AUD401

# Dynamic Audio for Digital Media

## Lecture 4 ~ Sound Synthesis

Dr Christos Michalakos
c.michalakos@abertay.ac.uk

# So far:

1. Counters
2. Digital Audio
3. Loading Samples
4. Playing them Back

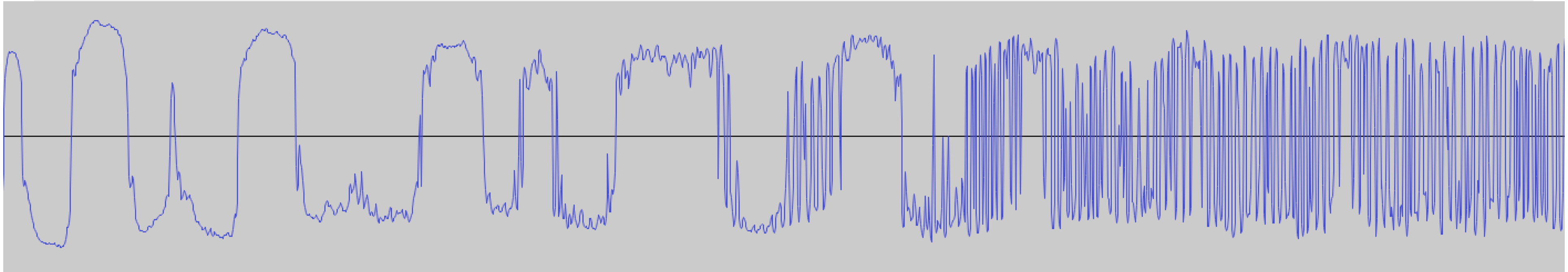
# Today

**Additive Synthesis**
**FM Synthesis**

## But first

## Sound Synthesis Fundamentals

# Pitch / Noise



# Some definitions

## Fundamental

The lowest frequency of a periodic waveform

## Overtones

All sinusoidal peaks above that

## Partials

All of the above

## Pitched:

## Harmonic / Inharmonic Sound

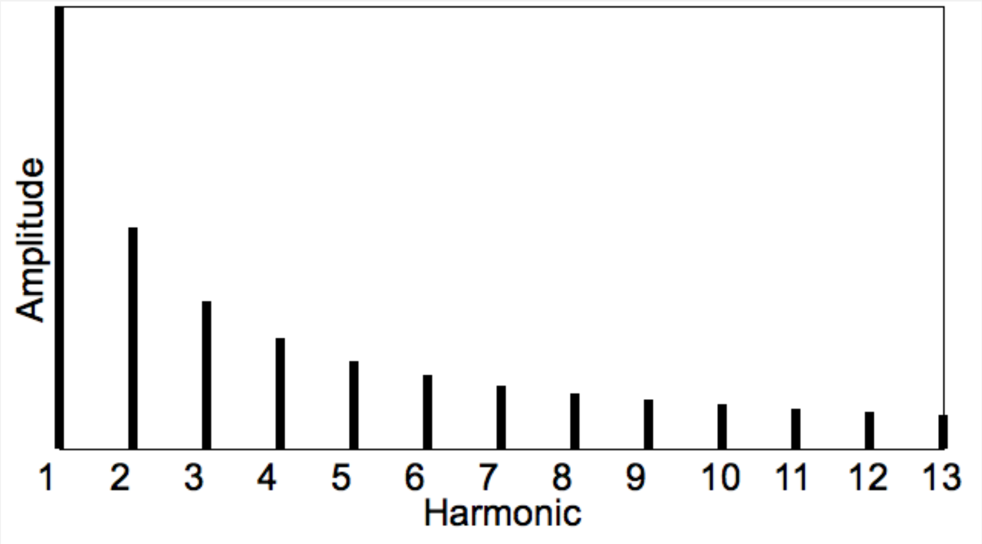
## Harmonic

- Clear Sinusoidal peaks
- Integer multiples

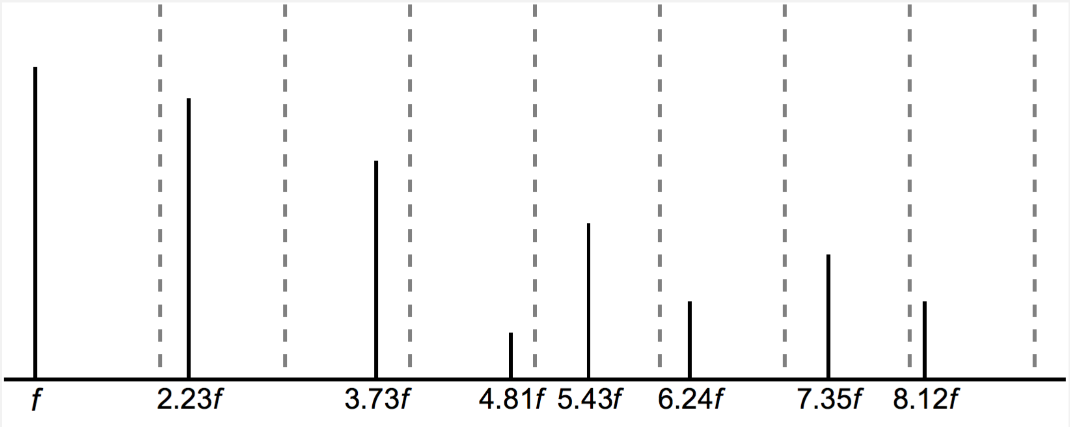
## Inharmonic

- Clear Sinusoidal peaks
- **NOT** integer multiples

# Harmonic




# Inharmonic

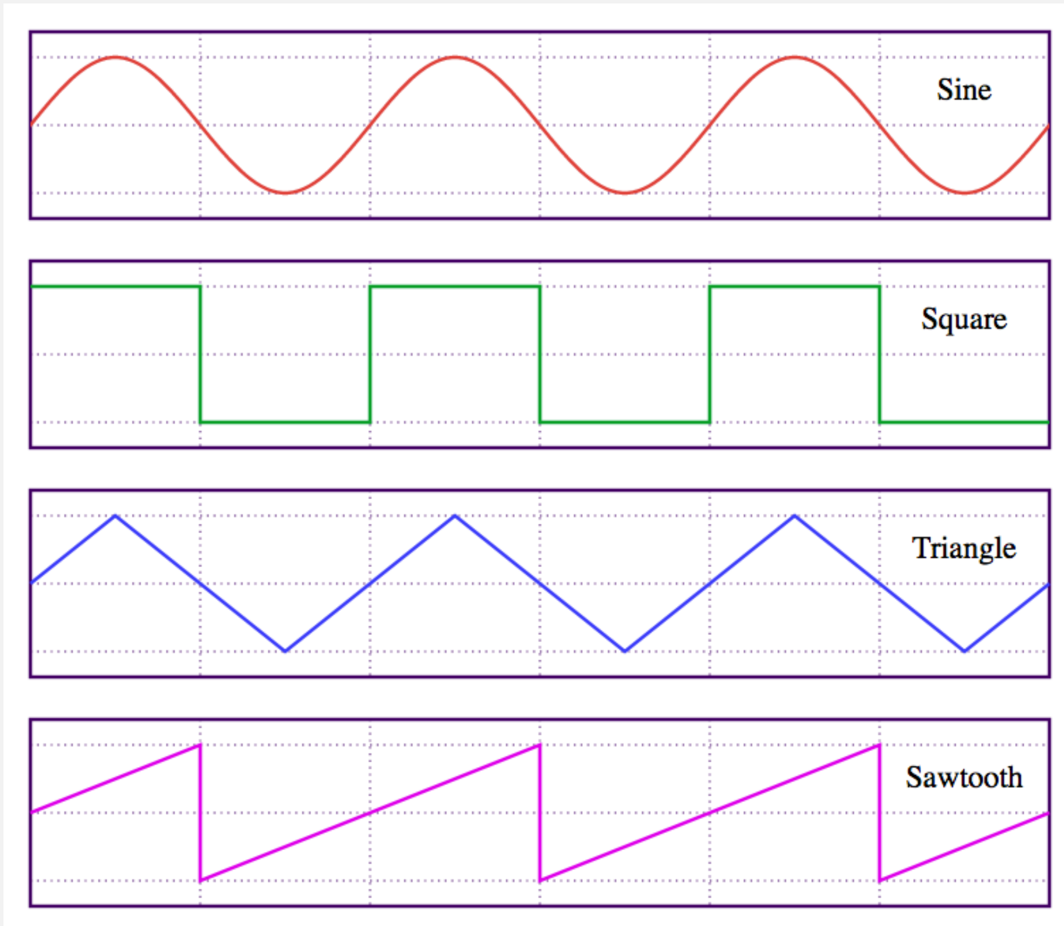


# Fourier Series

Any periodic waveform can be represented as a sum of simple sine waves

Mathematician **Joseph Fourier** 1768-1830

# How can we express our basic waveforms by adding sinusoids?



**…we can do this using Wavetables**

# Wavetables

- Storage of waveform data in an array (i.e. memory)
- Stored as amplitude vs time: *i.e.* indexed by time
- Lookup data at a particular frequency (sample the wavetable)
- Use interpolation to produce values not directly represented in the table

# Wavetables

- Useful for static waveforms as often takes less computation to lookup a table value, than directly calculate values from waveform functions
- Can crossfade between several wavetable oscillators to create varying timbres

# Wavetables

- In pd, create an **`array`** to hold your wavetable in

- You will need to decide on a length. If you are using **`tabosc4~`**, the length of the array should be a power of two plus 3 extra points to allow for the 4-point interpolation. *e.g.* 131, 259, 515, 1027

- You can draw the waveform directly into the table

# Wavetables

Array messages:

**`sinesum`** creates the weighted sum of sine waves and puts the results into the array. It takes the form:

**`sinesum length a(0) a(1) a(2) … a(x)`**

Where **length** is the length of the array, and **a(0)** the amplitude of the fundamental, **a(1)** the amplitude of the first harmonic, *etc.* **NB** unlike cosinesum there is **no** offset argument!

# Wavetables

With **`sinesum`**:

- Notice that it **resizes** your array to fit the length specified. It automatically add the guard points required for interpolation!

- Watch that the total magnitude does not go above 1.0 or you will get distortion. You can send a `normalize` message to the table to get the waveform within the proper bound

# Wavetables

Other messages:

- **`const`** set array to constant value
- **`normalize`** normalize array to certain value

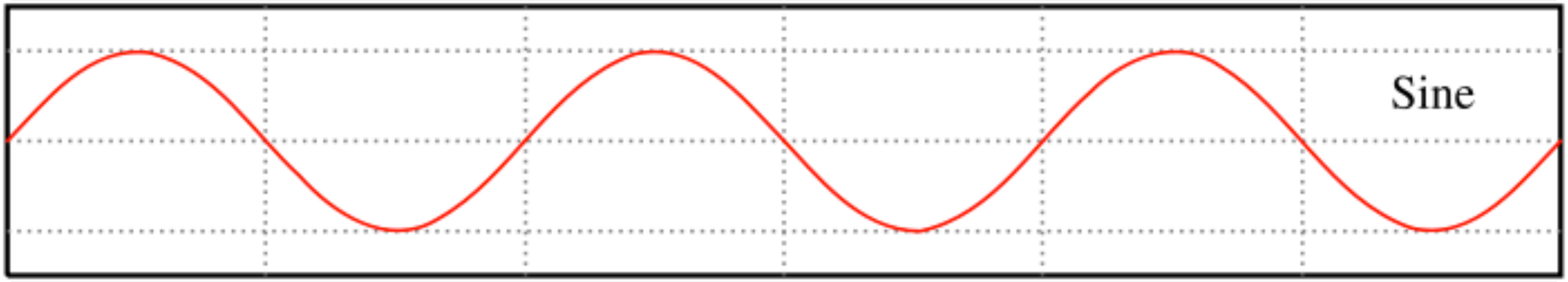
# tabosc4~

- Leftmost inlet: frequency
- Right inlet: reset phase (*cf* `osc~`)
- Use `set` messages to switch between tables (<u>do not</u> use sinesum or cosinesum messages to array while running to change timbre!)

**Again:**
Any periodic waveform can be represented as a sum of simple sine waves

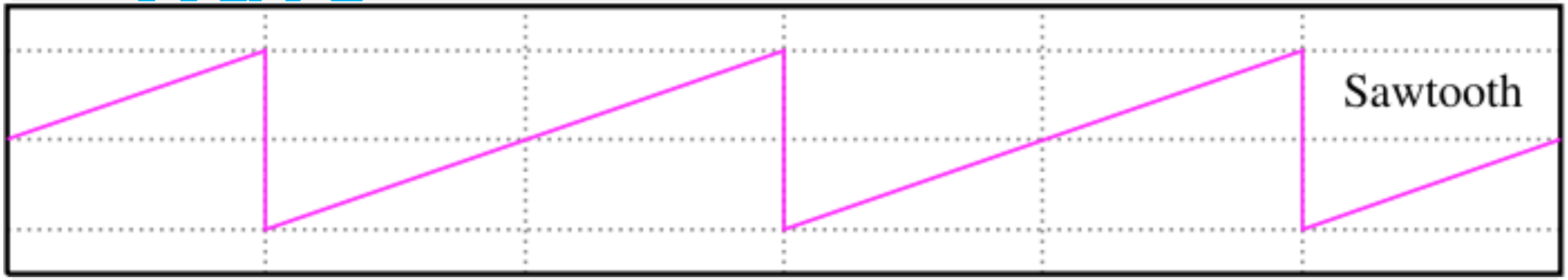
**…so let's start adding sine waves**

# Sine Wave



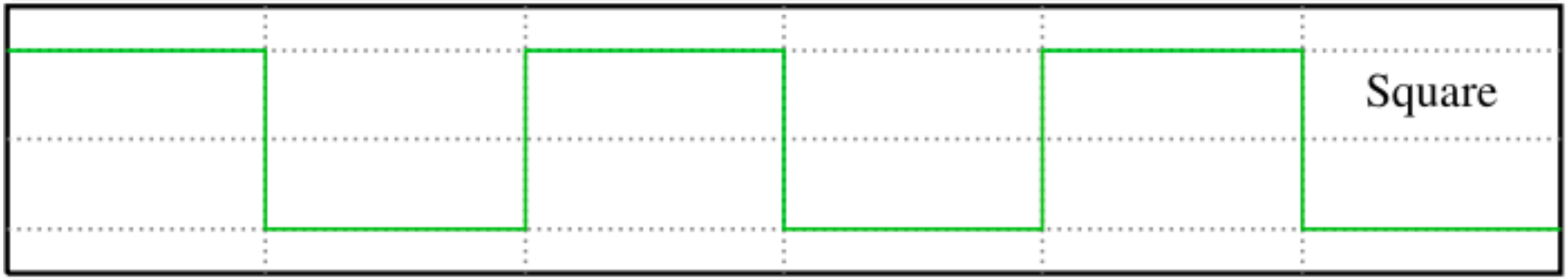

- Fundamental only
- No additional harmonics

# Sawtooth Wave
Wave



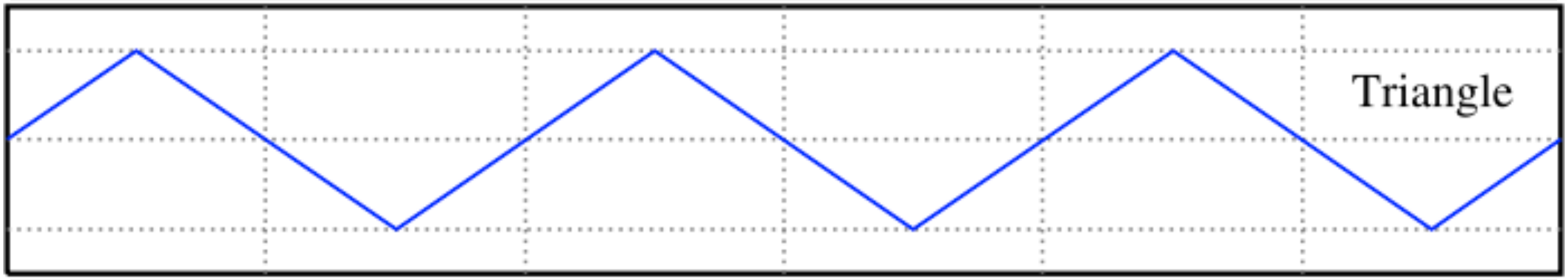

- All partials
- Partial amplitude = 1/p#

# Square Wave




- Odd-numbered partials only
- Partial amplitude = 1/p#

# Triangle Wave




- Odd-numbered partials only
- Partial amplitude = $1/p\#^2$
- Amplitude of every other harmonic multiplied by -1

# Additive Synthesis!

# Additive Synthesis

• The basic force behind the use of additive synthesis is a desire to create complex musical sounds by adding (mixing) together multiple, simpler, sound components.

• The component sounds are not usually perceived, but instead contribute to the quality (timbre) of the synthesised resulting sound.

# Additive Synthesis

• The theoretical framework for additive synthesis based on Fourier analysis.

• To put Fourier's theorem in basic terms: periodic (acoustic) waveforms can be expressed as a sum of harmonically related sine waves, each with a particular phase and amplitude. This sum may, however, be infinite!

# Additive Synthesis

- Although sine waves are often used as the building blocks for additive synthesis, different waveforms may also be used to create more complex results.

# Frequency Modulation Synthesis (FM)

# FM Synthesis

- The ‘carrier’ frequency is being frequency modulated.

- Terms:

  **Carrier** — waveform being modulated.
  **Modulation Frequency**—rate of modulation.
  **Modulation Index**—how much the signal is modulated by.

# FM Synthesis

- Discovered by John Chowning in the early 70’s and patented to Yamaha (used in, for example, the Yamaha DX7).

- In its simplest form, it comprises a sine wave carrier whose frequency is varied (modulated) by another waveform (e.g. another sine wave).

- When the modulation frequency is sub-audio (below c. 20Hz, the change in pitch is perceptible.

# FM Synthesis

If **fm / fc** is a positive integer, then the spectrum of the resulting waveform will be **harmonic**.

• If **fm / fc** is a positive non-integer, then the spectrum will be **inharmonic**. This can be useful for synthesizing things such as bells, gongs and even drums.

# Tutorial