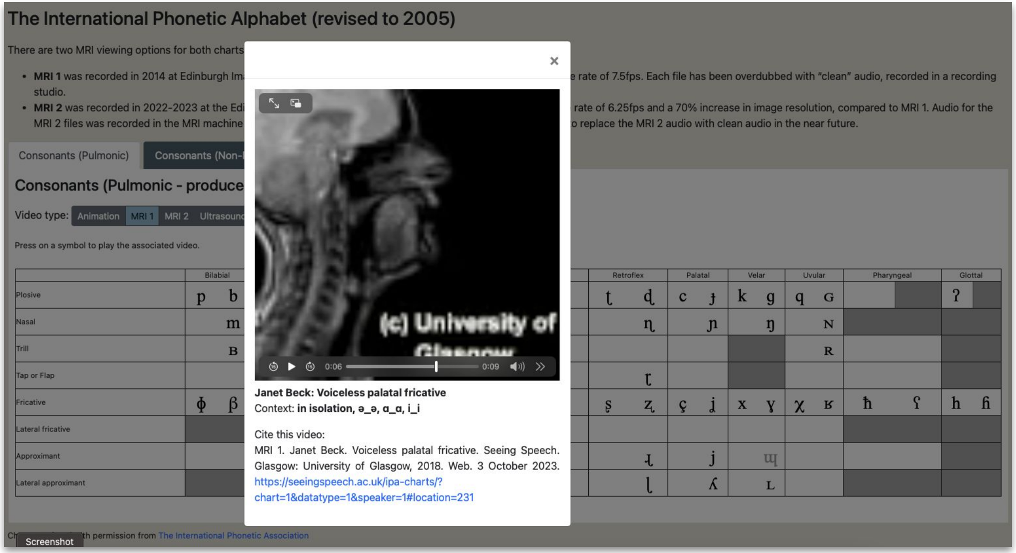
Where symbols appear in pairs, the one to the right represents a rounded vowel.

articulatory/acoustic mapping: what can we infer about articulation from the sound wave? What features of the sound wave are informative of this?

Study aid: Seeing Speech

An interactive IPA chart with MRI, X-ray and animations of speech sounds:

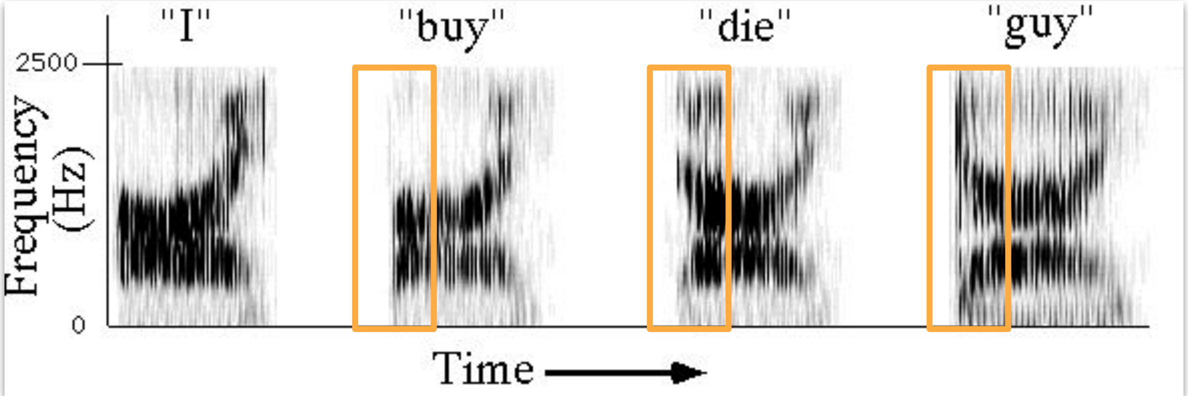
https://www.seeingspeech.ac.uk/ipa-charts/



Tip: Phonetics is generally easier to learn by doing! Try looking at the articulators in the video and try it yourself. Record yourself and look at the spectrogram in Praat. You'll pick up the terminology with practice (our tests are open book anyway)!

Acoustic phonetics

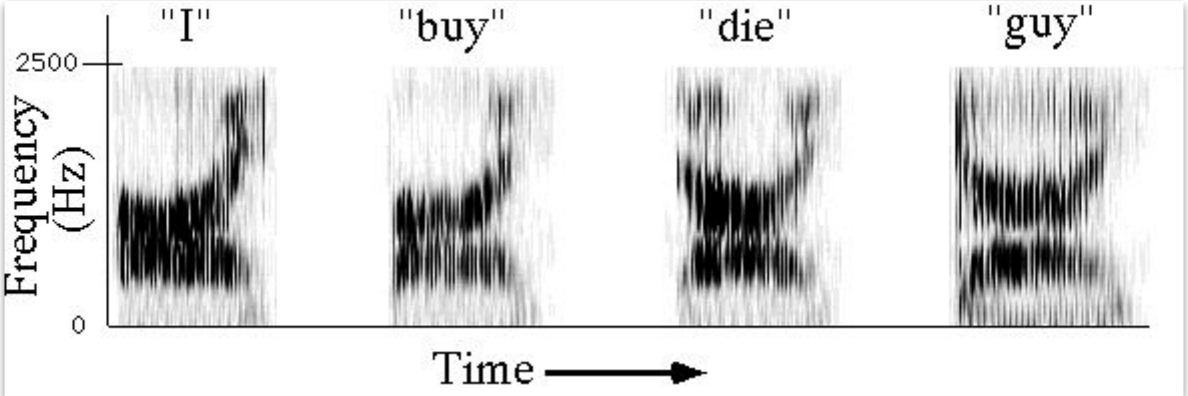
Oral stops (aka plosives) at the start of a word can be distinguished by formant transitions into the following vowel



From phonetics lecture notes by Louis Goldstein

We can “see” differences in place of articulation and manner of speech sounds by looking at how at the spectral characteristics of speech (i.e. the frequencies present in the sound) and how it changes over time → spectrograms

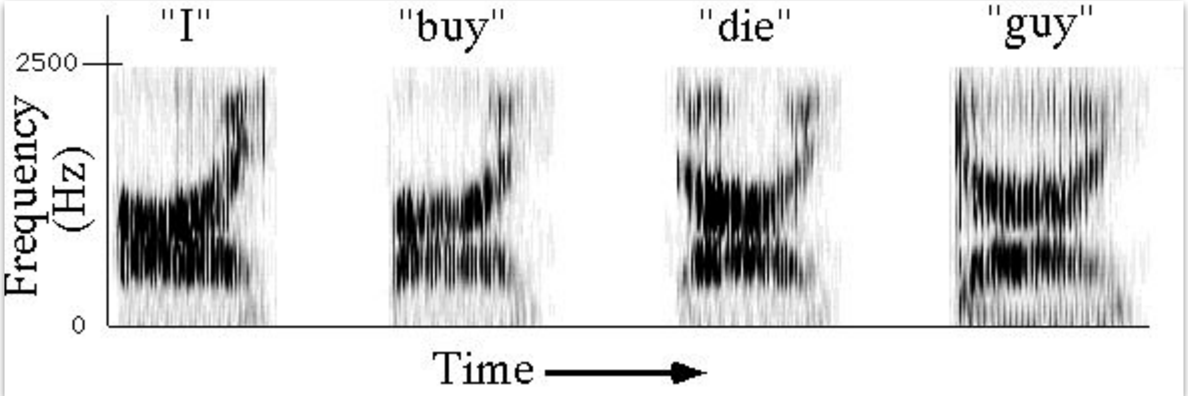
Acoustic phonetics



From phonetics lecture notes by Louis Goldstein

This frequency information is key to both automatic speech recognition and speech synthesis: We can determine what is being said without seeing the actual articulations, and we can generate sounds with a vocal tract!

Acoustic phonetics



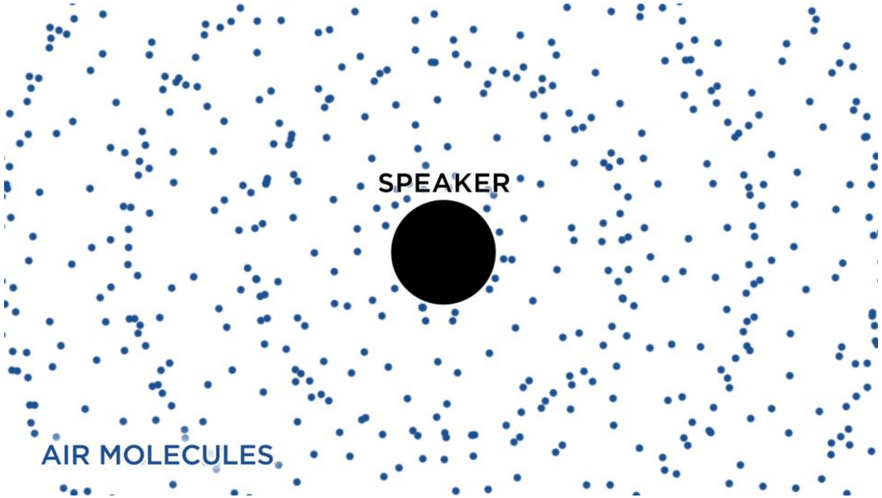
From phonetics lecture notes by Louis Goldstein

Question of the week: We how do we go from sound in the real world to a spectrogram on a computer?

Sound in the Time Domain and the Frequency Domain

Sound waves

Air particles bounce back and forth at different frequencies. This causes changes in pressure in the air: particles squashed together → higher air pressure

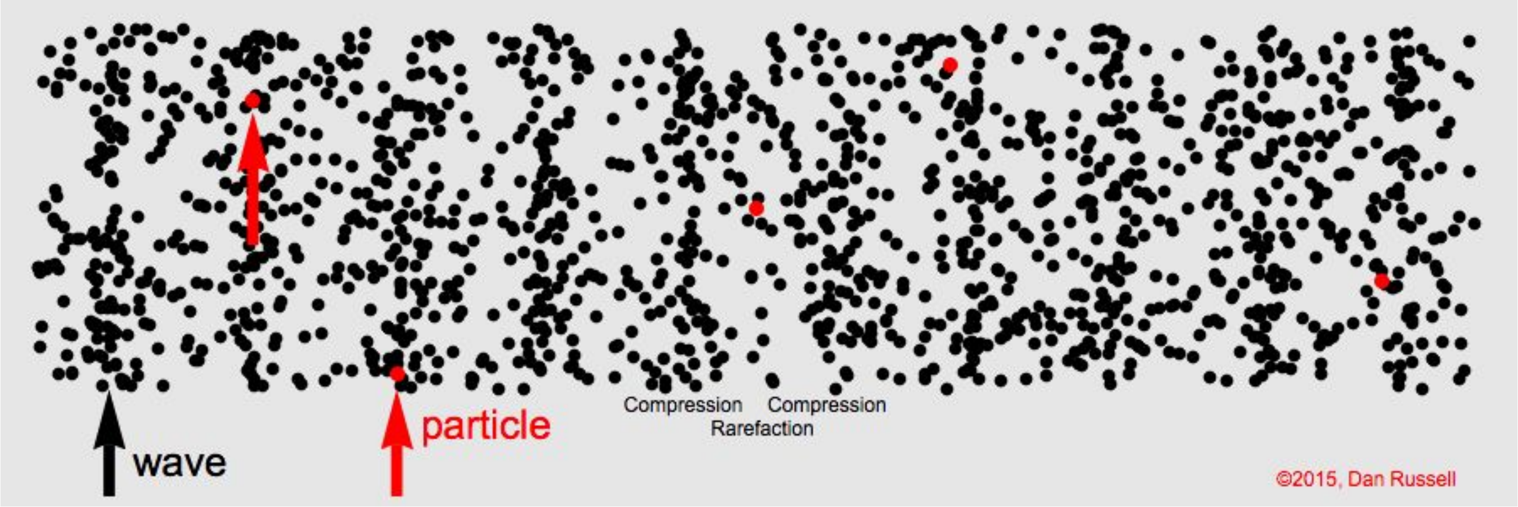


https://media.npr.org/assets/img/2014/04/08/compressionwave.gif

Sound waves in air

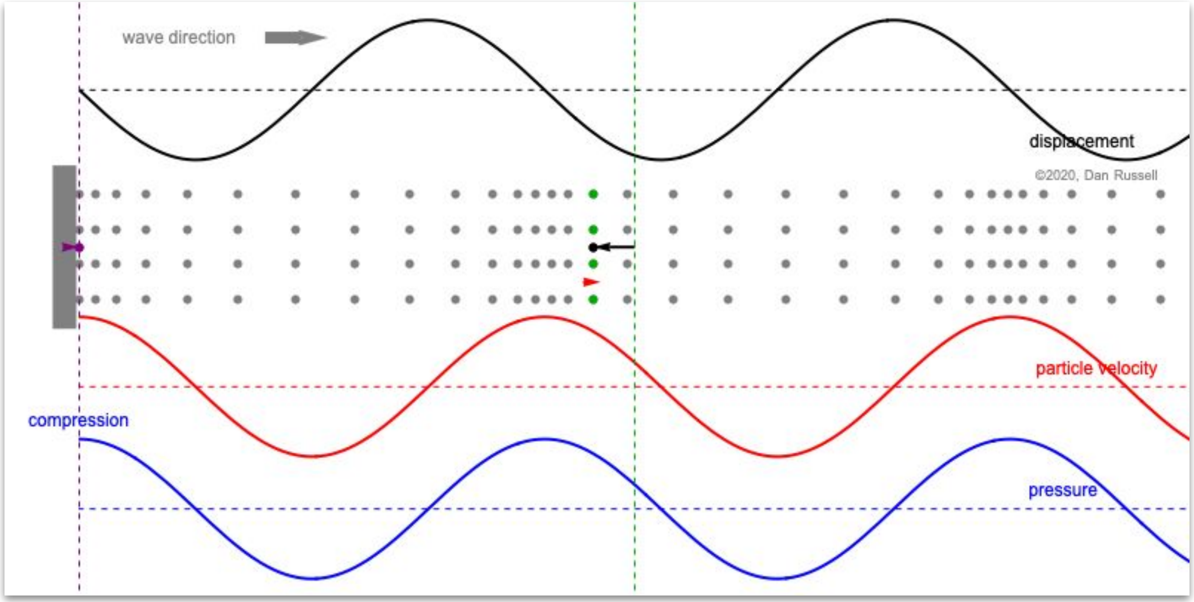
Compression ↦ “squished” ↦ higher pressure
Rarefaction ↦ “spread” ↦ low pressure

Air particles bounce back and forth at different frequencies: we observe a ‘wave’ of compression (and rarefaction) travelling through the air



https://www.acs.psu.edu/drussell/Demos/waves/wavemotion.html

Sound waves: displacement and pressure

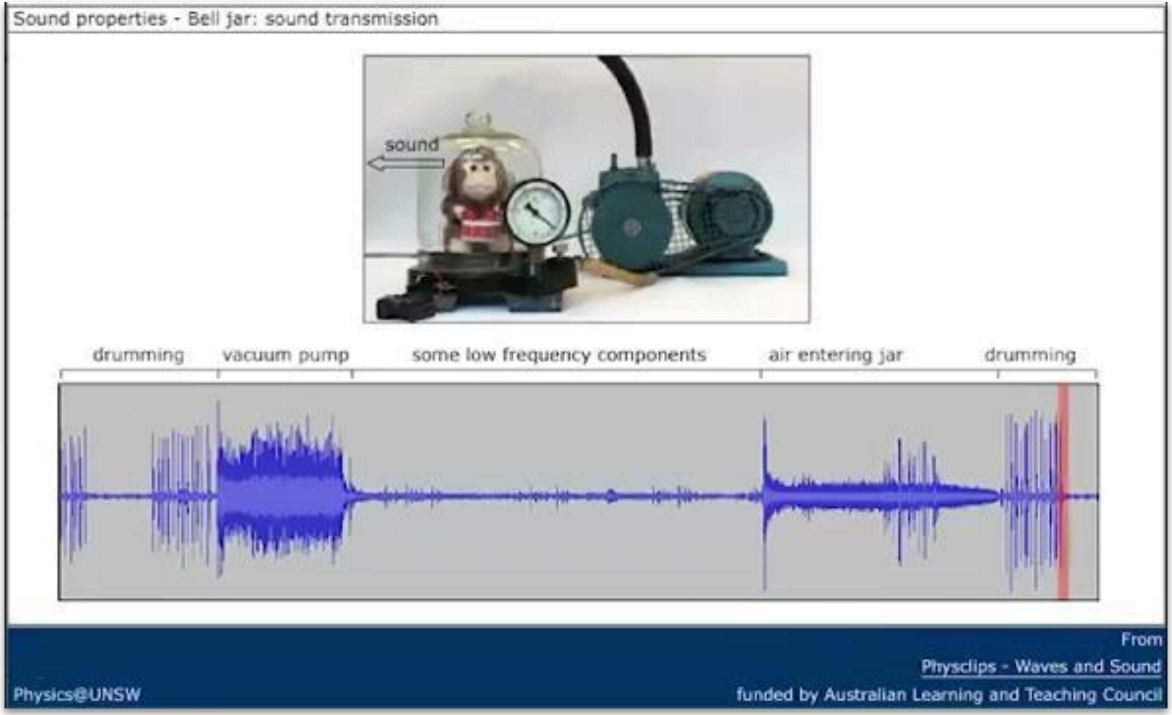


[Figure by Dan Russell's acoustics and vibration animations]

Air pressure is highest when the air particles are compressed (i.e. squished together). This produces a sinusoidal pattern as pressure at a single point changes in time.

Sound waves: (air) pressure

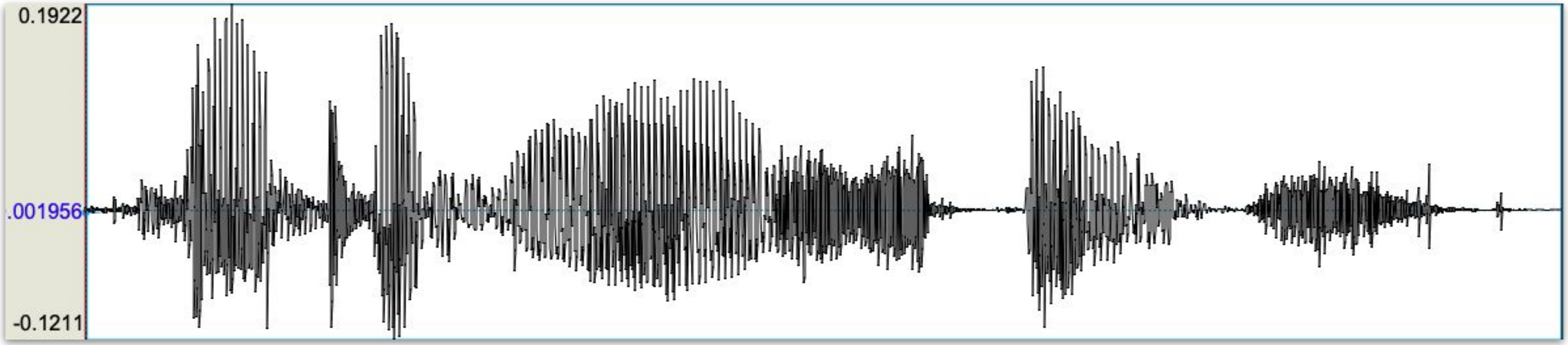
We generally characterise sound waves in terms of changes in pressure in a medium (usually air) caused by some physical source.



https://www.animations.physics.unsw.edu.au/waves-sound/sound/index.html

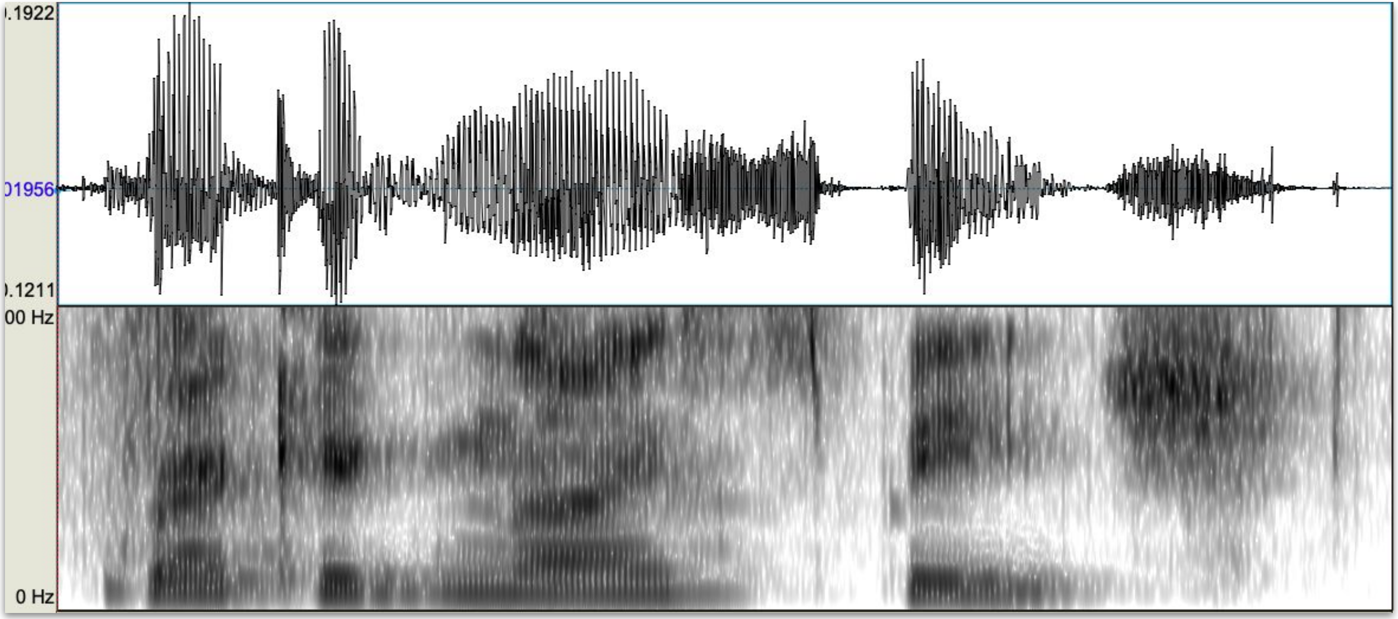
Speech in the Time Domain

Time domain: amplitude (measured pressure relative to atmospheric pressure) over time



Lab 1 learning outcome: It's very hard to determine difference vowels and consonants just from the time versus amplitude graph!

Spectrograms: The Frequency Domain through Time

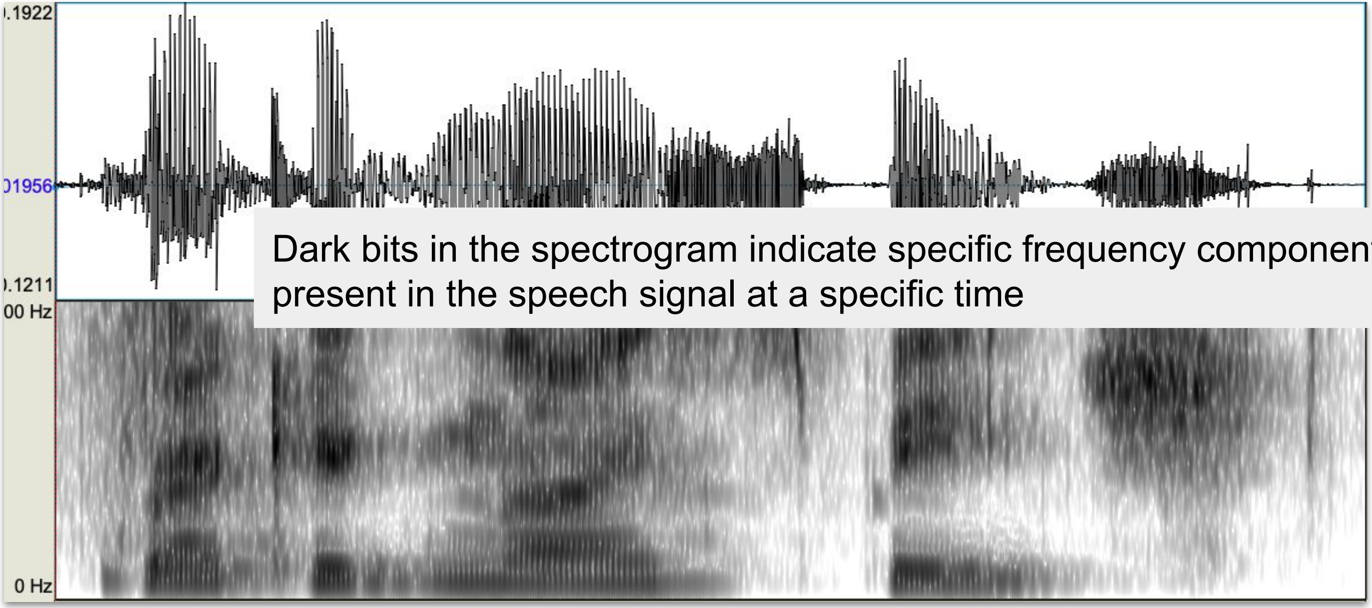


The spectrogram shows the **frequency characteristics** of the waveform through time. Each vertical bar represents frequencies present in a small window of time.

Spectrograms: Speech in the Frequency Domain

Amplitude

Frequency
(Hertz)



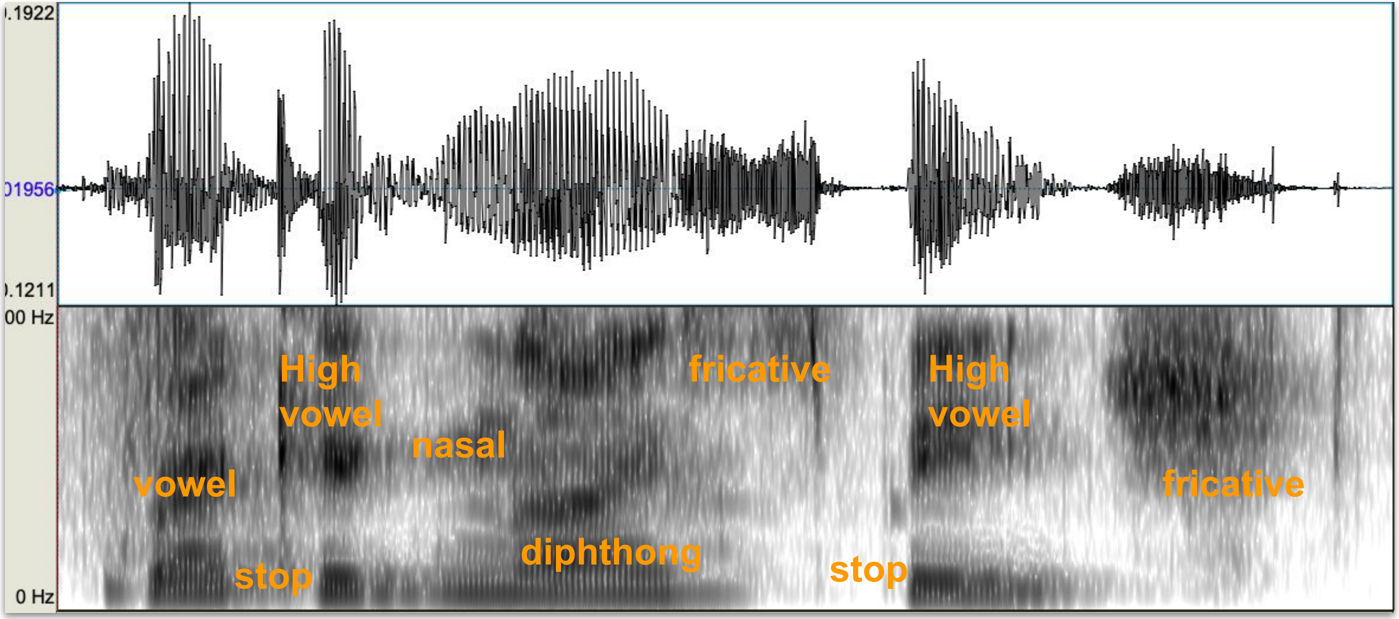
Dark bits in the spectrogram indicate specific frequency components are present in the speech signal at a specific time

Time (s)

Individual frequencies represent “pure tone” sine waves: e.g., 200 Hz, 300 Hz

Extra: Want to do more spectrogram reading? Check of the (no longer updated) monthly mystery spectrogram site: http://home.cc.umanitoba.ca/~robh/index.html

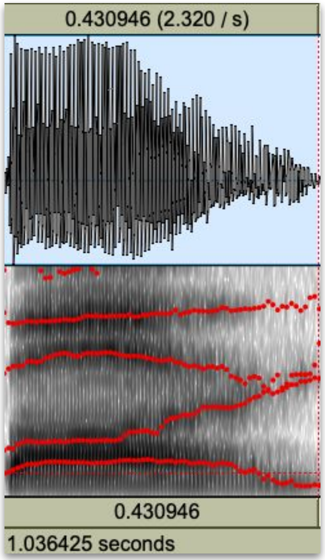
Viewing speech as a spectrogram



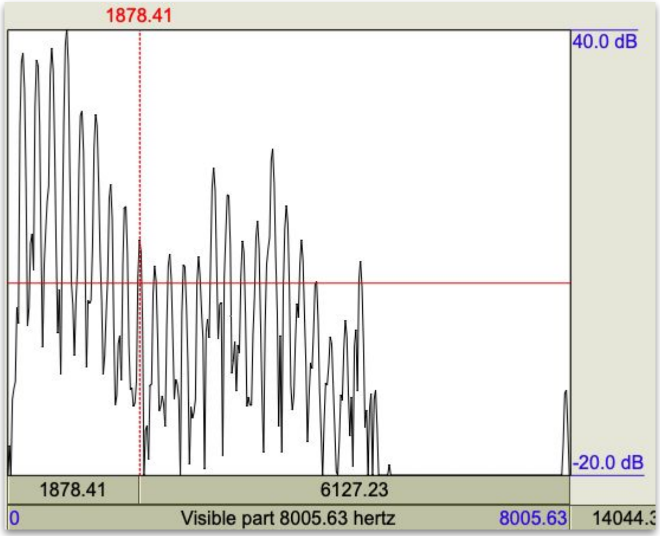
We recognise articulation in terms of frequency components of the sound wave over short periods of time → use this to learn mapping between words and acoustics

See Module 2 lab!

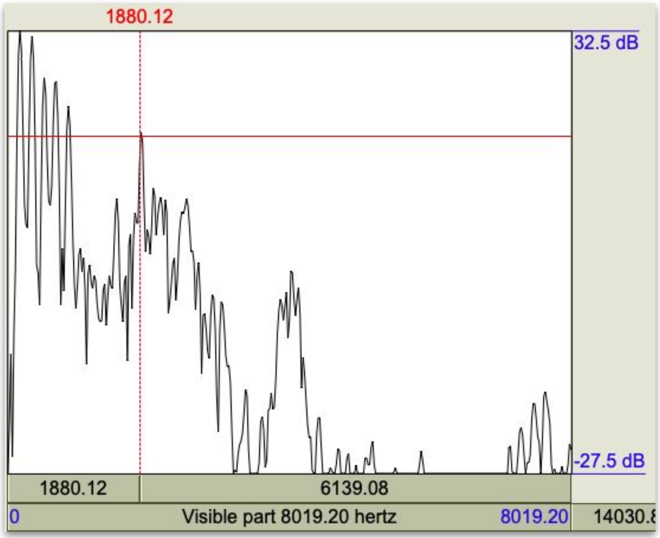
Spectral slices: moments in time



Spectrogram [ai]



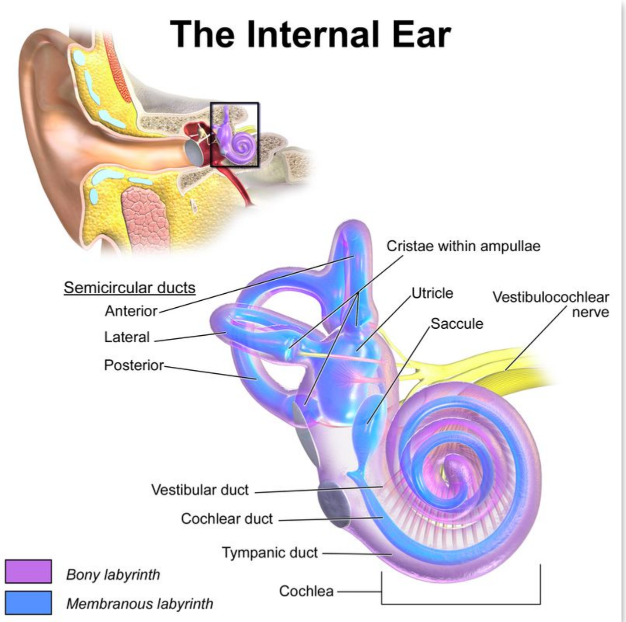
Spectrum [a]



Spectrum [i]

The overall shape of the spectrum (the spectral envelope) changes depending on articulator positions. But the the size (and shape) of the slice can change the shape of spectrum!

Computer Hearing?



- In the **human ear**, different parts of the cochlea are sensitive to sounds of different frequencies.
- Pressure fluctuations at different frequencies are detected and transmitted to the brain via electrical signals
- For a **computer**, we use the **Discrete Fourier Transform** to convert recordings from a time series of pressure amplitude measurements into frequencies
- But first we need to get the sounds into a representation the computer can understand!

(A bit more on human hearing later in the course…)

Digital Speech Signals

Digital sound waves

- Microphones capture changes in air pressure to record sound
- Converted into a continuous electrical signal: “Analogue”

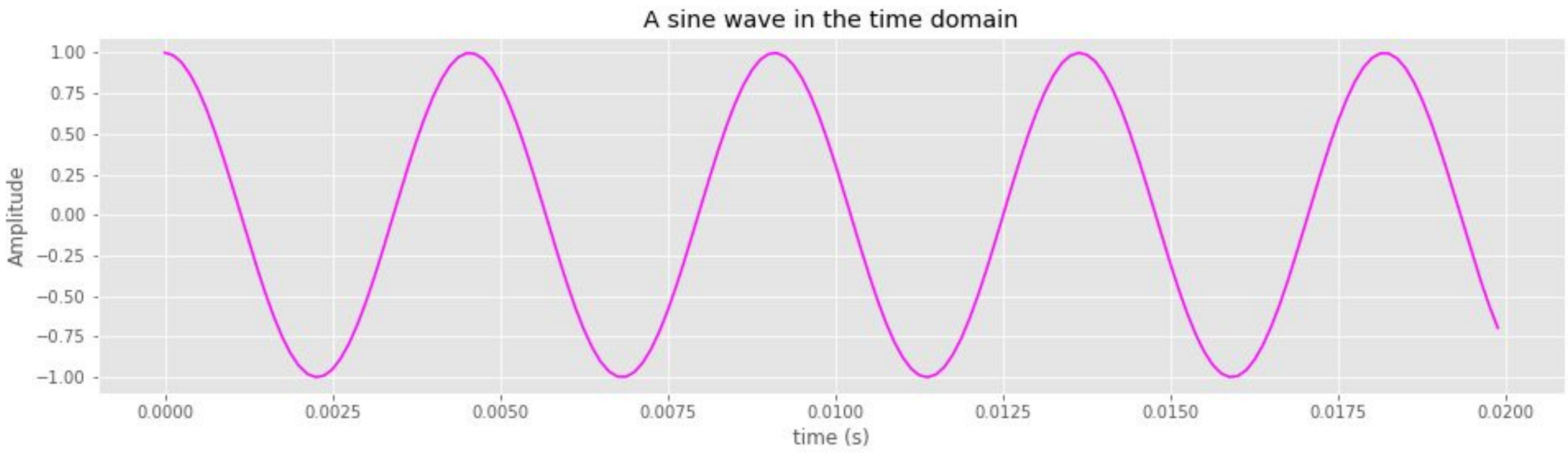
Problem: computers deal in discrete data: 1s and 0s (binary numbers)

We need to convert the continuous sound recording into a digital representation

→ We need to sample the wave and store amplitude values in binary

Analogue to digital conversion

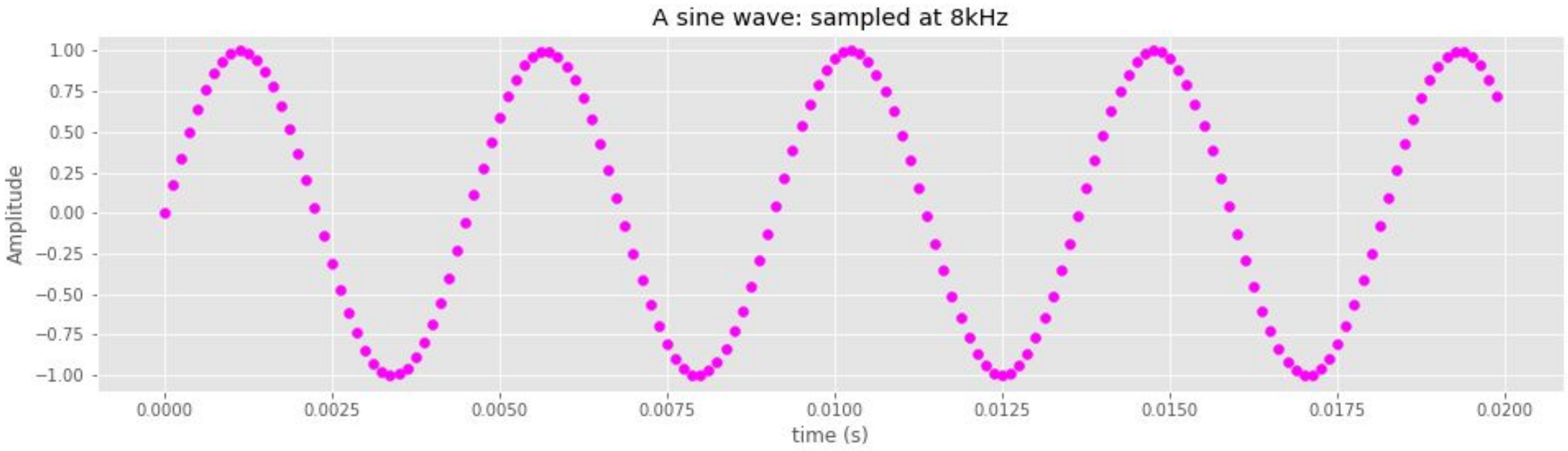
To process speech on a computer we need to convert a continuous signal into a series of discrete values



A representation of a continuous sound wave

Analogue to digital conversion: Sampling

The **sampling rate** (samples/second = Hz), aka sampling frequency, determines how often we record a value from wave



Sampling period = 1/sampling rate (seconds)

Sampling rate differences

- 16000 Hz
- 8000 Hz
- 4000 Hz
- 2000 Hz

Binary Representation

Computers represent and process information in terms of binary numbers:

- 1-bit: 0, 1 (2 values)
- 2-bit: 00, 01, 10, 11 (2x2 = 4 values)
- 3-bit: 000, 001, 010, 011, 100, 101, 110, 111 (2x2x2 = 8 values)
- …
- 16-bit: (2^16 = 65536 values)

The number of bits you can use determines the precision with which you can represent the signal

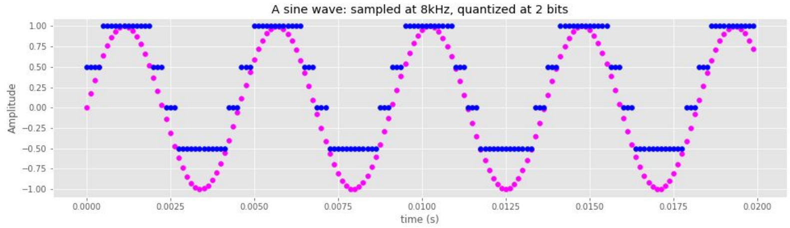
44100 Hz sampling rate: 16 bits 8 bits

Quantization

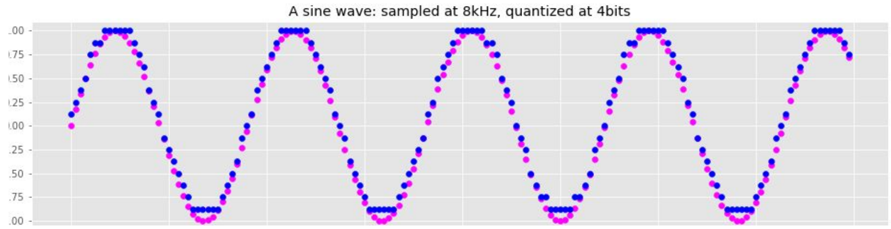
To give the waveform a binary representation, we need to map amplitudes to discrete bins. The number of bins determines how faithfully you can represent the wave

2 bits

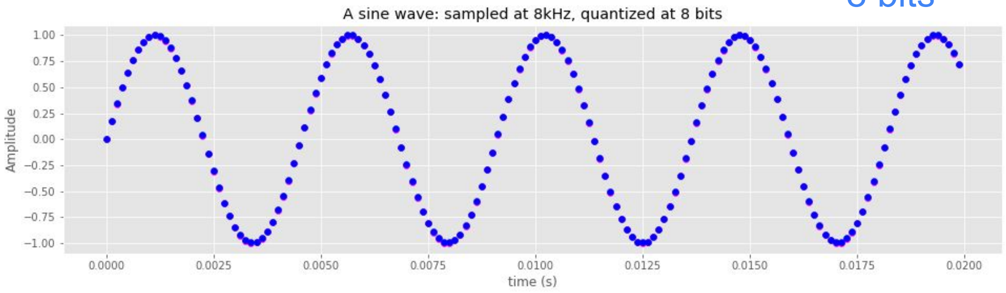
11
10
01
00



4 bits



8 bits



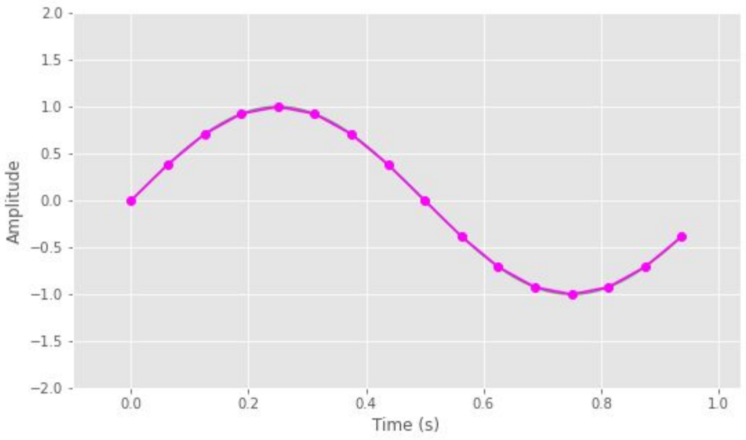
Note the small range in this example!
We need more bits if we want to capture a bigger dynamic range.

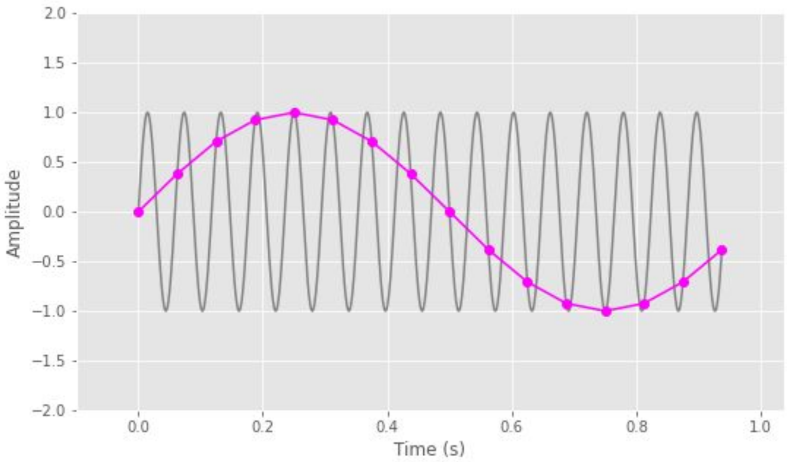
16 bits is usually ok!

More examples: https://dspillustrations.com/pages/posts/misc/how-does-quantization-noise-sound.html

Sampling and Aliasing

Frequencies above half the sampling rate (the **Nyquist Frequency**) will be indistinguishable from frequencies below the Nyquist frequency (i.e., the frequencies are *aliased* - you can't tell what they really are!)





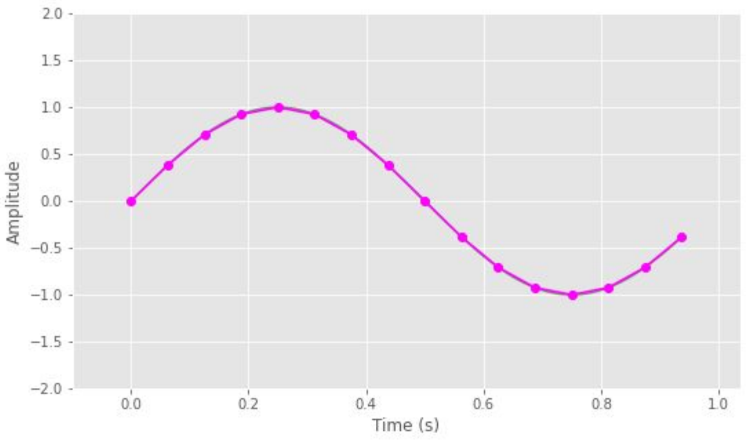
Question

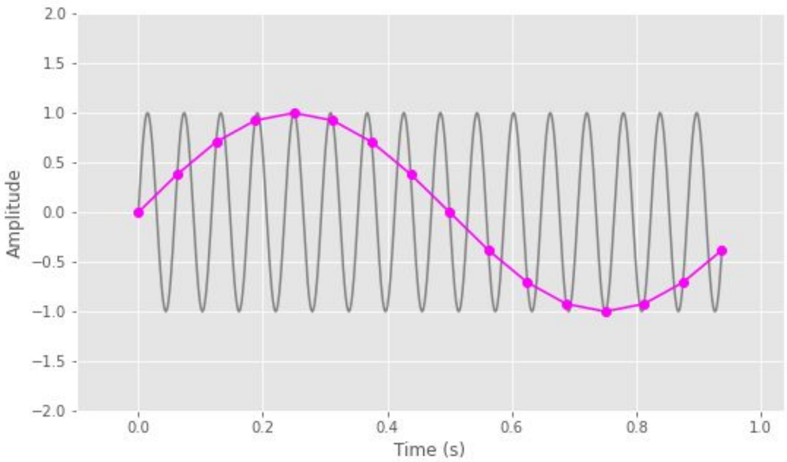
What happens if we have frequency components in our recording that are higher than the Nyquist Frequency?

e.g., if our sampling rate is 8000 Hz but the actual sound contains an 5000Hz component will it actually appear in our digitized recording? What problems might this cause?

Sampling and Aliasing

Frequencies above half the sampling rate (the **Nyquist Frequency**) will be indistinguishable from frequencies below the Nyquist frequency (i.e., the frequencies are *aliased* - you can't tell who they really are!)





To be sure of our frequency analysis we first need to filter out high frequencies

More on filters next week!

Generating Spectrograms

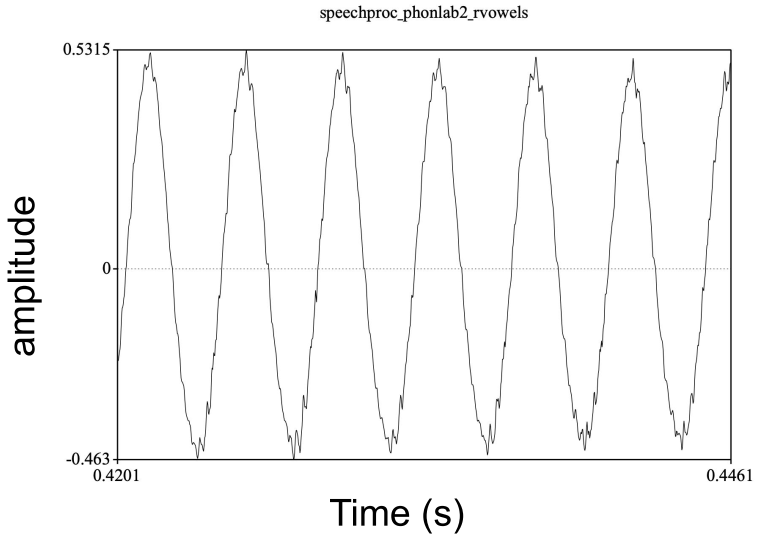
- Recording of sound
  - Filtering (e.g. frequencies above the desired Nyquist Frequency)
- Digitization
  - Sampling (sampling rate)
  - Quantization (bit depth)
  - A discrete representation in the time domain
- Discrete Fourier Transform (windowed)
  - Maps from time domain to frequency domain
  - Applied to short windows of speech
  - Outputs magnitude and phase spectrum

Spectrogram: time vs frequency 'heatmap', where colour (darkness in Praat) corresponds to the 'strength' of different frequencies component in the signal.

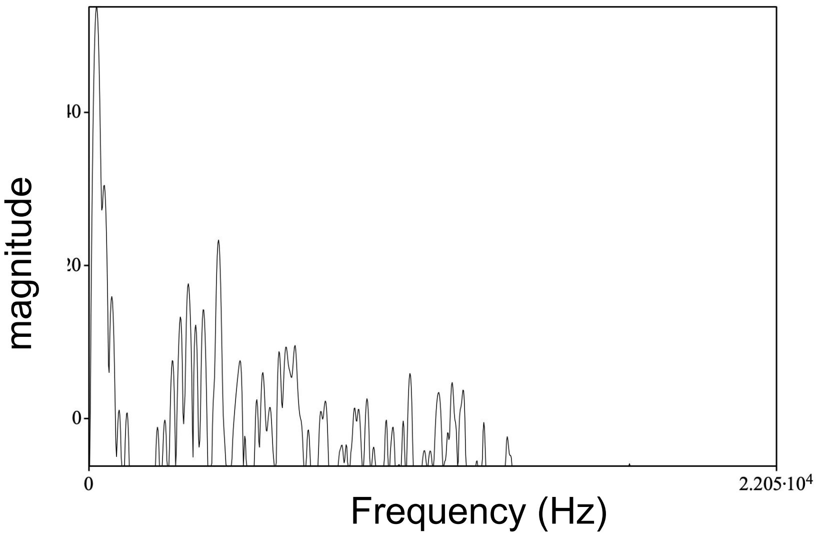
The Discrete Fourier Transform

Discrete Fourier Transform

The Discrete Fourier Transform (DFT) is mathematical procedure we can use to determine the frequency content of a discrete signal sequence



Time domain



Frequency domain

Discrete Fourier Transform

“Wasn’t the Fourier Transform about sine waves???”

A mathematical procedure we can used to determine the frequency content of a discrete signal sequence

Mathematical view: for input x[n] with n=0,...,N-1 (N inputs)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi n}{N}k}$$

For k=0,..,N-1 (N analysis frequencies)

Discrete Fourier Transform

An equivalent formulation of the DFT using sines and cosines

A mathematical procedure we can used to determine the frequency content of a discrete signal sequence

Mathematical view: for input x[n] with n=0,...,N-1 (N inputs)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] \left[\cos\left(\frac{2\pi n}{N} k\right) - j \sin\left(\frac{2\pi n}{N} k\right) \right]$$

For k=0,..,N-1 (N analysis frequencies)

Derived from Euler's Formula

You don't have to memorize this equation! But we will try to develop the intuition behind it…

Periodic function

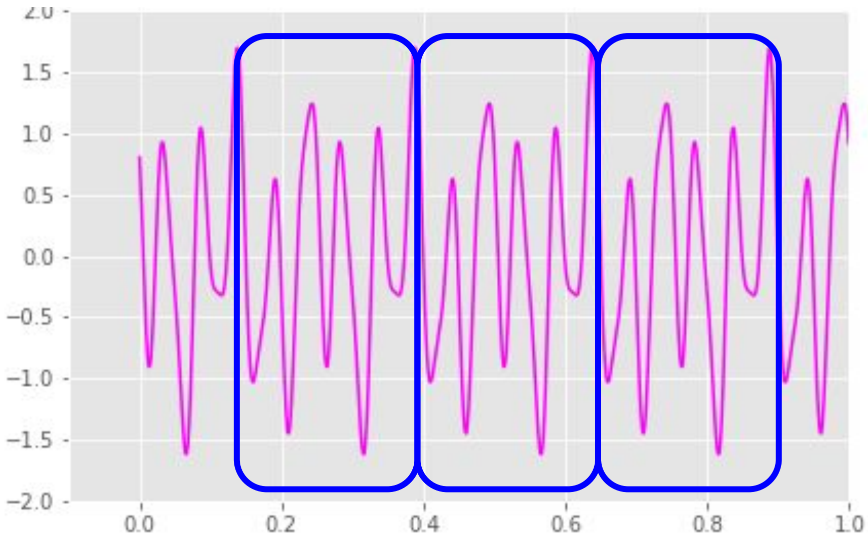
A **periodic** function repeats in time

A more formal way of saying it:
For function f which takes a time t as input, the output obeys:

$$f(t) = f(t + nT)$$

for some constant T (the period), for all times t and integers n

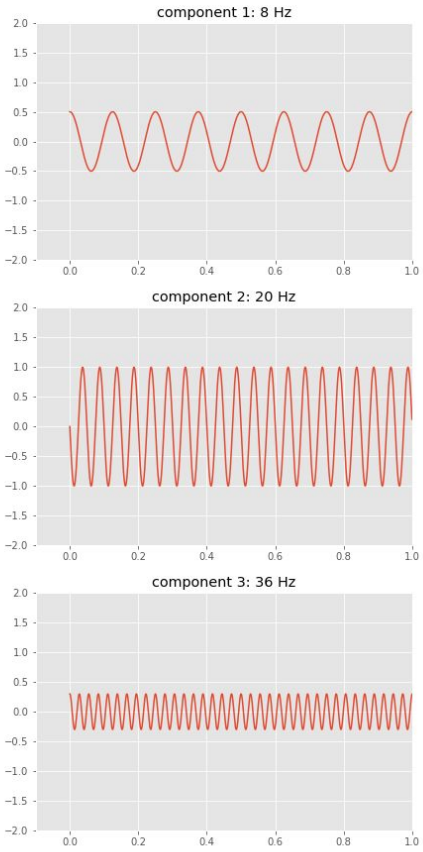
The function outputs the same pattern over and over again, predictably through time

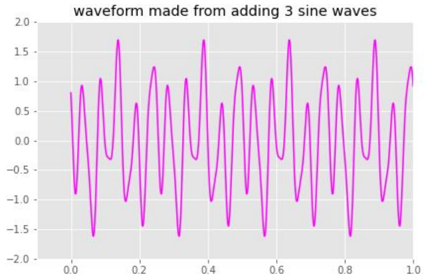


Fourier Analysis

A **periodic** function repeats in time

A periodic function can be written as **a discrete sum** of simple periodic (sinusoidal) functions (i.e. sine and cosine) of different frequencies





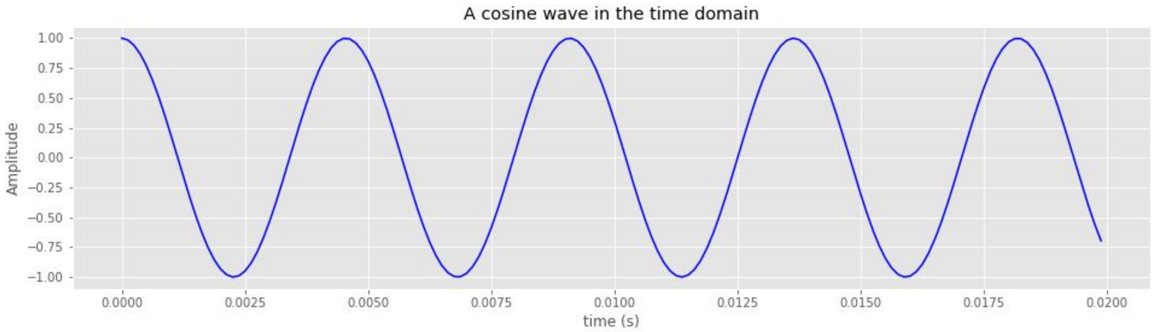
We can construct complex waveform by adding together simple periodic functions (sinusoids) with some scaling and shifting

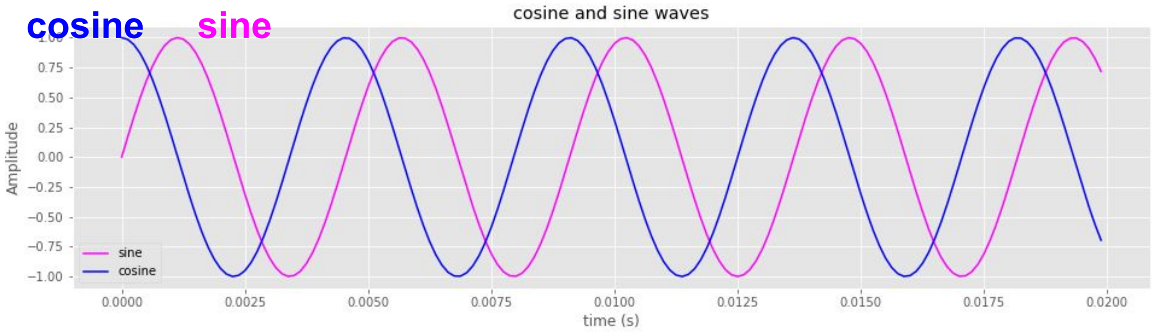
Cosine and sine functions

These are **simple** periodic functions. If we play them they produce “**pure tones**”

200Hz

300Hz





Think of a sine wave as a shifted cosine wave and vice versa

Fourier Analysis Demo

http://www.falstad.com/fourier/Fourier.html

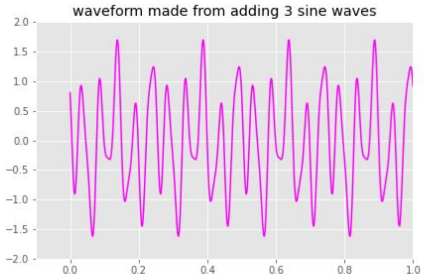
Question: How many sine waves do you need to make a square wave?

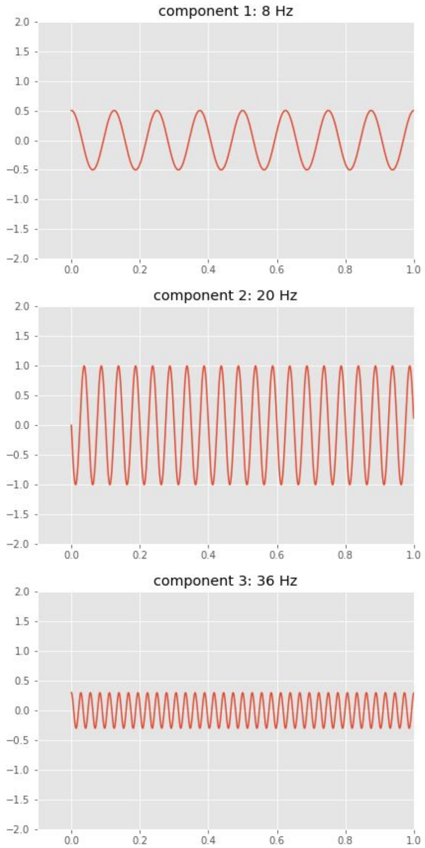
Fourier Transform

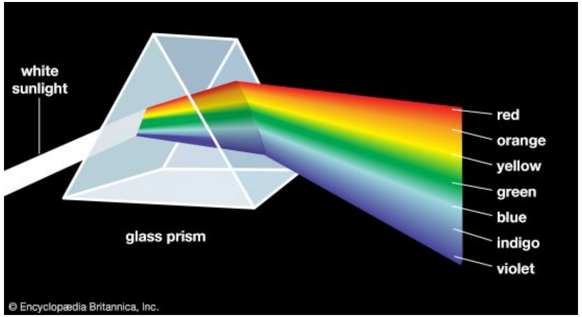
We can **decompose** a periodic waveform into a set of **simple periodic functions** (i.e., pure tones) of different frequencies.

Just like a prism splits light into component colours

A spectrum of colours!



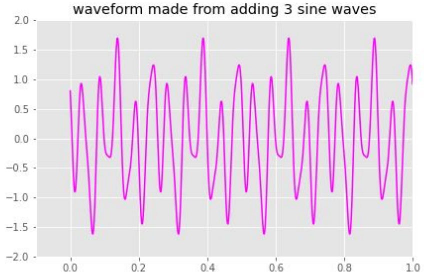


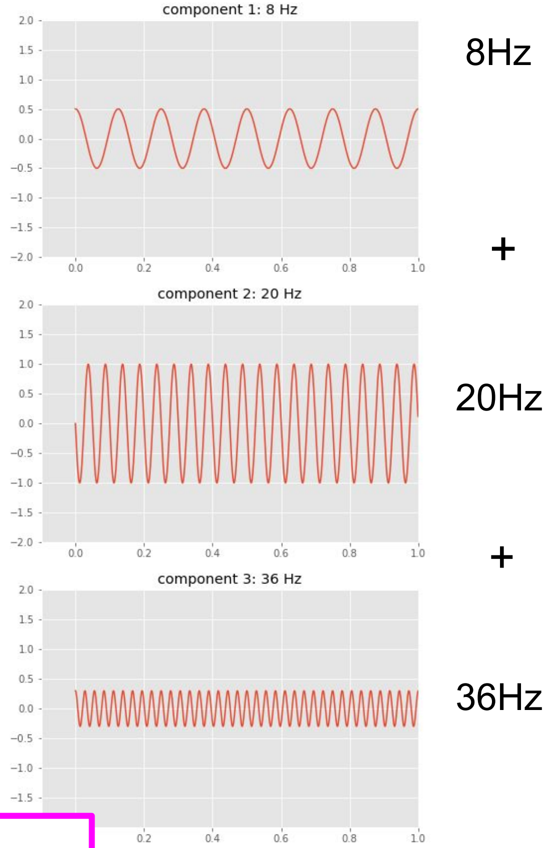


Fourier Transform

We can **decompose** a periodic waveform into a set of **simple periodic waves** (i.e., pure tones) of different frequencies.

If we **scale** and **shift** those pure tones appropriately we can approximate the original waveform by adding the scaled and shifted waves together

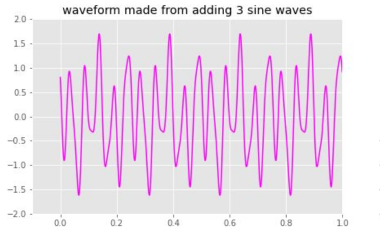


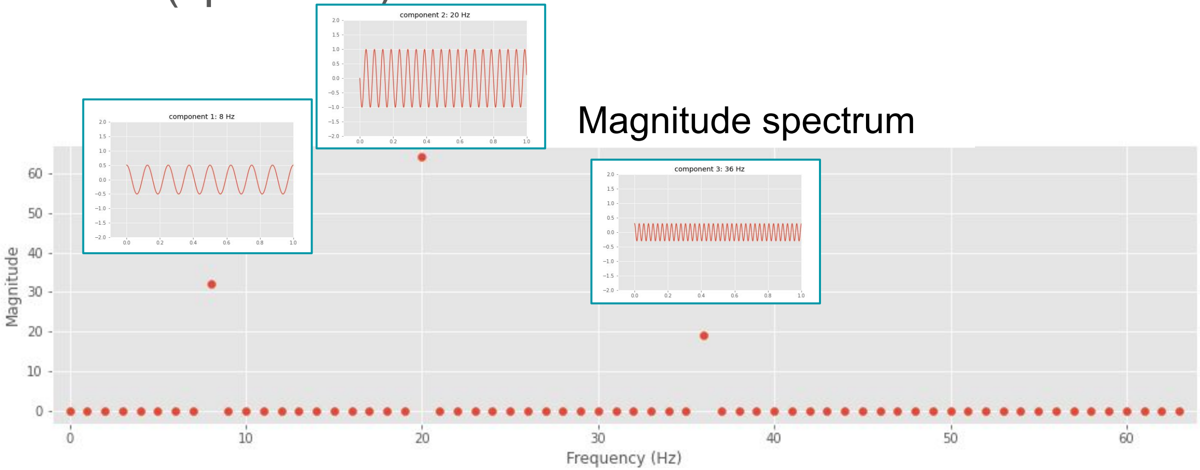


$$0.5\cos(2\pi.8t) + \cos(2\pi.20t + \pi/2) + 0.3\cos(2\pi.36t)$$

Fourier Transform

The Fourier Transform provides us with the “technology” to map between the time domain to the frequency domain (spectrum)





- It decomposes the time series waveform into component frequencies
- Non-zero magnitudes indicate that you would include that frequency in reconstructing the signal

Discrete Fourier Transform

- Input: a sequence of N values
  - e.g. amplitude values sampled in time
- Output: N complex numbers
  - Correspond to N sinusoids with frequencies spread between 0 and the sampling rate
  - The output coefficients tell us how to scale and shift the corresponding sinusoids so we can reconstruct the original input

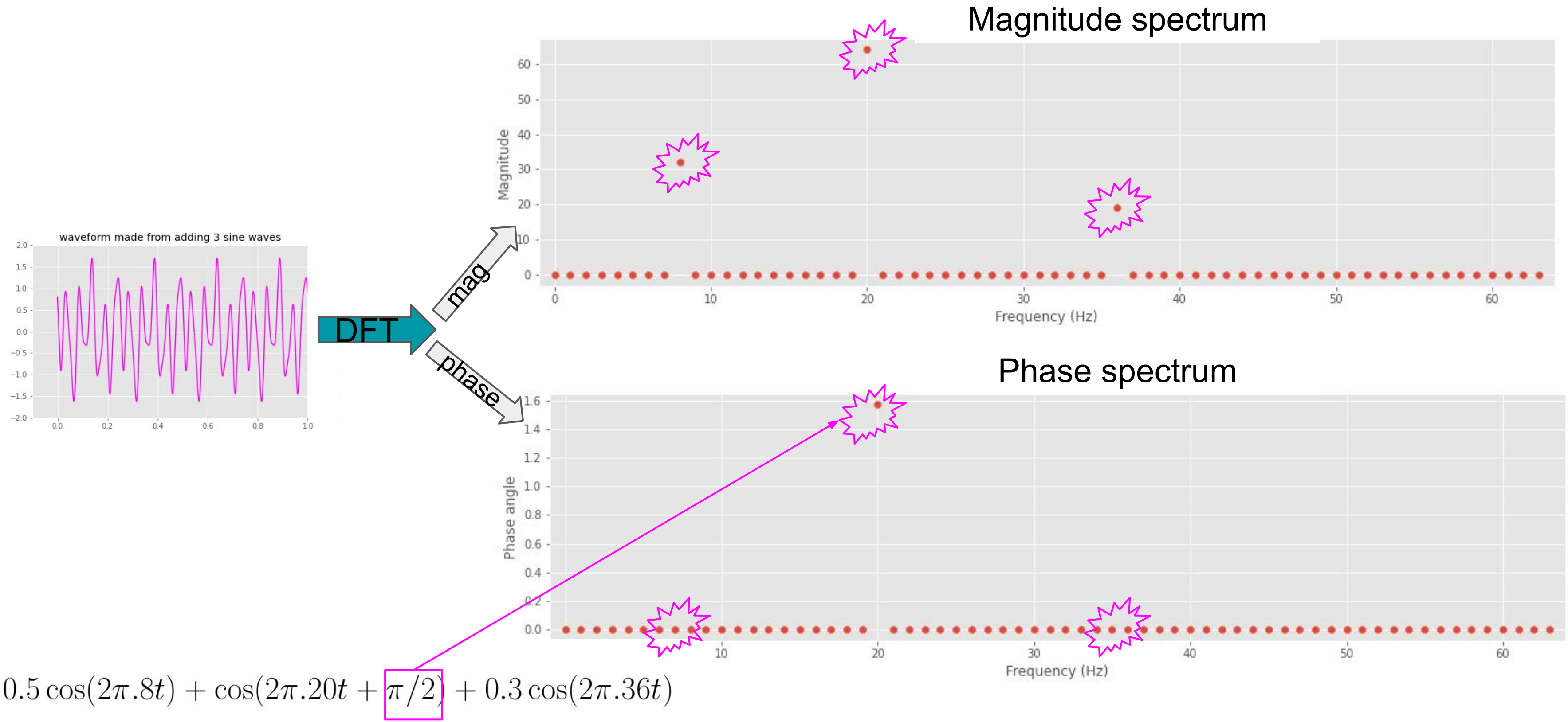
Discrete Fourier Transform Outputs

The output coefficients tell us how to scale and shift the corresponding sinusoids so we can reconstruct the original input

The complex number outputs can be interpreted in terms of:

- The **magnitude spectrum**: how much to **scale** the different pure tone frequency components
- The **phase spectrum**: how much to **shift** the different pure tone frequency components

DFT output as magnitude and phase

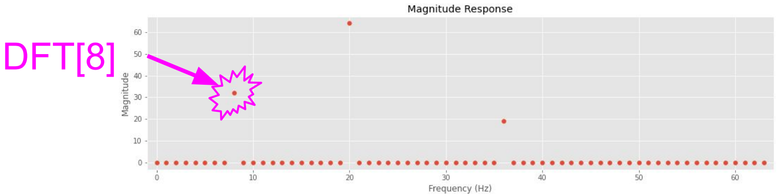


$$0.5\cos(2\pi.8t) + \cos(2\pi.20t + \pi/2) + 0.3\cos(2\pi.36t)$$

Questions: Spectral Slices

The spectral slice function in Praat performs the DFT on a selected window of speech.

- What part of the DFT output does the spectral slice show us?
- What frequencies can we view in a Praat spectral slice?
- How does the size of the input window change the spectral slice?
- How does input size relate to wide and narrowband spectrograms?



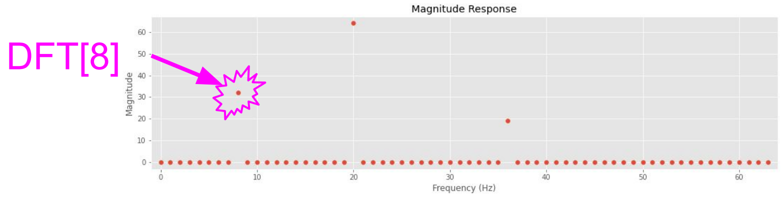
How does it work?

Let’s call our input sequence x and the sinusoid associated with DFT[k], s_k

- The DFT[k] is the **inner product** x and s_k (notation: $<x, s_k>$, aka dot product)
    - You can interpret this as similarity or, very loosely, as correlation (but it’s not a statistical property here)
- The sinusoids we are considering form an **orthogonal basis**:
    - The inner product of two of these sinusoids is non-zero only if their frequencies are the same

So, roughly, the inner product $<x, s_k>$ picks out only bits of the input that have the same frequency as the sinusoid s_k (if so, with what scale and shift). If there is no periodic component with that frequency the output DFT[k] will be zero.

Extra: See Module 3 Lab extension notebooks for an example in gory detail.



How does it work?

Let's call our input sequence $\boldsymbol{x}$ and the sinusoid associated with DFT[k], $\boldsymbol{s_k}$

- Calculate the similarity between DFT[k] and input x
  - i.e., take the **dot product** of $\boldsymbol{x}$ and $\boldsymbol{s_k}$ (notation: $\langle \boldsymbol{x}, \boldsymbol{s_k} \rangle$, aka inner product)
  - Multiply the equivalent points in time for $\boldsymbol{x}$ and $\boldsymbol{s_k}$, the add it all up
- This measure tells us whether the input include a frequency component with the frequency as the sinusoid $\boldsymbol{s_k}$ (possibly scaled and shifted)
  - If sk is not present the output DFT[k] will be zero.

Extra: See Module 3 Lab extension notebooks for an example in gory detail.

DFT Analysis frequencies

- For N input values, we get N output analysis frequencies spread evenly between 0 and the sampling rate f_s:

$$Freq(DFT[k]) = \frac{k f_s}{N}$$

- This formulation ensure the analysis sinsuoids form an **orthogonal basis** since we are dealing with sampled sinuisoids.

What frequencies?

The sinusoids associated with DFT outputs have frequencies corresponding to represent N values spread evenly between 0 and the sampling rate f_s:

$$Freq(DFT[k]) = \frac{k f_s}{N}$$

- These are frequencies that complete a whole number of periods in the input window time.
- But we only use the first half of those outputs for analysis. Why?

Questions

If our input window is 100 samples how many DFT outputs will the DFT have?

What frequencies will the DFT outputs represent?

Questions

If our sampling rate is 8000 Hz and our input analysis window (frame) contains 80 samples

- How many DFT outputs will we get?
- What frequencies are represented by the DFT output (i.e., the magnitude spectrum)?
- What frequencies in the input signal will we actually be able to detect?

Questions

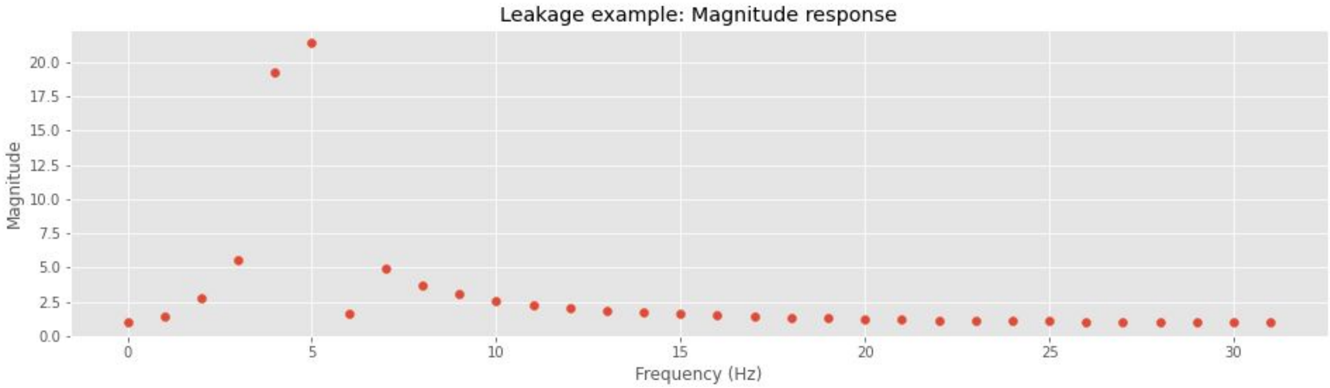
If our sampling rate is 16000 Hz and our analysis window (frame) is 25 ms

- How many DFT outputs?
- What frequencies can we detect?

Leakage

What happens if the input frequency falls between the outputs? Leakage!

Positive magnitudes for the DFT outputs near the actual input frequency (try it in the lab!)

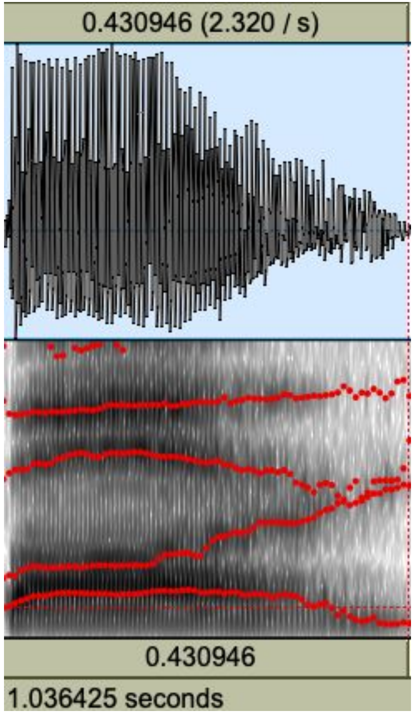


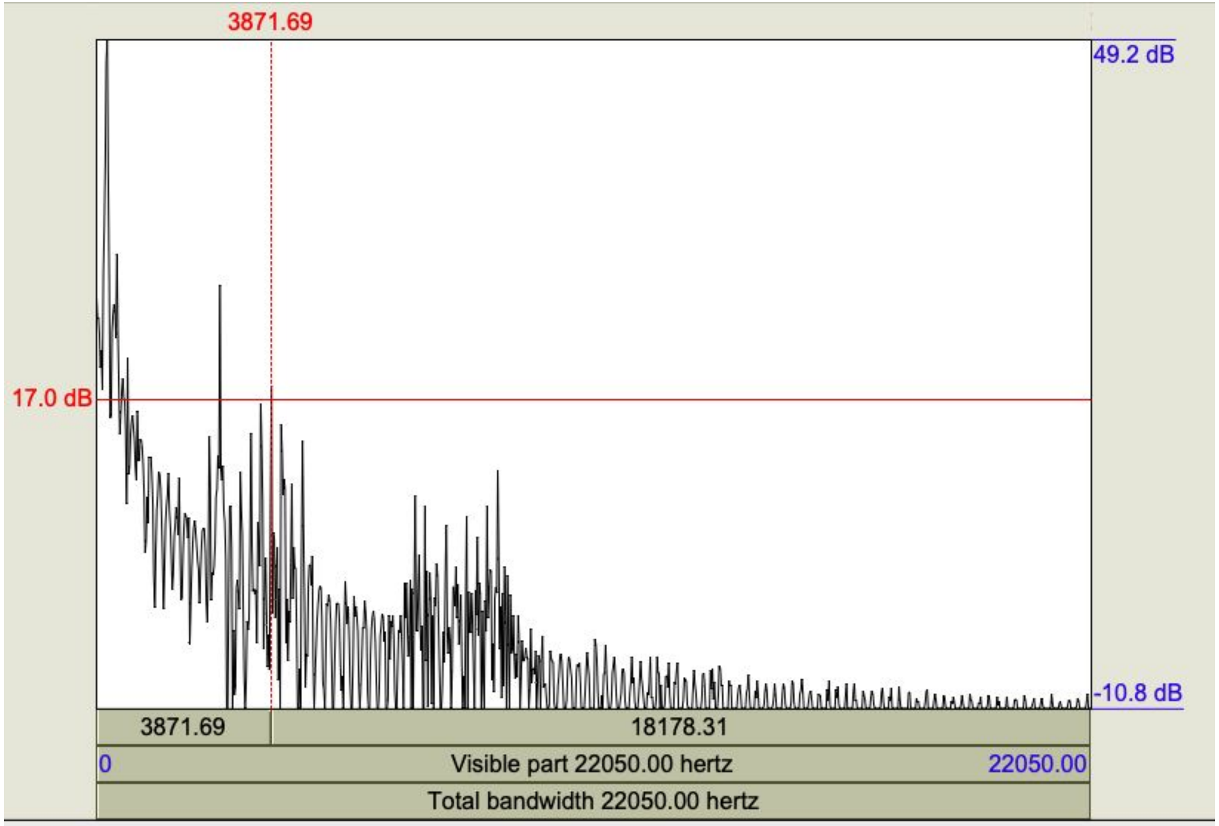
If we want to be able to analyse lots of frequencies, we need a lot of input values

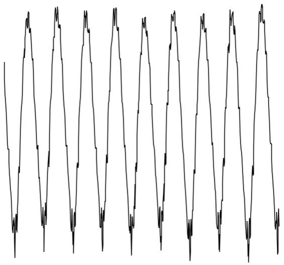
From DFT to Spectrogram

Spectrogram is a series of DFTs in time: it creates a time-series of frequency domain features

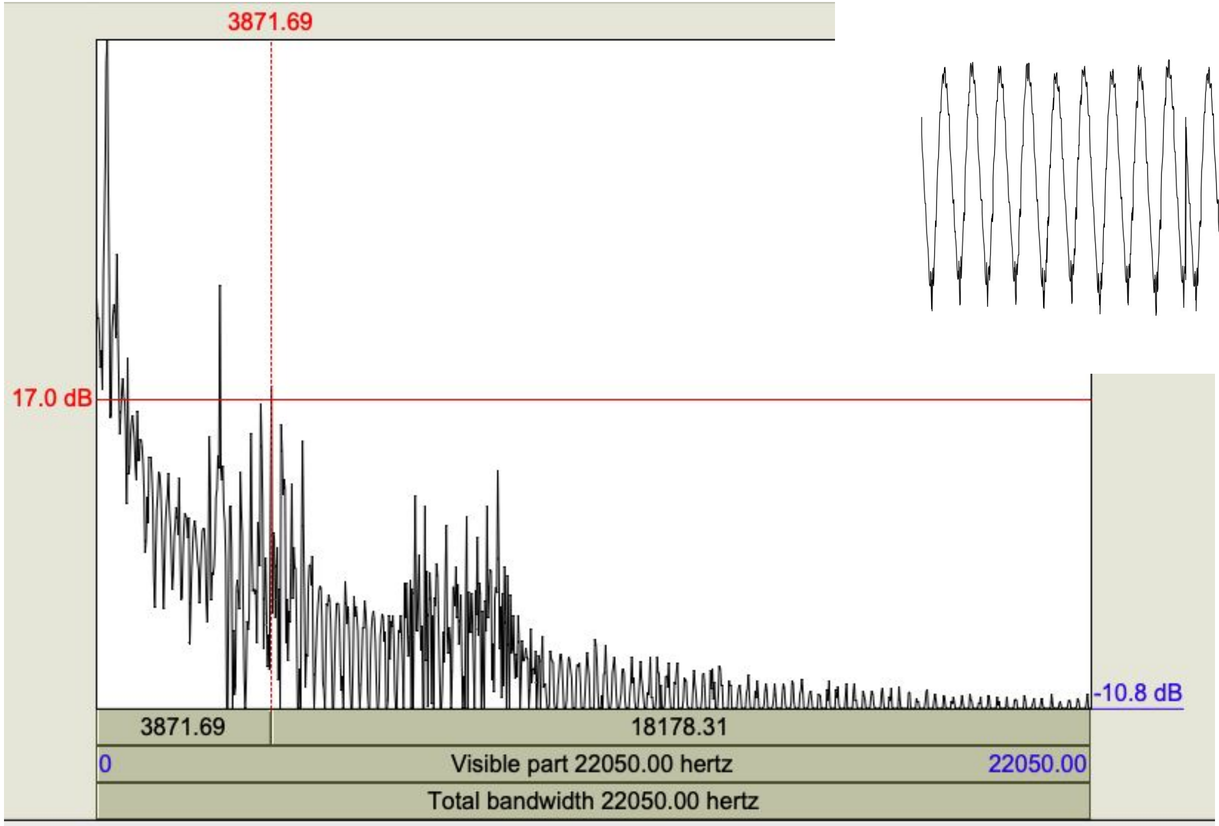
- DFT maths assumes that a signal continues forever in time
- Real world signals are (sort of) locally periodic
- So, we perform the DFT on **short** regions (**windows**/analysis **frames**) in the signal, i.e., the Short Time Fourier Transform (STFT)
- The type of **window** can change the output!

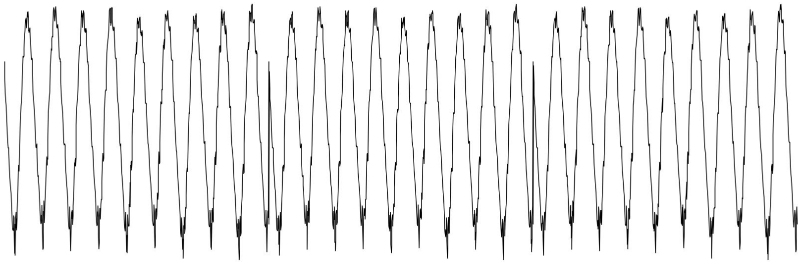






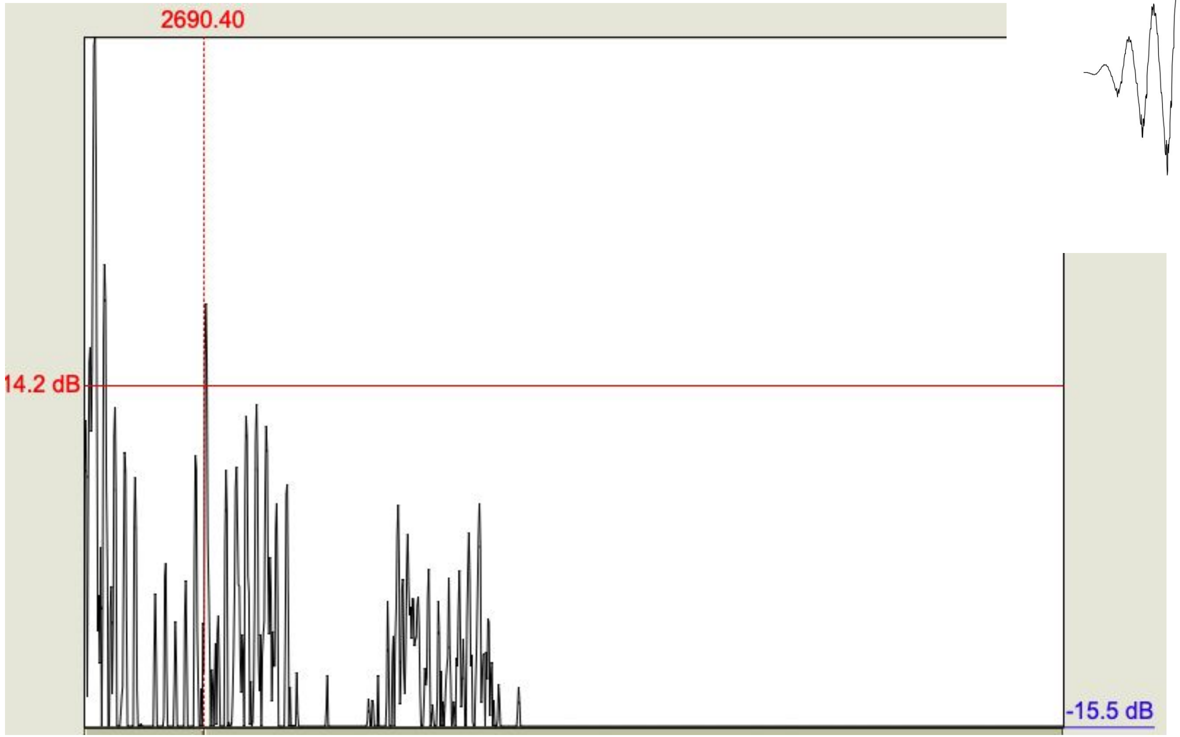
Window with abrupt ending (rectangular window)

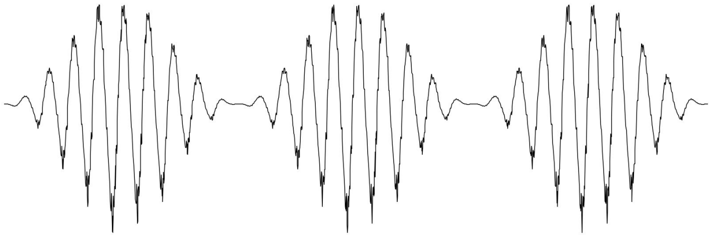




Artifacts due to discontinuity at edges of the window

Spectrum shows positive mag across frequencies→ leakage





We can reduce artifacts due to discontinuitues by using a tapered window, e.g. Hanning, instead of a plain rectangle

With the Hanning window, the spectral characteristics are sharper, less leakage!

Extension:
Understanding the DFT equation

Discrete Fourier Transform

This is what you'll see in textbooks and computing packages - see the Module 3 lab!

A mathematical procedure we can used to determine the frequency content of a discrete signal sequence

Mathematical view: for input x[n] with n=0,...,N-1 (N inputs)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi n}{N}k}$$

For k=0,..,N-1 (N analysis frequencies)

Discrete Fourier Transform

An equivalent formulation of the DFT using sines and cosines

A mathematical procedure we can used to determine the frequency content of a discrete signal sequence

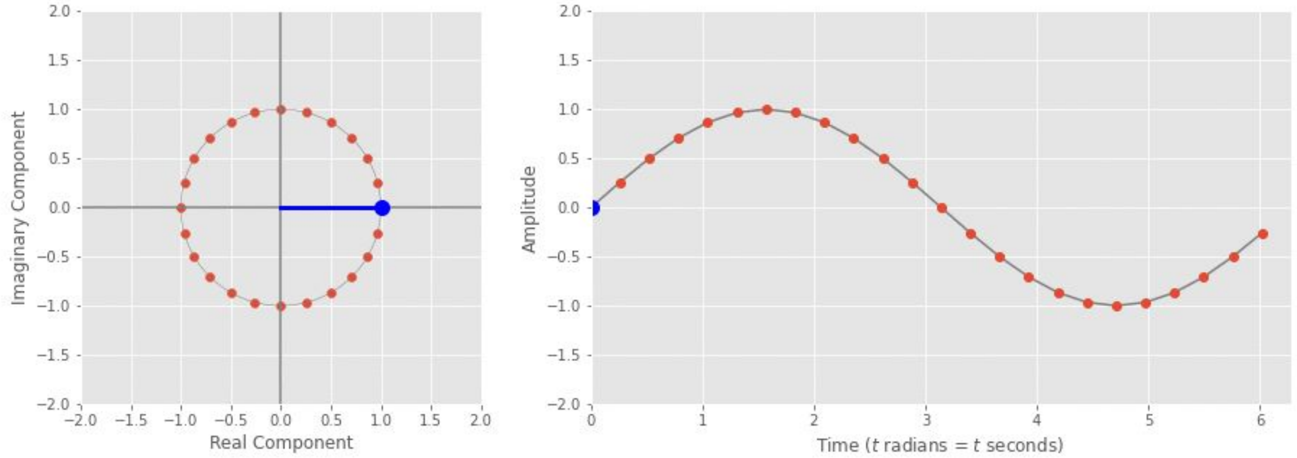
Mathematical view: for input x[n] with n=0,...,N-1 (N inputs)

$$\text{DFT}[k] = \sum_{n=0}^{N-1} x[n] \left[\cos\left(\frac{2\pi n}{N} k\right) - j \sin\left(\frac{2\pi n}{N} k\right) \right]$$

For k=0,..,N-1 (N analysis frequencies)

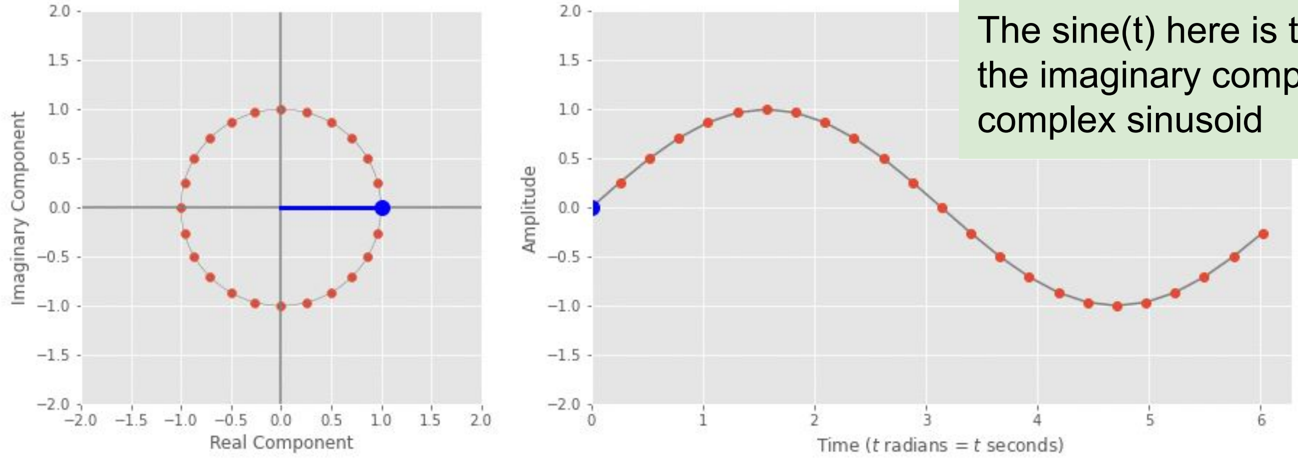
Derived from Euler's Formula

A different view of periodicity



You can see think of a sine wave as the vertical projection of a vector rotating at a constant speed drawing out a circle (counter-clockwise). **A period** is characterised by one complete 360 degree rotation (i.e., cycle).

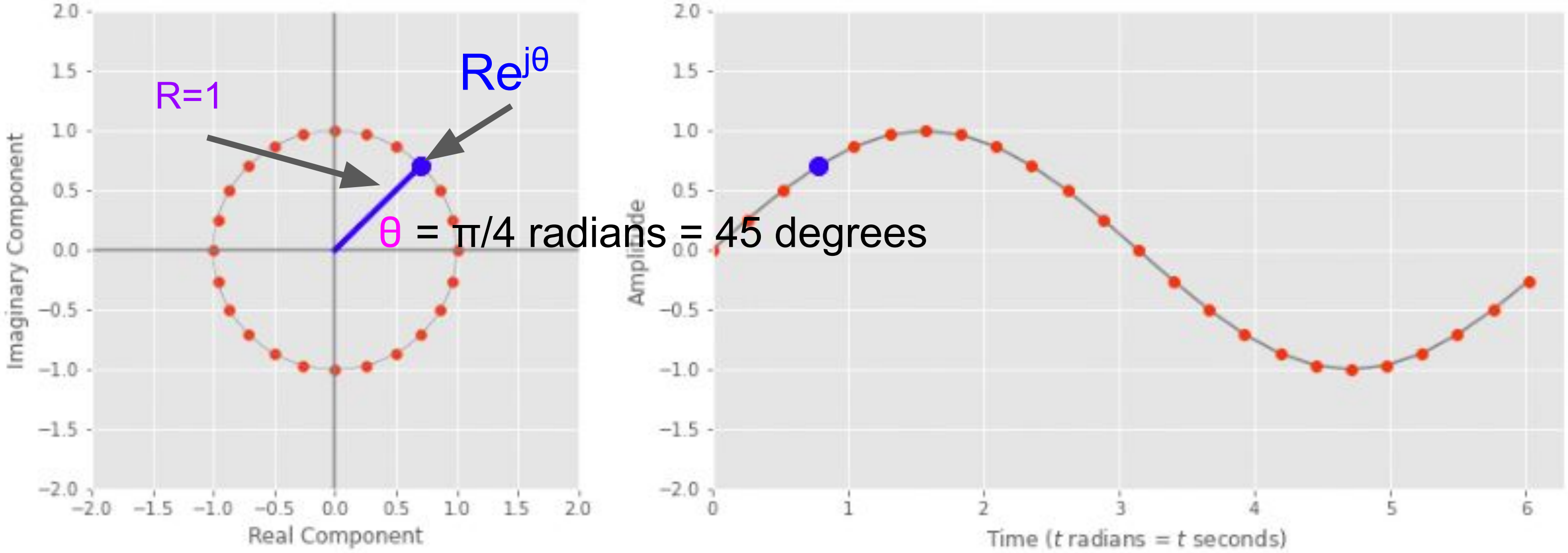
A different view of periodicity: Complex Sinusoids



The sine(t) here is the projection of the imaginary component of the complex sinusoid

The rotating vector (on the left) is a **complex sinusoid**. It lives in the complex plane! We describe points on the circle as $Re^{j\theta}$, where $j=\sqrt{-1}$ is an **imaginary number**

A different view of periodicity: Complex Sinusoids



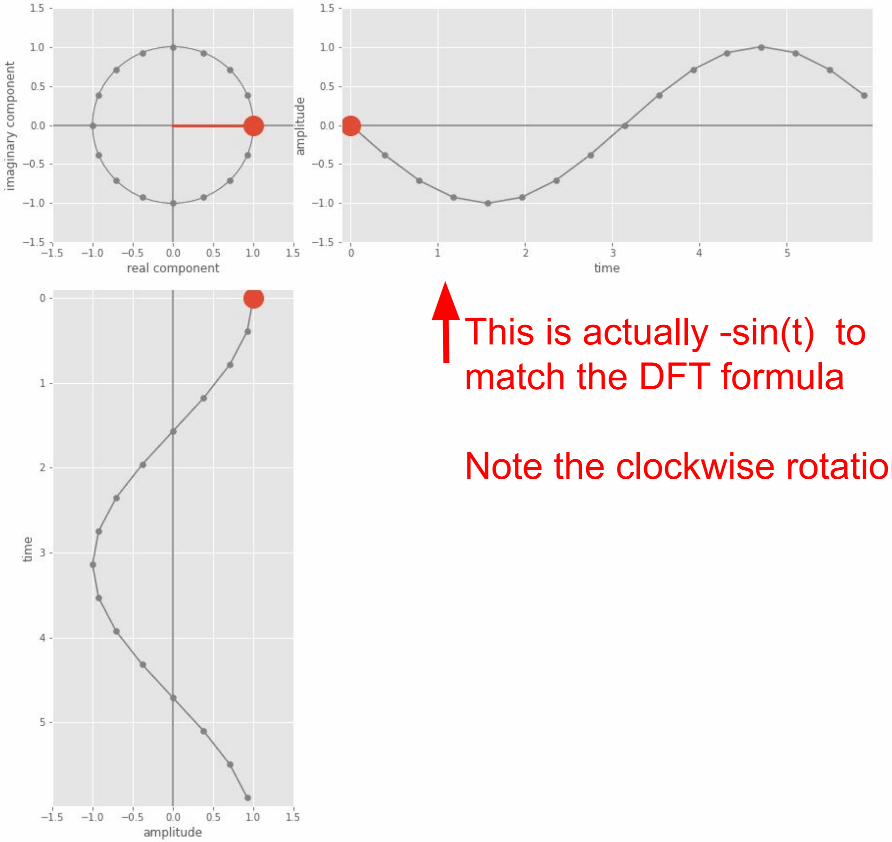
We describe points on the circle as $Re^{j\theta}$ where R describes the magnitude of the vector and θ describes the angle of rotation (i.e., the phase) from (1,0)

Sine and cosine

We now define sine and cosine in terms of the vector rotation

- **Sine** is the vertical projection of the rotating vector
- **Cosine** is the horizontal projection of the rotating vector

Infinite repetition in a finite space!



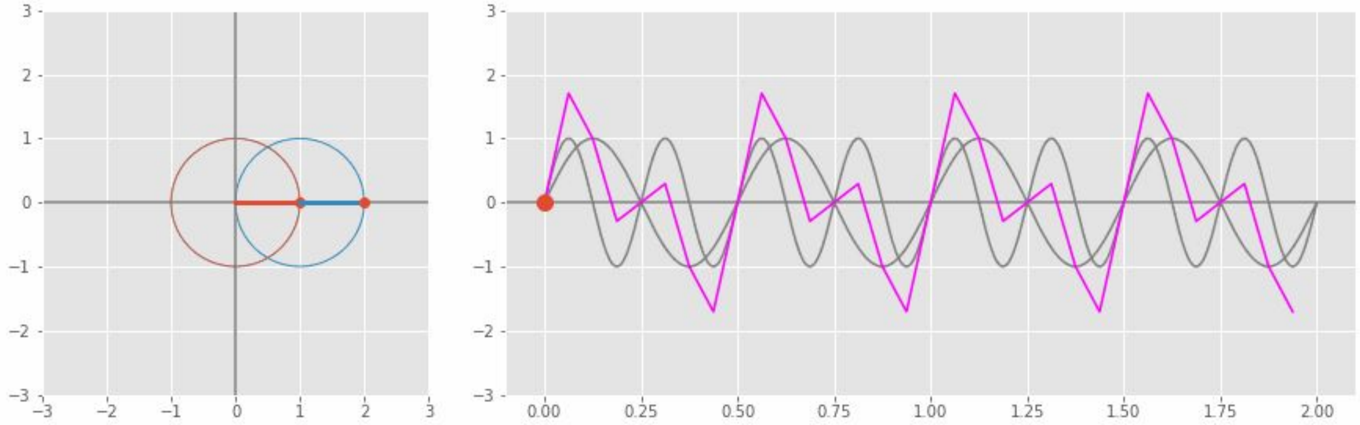
This is actually -sin(t) to match the DFT formula

Note the clockwise rotation

Adding sinusoids: Superposition

We can add complex sinusoids in the same way as we add simple sine waves together (time wise).

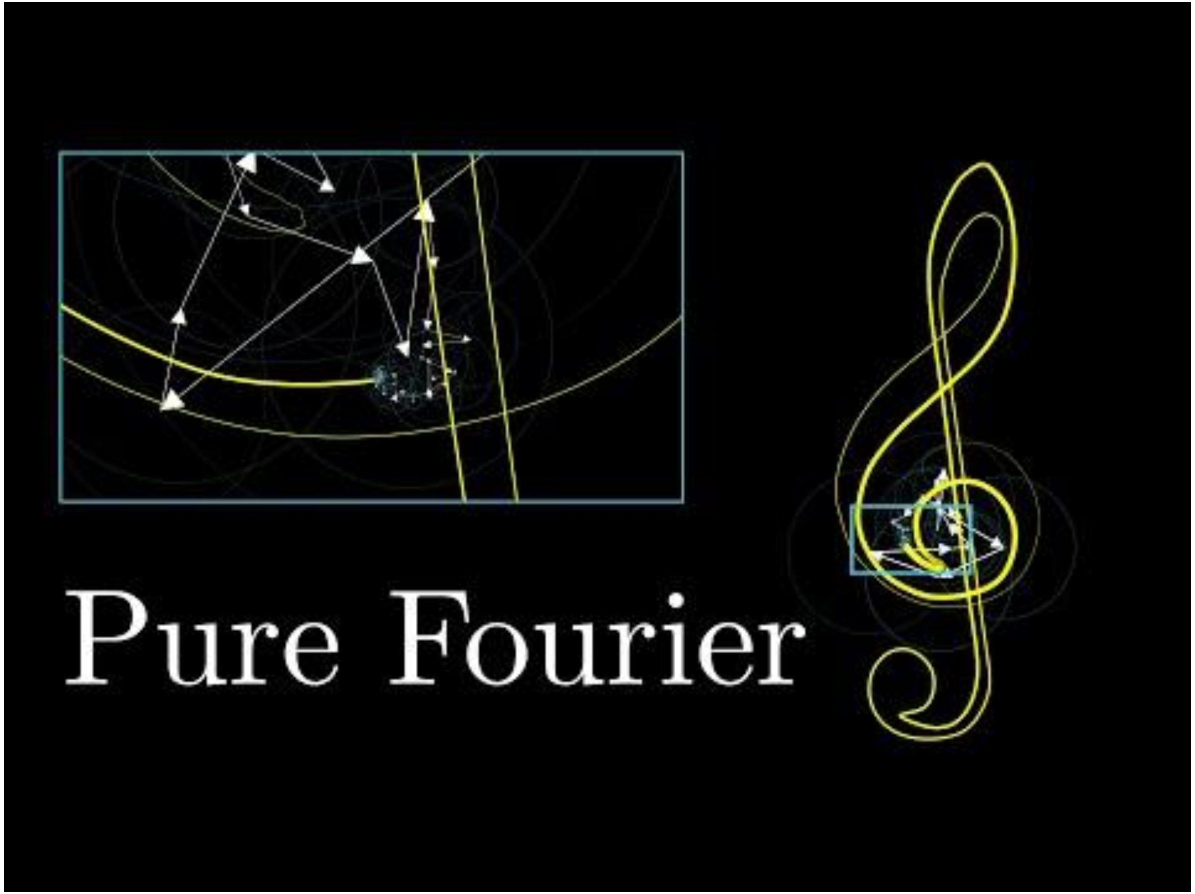
This complex number addition is actually what the DFT formula is expressing - hence the complex numbers in the formula!



Superposition

With enough complex sinusoids, we can approximate any function to basically an arbitrary degree of precision.

But again, in the real world, we don't have infinite anything!



From 3blue1brown: https://www.youtube.com/watch?v=-qgreAUpPwM

Key Points

- In order to analyze speech computationally, we need to digitize it
  - Sampling rate
  - Quantization
- Digitization brings in constraints
  - Nyquist Frequency: limits the frequencies we can actual captures
  - Aliasing: makes higher frequencies appear the same as lower frequencies

Key Points

- Map from the time domain to the frequency domain using the DFT
  - Frequency domain gives more direct characterisation of articulation from the signal
  - Analysis frequencies are determined by input size and sampling rate
  - We can only analyze frequencies up to the half the sampling rate (the Nyquist Frequency)
- Many engineering techniques have been developed to improve the accuracy of the DFT output
  - Windowing, and many other techniques
- Speech technologies use (variations of) the spectrogram to learn the relationship between speech, acoustics, and language automatically

Next week

- The source filter model, from a computational perspective

Extension:
The DFT equation in more detail

Discrete Fourier Transform

A mathematical procedure we can used to determine the frequency content of a discrete signal sequence

Mathematical view: for input x[n] with n=0,...,N-1 (N inputs)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi n}{N}k}$$

For k=0,..,N-1 (N analysis frequencies)

Discrete Fourier Transform

for input x[n] with n=0,...,N-1 (N inputs), for k=0,..,N-1 (N analysis frequencies)

The input sequence

A complex sinusoid rotating
at a specific frequency

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi n}{N}k}$$

Dot-product: a measure of similarity
between two sequences

Discrete Fourier Transform

for input x[n] with n=0,...,N-1 (N inputs), for k=0,..,N-1 (N analysis frequencies)

A magnitude (scale factor)

A phase angle (shift factor)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] e^{-j \frac{2\pi n}{N} k} = M e^{j\varphi}$$

A complex number

The DFT formula calculates the **similarity** between the input and the complex sinusoid of a specific frequency. It's output is a **complex number** that tells you how you would scale and shift that sinusoid in order to reconstruct the original input (summing the complex sinusoids corresponding to the analysis frequencies)

Discrete Fourier Transform

for input x[n] with n=0,...,N-1 (N inputs), for k=0,..,N-1 (N analysis frequencies)

A magnitude
(scale factor)

A phase angle
(shift factor)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi n}{N}k} = M e^{j\varphi}$$

A complex number

The DFT outputs represent N complex sinusoids whose frequencies are multiples of the 1st actual analysis frequency (i.e., DFT[1])

Discrete Fourier Transform

for input x[n] with n=0,...,N-1 (N inputs), for k=0,..,N-1 (N analysis frequencies)

A magnitude (scale factor)

A phase angle (shift factor)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi n}{N}k} = M e^{j\varphi}$$

A complex number

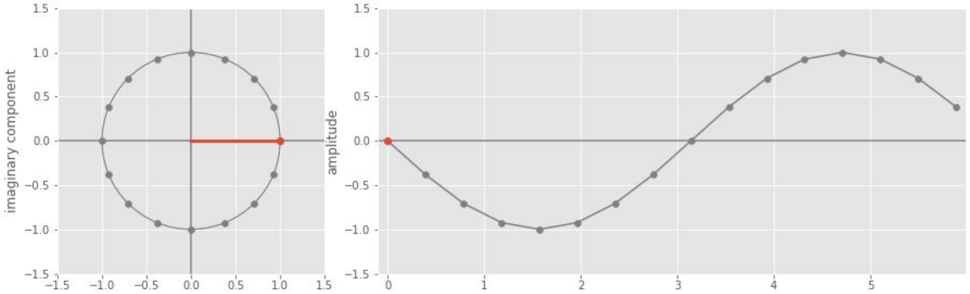
The fact the analysis frequencies are integer multiples of the first one means the (sampled) complex sinsuoids form are **orthogonal**: sinusoids of different frequencies have zero similarity. This is what allows the DFT to pick out specific frequencies as being in the input signal

DFT sinusoids

Input size N=16
So, N=16 DFT outputs

Assume a sampling rate of
800 samples per second

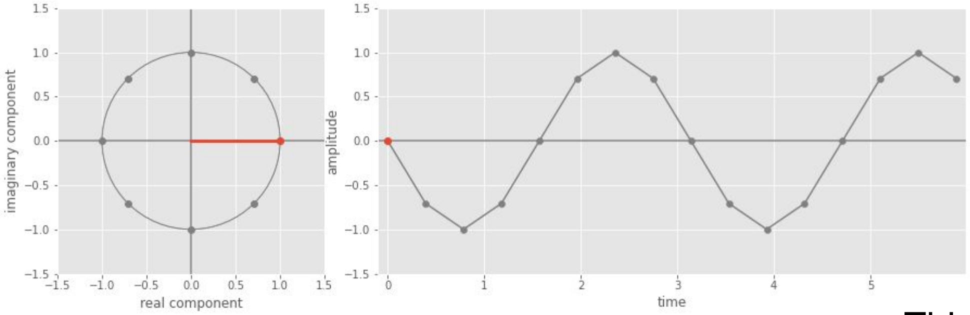
DFT[1]



16 steps for 1 cycle, 50 Hz

$$\mathrm{DFT}[1] = \sum_{n=0}^{N-1} x[n] e^{-j \frac{2\pi n}{N} \times 1}$$

DFT[2]



16 steps for 2 cycles, 100 Hz

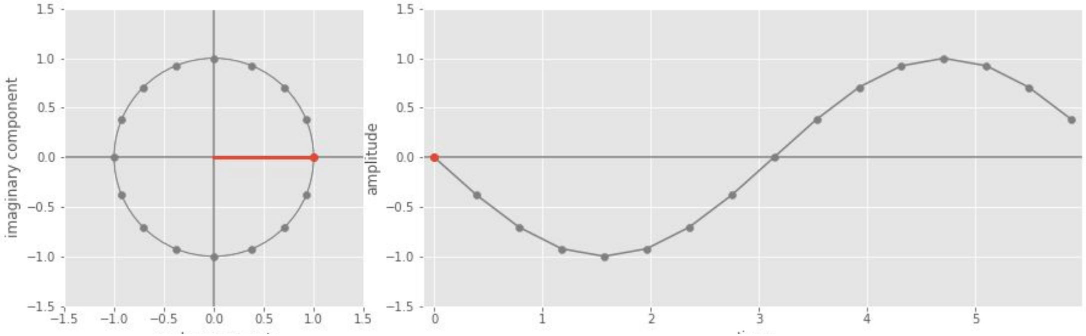
$$\mathrm{DFT}[2] = \sum_{n=0}^{N-1} x[n] e^{-j \frac{2\pi n}{N} \times 2}$$

Think of this as landing on every 2nd point of the DFT[1] phasor

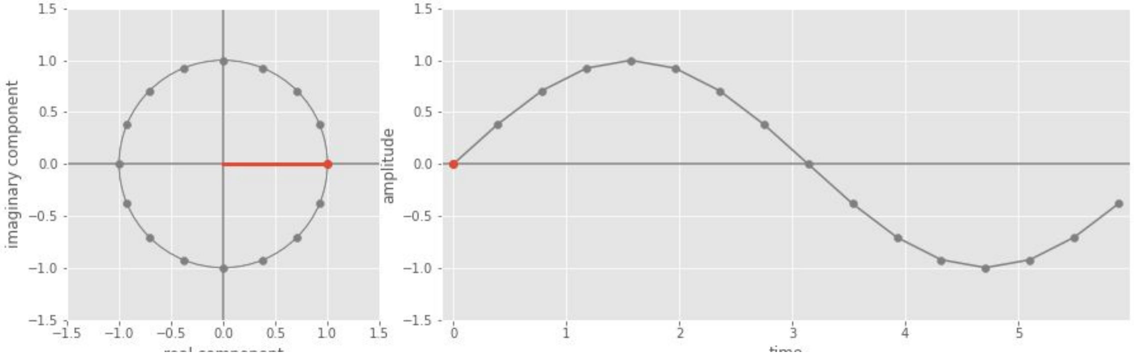
Aliasing again

Input size N=16
So, N=16 DFT outputs

DFT[1]



DFT[15]



DFT[15] is taking 15 steps for every 1 of DFT[1], so the phasor appears to be going backwards!

‘Wagon-wheel effect’

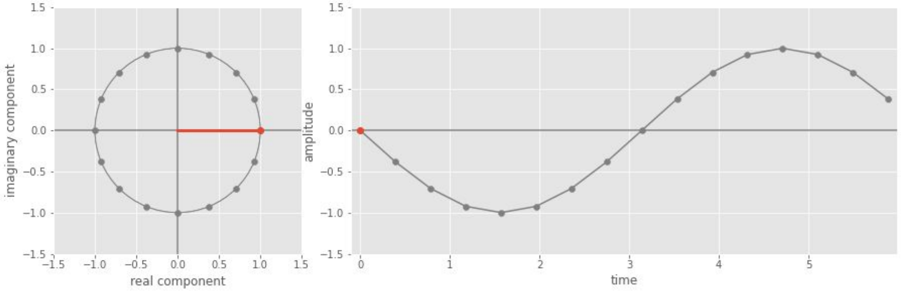
Aliasing again

Input size N=16
So, N=16 DFT outputs

If you look at the full DFT output you will see that the top half mirrors the bottom half, suggesting high frequency components that aren't there (see module 3 lab)

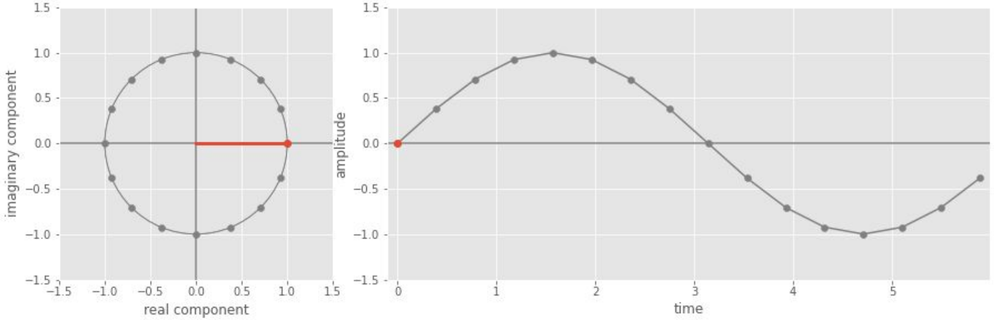
This is why visualizers, like Praat, just show up to the Nyquist frequency

DFT[1]



We can't actually capture the frequencies represented after the DFT[N/2], the Nyquist Frequency, because of the limit in sampling.

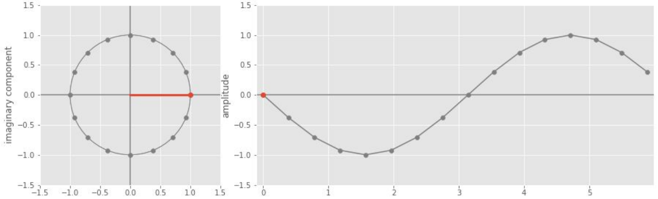
DFT[15]



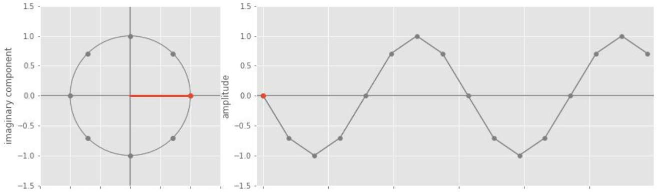
Aliasing again

Input size N=16
So, N=16 DFT outputs

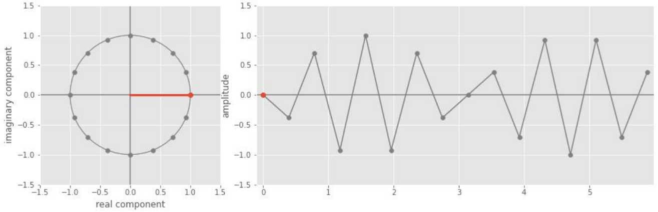
DFT[1]
50 Hz



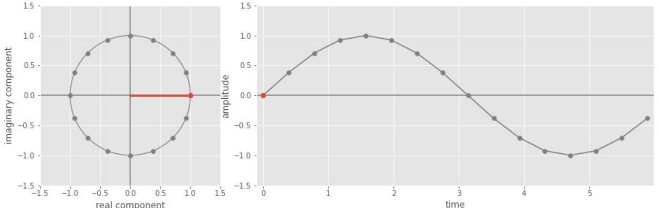
DFT[2]
100 Hz



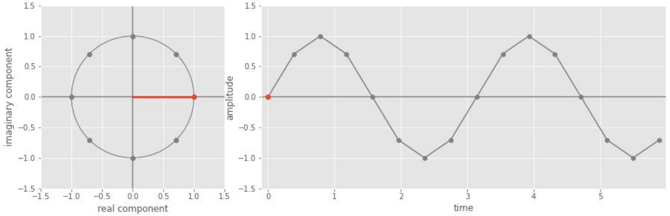
DFT[7]
350 Hz



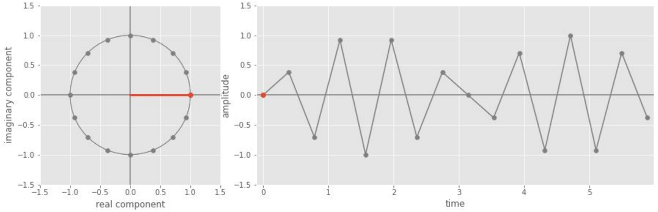
DFT[15]
750 Hz?



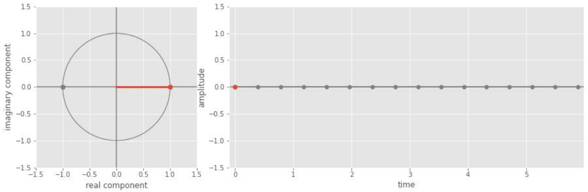
DFT[14]
700 Hz?



DFT[9]
450 Hz?



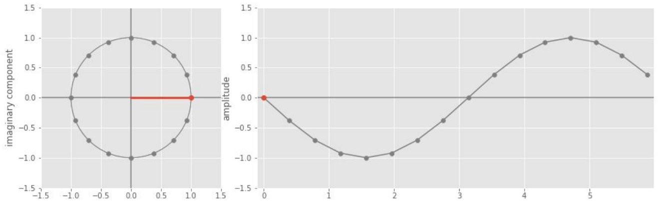
DFT[8]
400 Hz



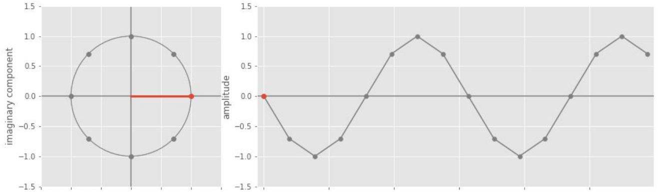
Aliasing again

Input size N=16
So, N=16 DFT outputs

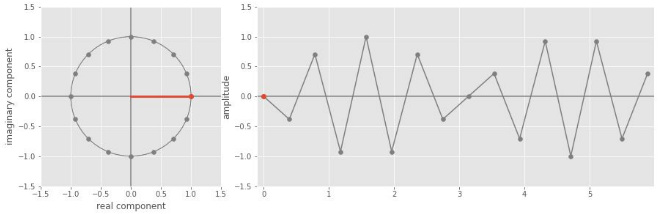
DFT[1]
50 Hz

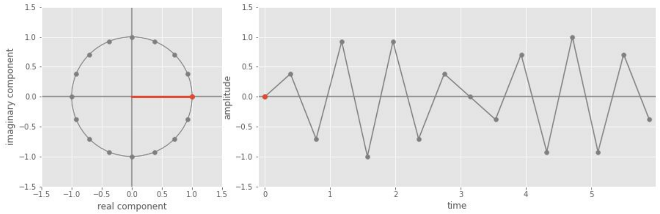


DFT[2]
100 Hz

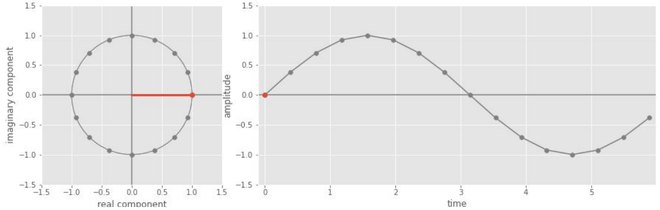


DFT[7]
350 Hz

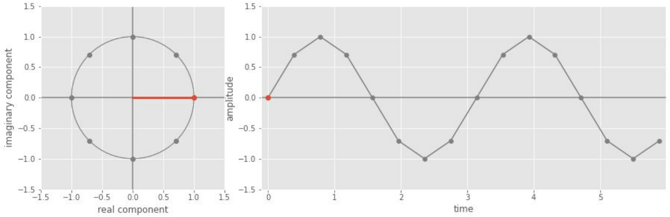




DFT[15]
~~750 Hz?~~
50 Hz

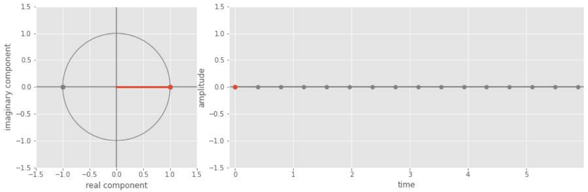


DFT[14]
~~700 Hz?~~
100 Hz



DFT[9]
~~450 Hz?~~
350 Hz

DFT[8]
400 Hz



Discrete Fourier Transform: cos and sine version

A mathematical procedure we can used to determine the frequency content of a discrete signal sequence

Mathematical view: for input x[n] with n=0,...,N-1 (N inputs)

$$\mathrm{DFT}[k] = \sum_{n=0}^{N-1} x[n] \left[\cos\left(\frac{2\pi n}{N} k\right) - j \sin\left(\frac{2\pi n}{N} k\right) \right]$$

For k=0,..,N-1 (N analysis frequencies)

Euler's Formula

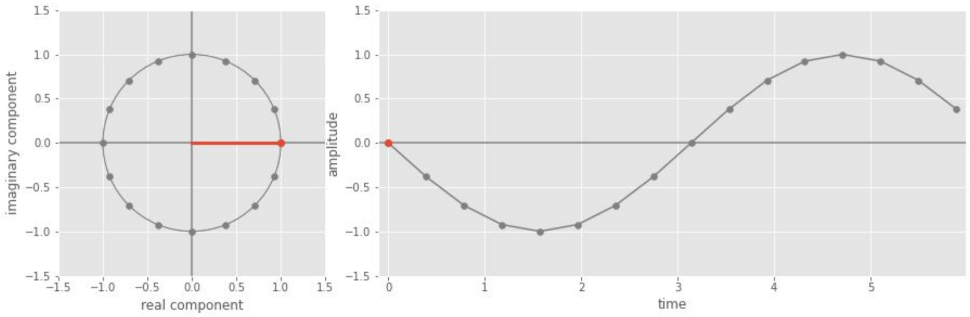
Some Extra Slides

DFT frequencies

Input size N=16
So, N=16 DFT outputs

Assume a sampling rate of
800 samples per second

DFT[1]



16 steps for 1 cycle

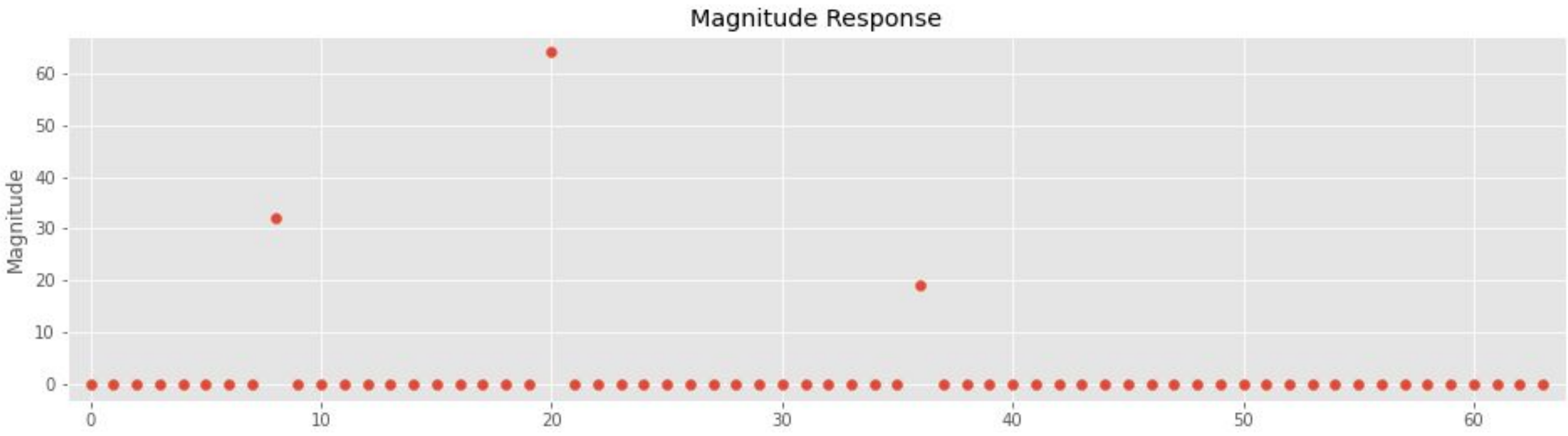
$$\mathrm{DFT}[1] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi n}{N}}$$

Sampling rate = 800 samples/second
Sampling time = 1/800 seconds

Q: How long does 1 cycle take?
A: the period T = 16 * 1/800 = 0.02 seconds

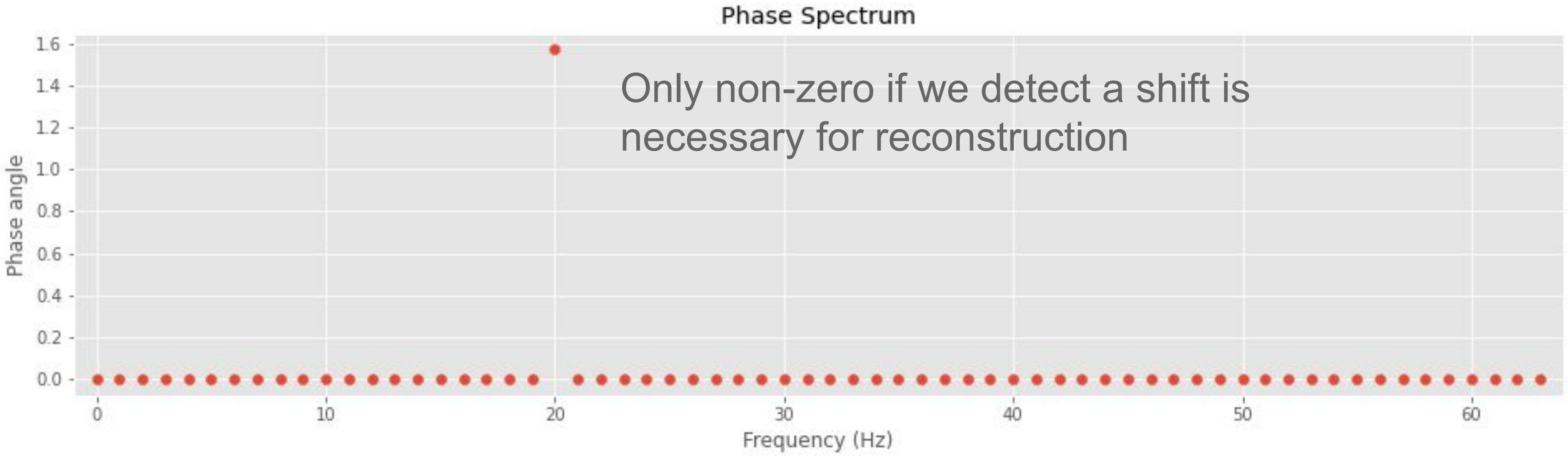
Q: What's the frequency associated with DFT[1]
A: frequency f = 1/T = 50 Hz

Magnitude Spectrum: scale



The zero magnitudes here indicate that we don't need these frequencies for reconstructing the input.
We do need the 8Hz, 20Hz and 36Hz frequencies!

Phase Spectrum: shift



We often ignore the phase spectrum in speech analysis as it doesn’t have much effect on human perception

Working out the Analysis Frequencies

- DFT[0] -> 1
  - Constant function (0 cycles because only 1 value)
- DFT[1] -> sinusoid which completes 1 cycle over the length of the input window
  - If N = number of input samples and f_s = sampling rate
  - What's the length of the input window in seconds? (T = N * (1/f_s))
  - What's the frequency of a sinusoid that completes 1 cycle in that time? (1/T = 1/(N/f_s) = f_s/N)
- DFT[2] -> sinusoid which completes 2 cycles over the length of the input window
- …

Working out the Analysis Frequencies

DFT[k] ↦ sinusoid which completes k cycles over the length of the input window

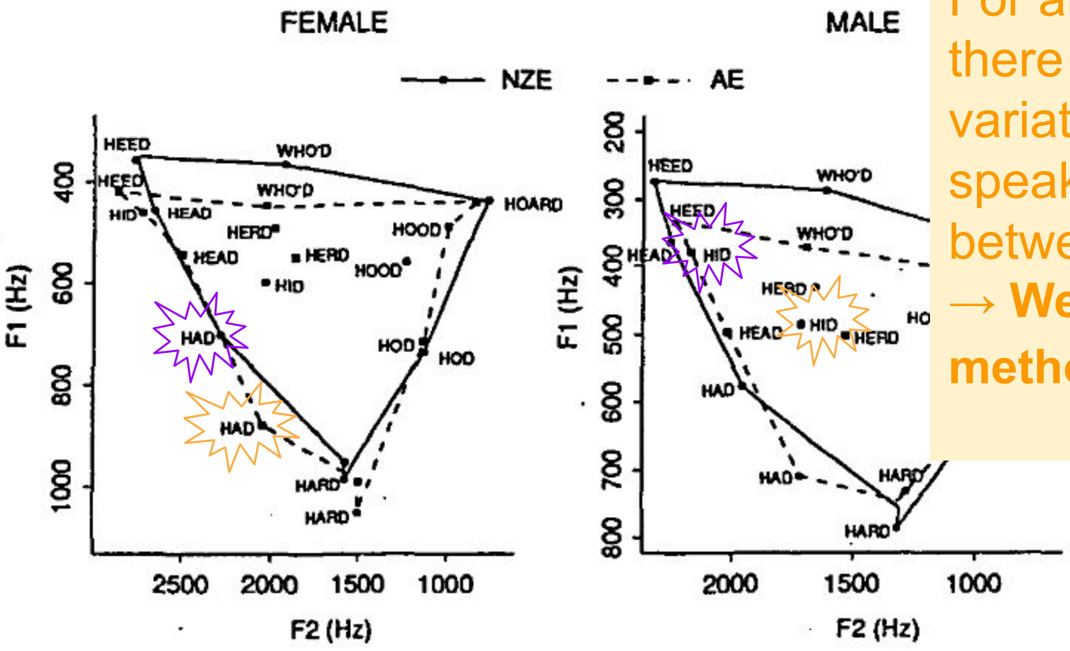
$$freq(DFT[N/2]) \rightarrow (N/2 * f_s)/N = f_s/2$$

- Which is half the sampling rate is the Nyquist Frequency, so now we have to think about aliasing!
- After sampling, sinusoids with frequencies higher than $f_s/2$ look the same as lower frequency ones

This means the full DFT output is actually mirrored around the Nyquist Frequency. This is why we only look at the first half of the mag spectrum.

(See module 3 lab)

Frequency Domain: Analyzing pronunciation differences New Zealand vs Australian English



For any specific phone, there is a lot of variation between speakers within and between dialects
→ **We need statistical methods!**

Figure 2. The centroid of the monophthong vowel targets from the NZE and AE for the female data *(left)* and the male data *(right)*.

Watson, C. I., Harrington, J., & Evans, Z. (1998). An acoustic comparison between New Zealand and Australian English vowels. *Australian journal of linguistics*, *18*(2), 185-207.