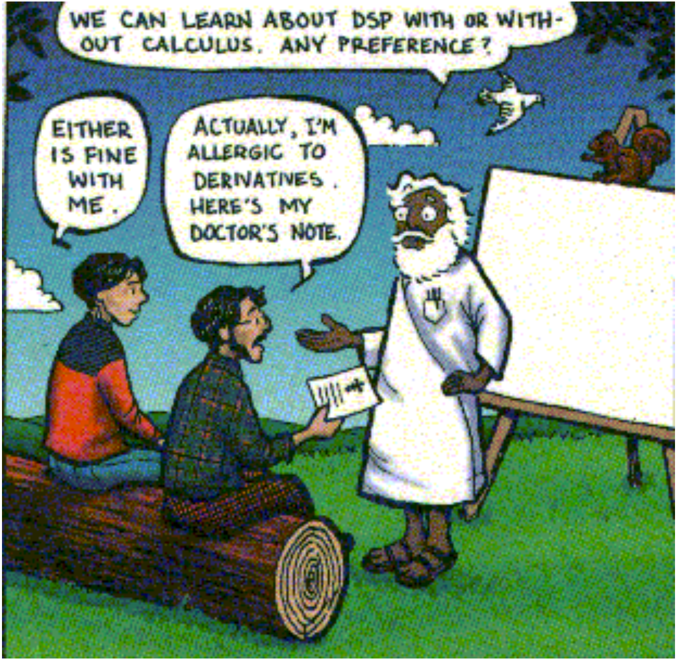
CM2202: Scientific Computing and Multimedia Applications
Digital Signal Processing 1. Introduction
Analogue and Digital Signals, and Sampling

Prof. David Marshall

School of Computer Science & Informatics


WE CAN LEARN ABOUT DSP WITH OR WITH-OUT CALCULUS. ANY PREFERENCE?
EITHER IS FINE WITH ME.
ACTUALLY, I'M ALLERGIC TO DERIVATIVES. HERE'S MY DOCTOR'S NOTE.

Digital Signal Processing and Digital Audio

Issues to be covered (Over next few lectures):

- Digital Signal Processing and Digital Audio
  - Sampling Theorem
  - Digital Audio Signal Processing
  - Digital Audio Effects

What is Digital Signal Processing (DSP)?

Digital Signal Processing (DSP)

DSP includes **many different topics**, such as:

- Digital filters
- Analysis of signals and systems (especially in terms of frequency)
- Synthesis of signals
- Detection of signals and estimation of signal and system parameters
- Data compression
- and on and on ...

DSP Related Subjects

Related Topics

DSP is the intersection of a number of different areas of study:

- Mathematics
- Electrical engineering
- Signals and systems
- Analog circuit theory
- Computer architecture, (and more)
- Probability and statistics
- Computer programming

Strong Link to **Image** and **Video Processing** — **more soon**

Digital Audio Applications

Application of Digital Audio — Selected Examples

Music Production

- Hard Disk Recording
- Sound Synthesis
- Samplers
- Effects Processing

Video

- Audio Important Element: Music and Effects

Web

- Many uses on Web:
- Streaming Audio
  - Spotify
  - Listen to Web Radio
- Element of a Web Page

What is Sound?

Sound Generation

Source — Generates Sound

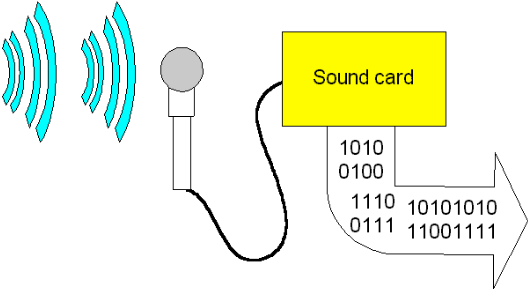
- Air Pressure changes
- *Electrical* — Loud Speaker
- *Acoustic* — Direct Pressure Variations

Sound Reception

Destination — Receives Sound

- *Electrical* — Microphone produces electric signal
- *Ears* — Responds to pressure **hear** sound (MPEG Audio — exploits this fact)

Digitising Sound



- Microphone:
  - Receives sound
  - Converts to **analog signal**.
- Computer like **discrete entities**

Need to convert Analog-to-Digital — Dedicated Hardware (*e.g.* Soundcard)

Also known as Digital Sampling

Computer Manipulation of Sound

Digital Audio Examples

Digital Signal Processing routines range from being **trivial** to **highly complex** :

- Volume
- Cross-Fading
- Looping
- Echo/Reverb/Delay
- Filtering
- Signal Analysis

Sample Rates and Bit Size

Bit Size — Quantisation

How do we store each sample value (**Quantisation**)?

8 Bit Value (0-255)

16 Bit Value (Integer) (0-65535)

Sample Rate

How many Samples to take?

11.025 KHz — Speech (Telephone 8 KHz)

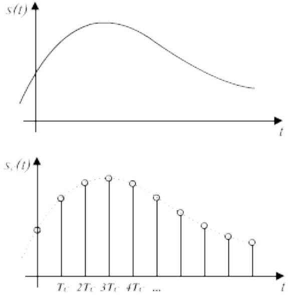
22.05 KHz — Low Grade Audio (WWW Audio, AM Radio)

44.1 KHz — CD Quality

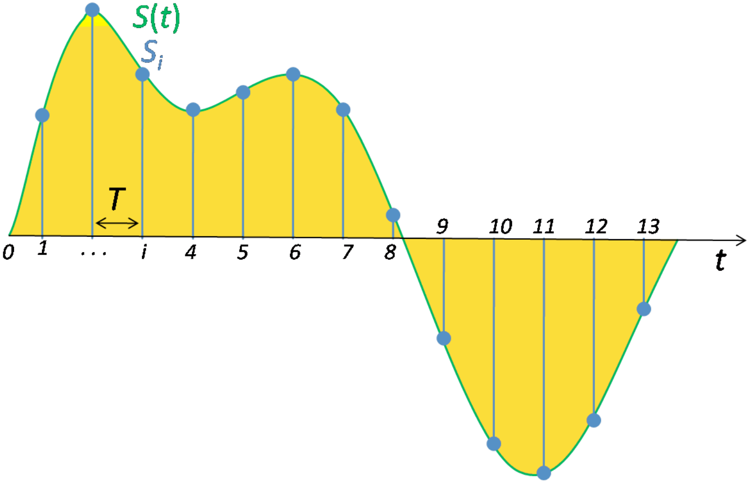
Digital Sampling (1)

Sampling process basically involves:

- **Measuring** the **analog signal** at **regular discrete intervals**
- **Recording** the **value** at **these points**



Digital Sampling (2)



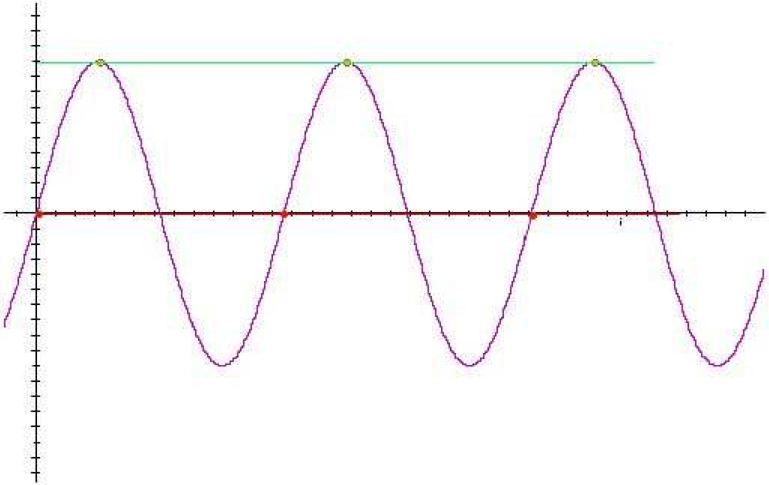
Nyquist's Sampling Theorem

The **Sampling Frequency** is **critical** to the **accurate reproduction** of a **digital version** of an analog waveform

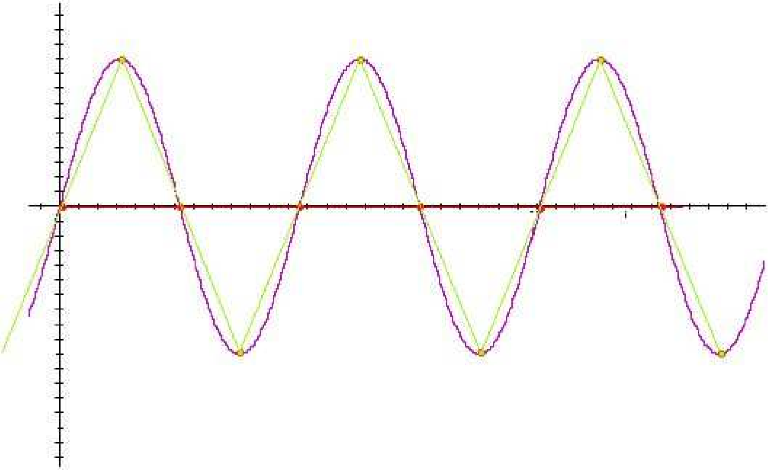
Nyquist's Sampling Theorem

The Sampling frequency for a signal must be **at least twice** the **highest frequency component** in the signal.

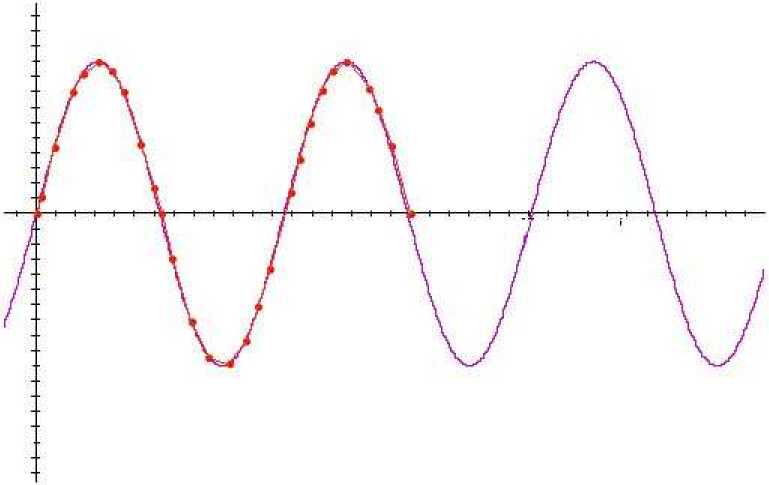
Sampling at Signal Frequency



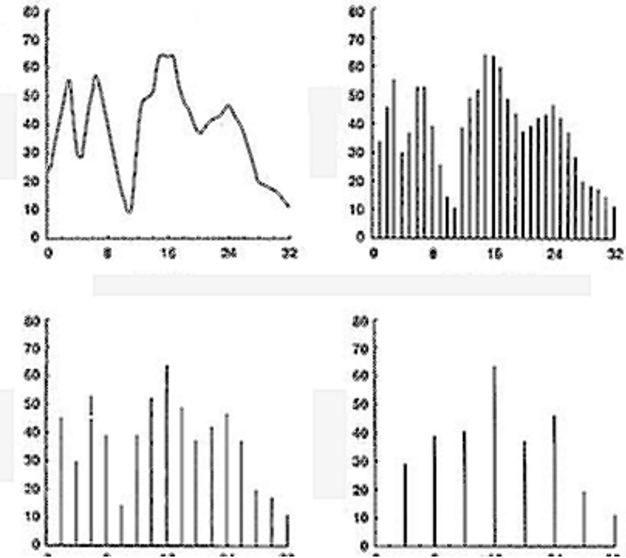
Sampling at Twice Nyquist Frequency



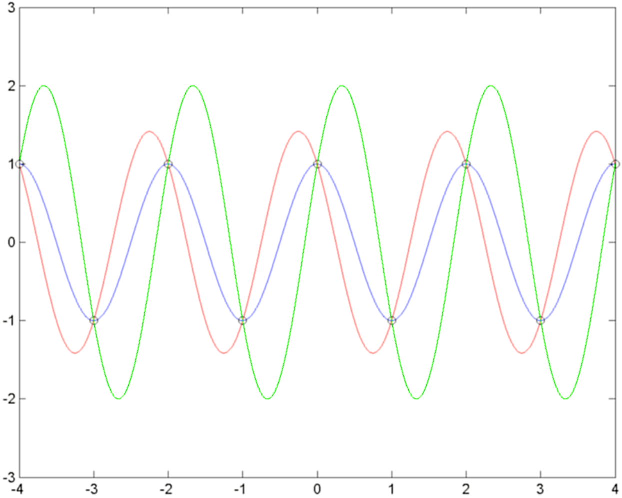
Sampling at above Nyquist Frequency



If you get Nyquist Sampling Wrong? (1)



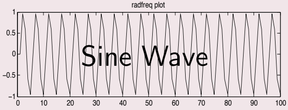
If you get Nyquist Sampling Wrong? (2)

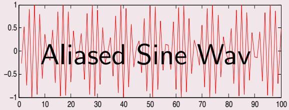


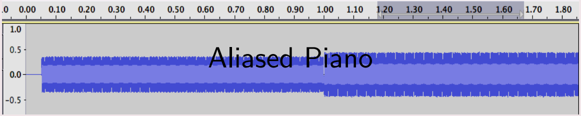
If you get Nyquist Sampling Wrong? (3)

What does aliasing sound like?

(**Click on Images to play sounds**)







MATLAB Code for Sine Demos above: Plot Version , Audio Version

Implications of Sample Rate and Bit Size (1)

Affects Quality of Audio

- Ears do not respond to sound in a linear fashion
- Decibel (**dB**) a logarithmic measurement of sound
- 16-Bit has a signal-to-noise ratio of 98 dB — virtually inaudible
- 8-bit has a signal-to-noise ratio of 50 dB
- Therefore, 8-bit is roughly 8 times as noisy
    - 6 dB increment is twice as loud

Implications of Sample Rate and Bit Size (2)

Audio Sample Rate and Bit Size Examples

File Type	Audio File (all mono)
44Hz 16 bit	
44KHz 8-bit	
22 KHz 16-bit	
22KHz 8-Bit	
11KHz 8-bit	

Web Link: Click Here to Hear Sound Examples

Implications of Sample Rate and Bit Size (2)

Affects Size of Data

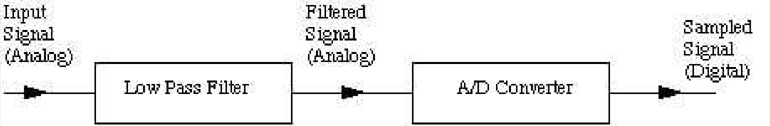
File Type	*44.1 KHz*	*22.05 KHz*	*11.025 KHz*
16 Bit Stereo	10.1 Mb	5.05 Mb	2.52 Mb
16 Bit Mono	5.05 Mb	2.52 Mb	1.26 Mb
8 Bit Mono	2.52 Mb	1.26 Mb	630 Kb

Memory Required for **1 Minute** of Digital Audio

Practical Implications of Nyquist Sampling Theory

Filtering of Signal

- Must (low pass) filter signal before sampling:



- Otherwise **strange artifacts** from high frequency (**above** Nyquist Limit)signals would appear in the sampled signal.

Why are CD Sample Rates 44.1 KHz?

Why are CD Sample Rates **44.1 KHz**?

Why are CD Sample Rates 44.1 KHz?

Why are CD Sample Rates **44.1 KHz**?

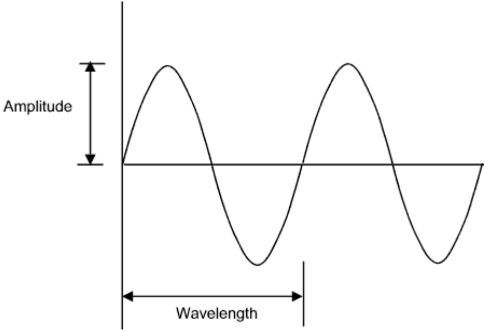
Upper range of human hearing is around 20-22 KHz — Apply Nyquist Theorem

Basic Digital Audio Signal Processing

In this section we look at some basic aspects of **Digital Audio Signal Processing**:

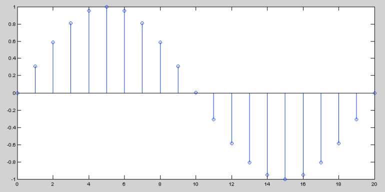
- Some basic definitions and principles
- Filtering
- Basic Digital Audio Effects

Simple Waveforms



- **Frequency** is the number of cycles per second and is measured in Hertz (Hz)
- **Wavelength** is *inversely proportional* to frequency i.e. Wavelength varies as $\frac{1}{frequency}$

The Sine Wave and Sound



The general form of the sine wave we shall use (quite a lot of) is as follows:

$$y = A.sin(2\pi.n.F_w/F_s)$$

where:

A is the amplitude of the wave,
F_w is the frequency of the wave,
F_s is the sample frequency,
n is the sample index.

MATLAB function: `sin()` used — works in radians

MATLAB Sine Wave Radian Frequency Period

Basic 1 period Simple Sine wave — **1 period is 2π radians**

Basic 1 period Simple Sine wave

```
% Basic 1 period Simple Sine wave

i=0:0.2:2*pi;
y = sin(i);
figure(1)
plot(y);

% use stem(y) as alternative plot as in lecture not
% see sample values

title('Simple 1 Period Sine Wave');
```

MATLAB Sine Wave Amplitude

Sine Wave Amplitude is -1 to +1.

To change amplitude multiply by some gain (`amp`):

Sine Wave Amplitude Amplification

```
% Now Change amplitude

amp = 2.0;

y = amp*sin(i);

figure(2)
plot(y);
title('Simple 1 Period Sine Wave Modified Amplitude');
```

MATLAB Sine Wave Frequency

Sine Wave Change Frequency

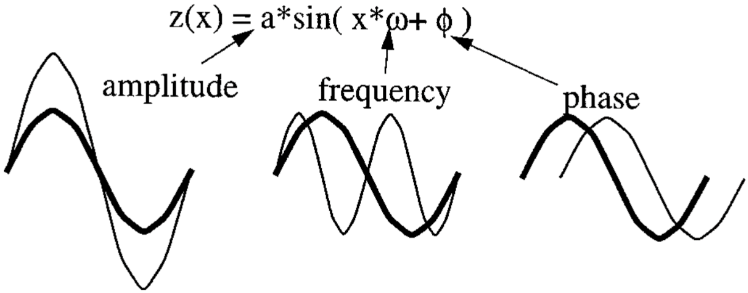
```
% Natural frequency is 2*pi radians
% If sample rate is F_s HZ then 1 HZ is 2*pi/F_s
% If wave frequency is F_w then freequency is F_w* (2*pi/F_s)
% set n samples steps up to sum duration nsec*F_s where
% nsec is the duration in seconds
% So we get y = amp*sin(2*pi*n*F_w/F_s);

F_s = 11025;
F_w = 440;
nsec = 2;
dur= nsec*F_s;

n = 0:dur;

y = amp*sin(2*pi*n*F_w/F_s);

figure(3)
plot(y(1:500));
title('N second Duration Sine Wave');
```

MATLAB Sine Wave Plot of *n* cycles

Plotting of *n* cycles of a Sine Wave

```
% To plot n cycles of a waveform

ncyc = 2;

n=0:floor(ncyc*F_s/F_w);

y = amp*sin(2*pi*n*F_w/F_s);

figure(4)
plot(y);
title('N Cycle Duration Sine Wave');
```

Relationship Between Amplitude, Frequency and Phase



MATLAB Sine Wave Frequency and Amplitude (only)

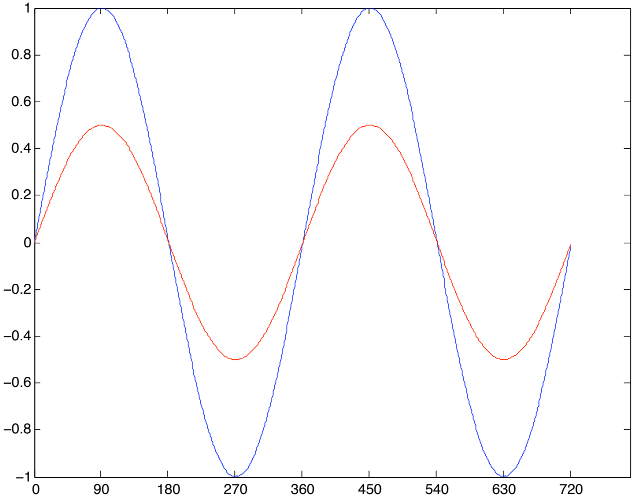
```
% Natural frequency is 2*pi radians
% If sample rate is F_s HZ then 1 HZ is 2*pi/F_s
% If wave frequency is F_w then frequency is
%        F_w* (2*pi/F_s)
% set n samples steps up to sum duration nsec*F_s where
% nsec is the duration in seconds
% So we get y = amp*sin(2*pi*n*F_w/F_s);

F_s = 11025;
F_w = 440;
nsec = 2;
dur= nsec*F_s;
n = 0:dur;

y = amp*sin(2*pi*n*F_w/F_s);
figure(1)
plot(y(1:500));
title('N second Duration Sine Wave');
```

Amplitudes of a Sine Wave

Code for sinampdemo.m

```
% Simple Sin Amplitude Demo
samp_freq = 400;
dur = 800; % 2 seconds
amp = 1; phase = 0; freq = 1;
s1 = mysin(amp,freq,phase,dur,samp_freq);

axisx = (1:dur)*360/samp_freq; % x axis in degrees
plot(axisx,s1);
set(gca,'XTick',[0:90:axisx(end)]);

fprintf('Initial Wave: \t Amplitude = ...\n', amp,
              freq, phase,...);

% change amplitude
amp = input('\nEnter Amplitude:\n\n');

s2 = mysin(amp,freq,phase,dur,samp_freq);
hold on;
plot(axisx, s2,'r');
set(gca,'XTick',[0:90:axisx(end)]);
```

Amplitudes of a Sine Wave: `sinampdemo` output



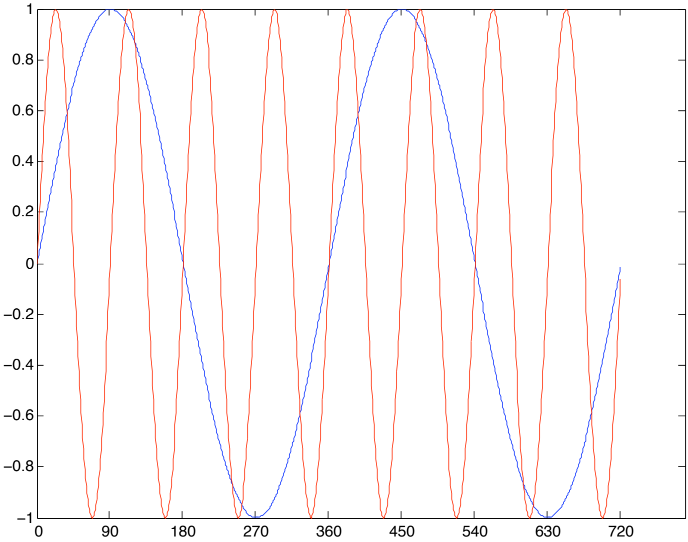
mysin.m — a convenience MATLAB sine function for amplitude, frequency and phase.

Frequencies of a Sine Wave

Code for sinfreqdemo.m

```
% Simple Sin Frequency Demo

samp_freq = 400;
dur = 800; % 2 seconds
amp = 1; phase = 0; freq = 1;
s1 = mysin(amp,freq,phase,dur,samp_freq);

axisx = (1:dur)*360/samp_freq; % x axis in degrees
plot(axisx,s1);
set(gca,'XTick',[0:90:axisx(end)]);

fprintf('Initial Wave: \t Amplitude = ...\n', amp, freq, phase,...

% change amplitude
freq = input('\nEnter Frequency:\n\n');

s2 = mysin(amp,freq,phase,dur,samp_freq);
hold on;
plot(axisx, s2,'r');
set(gca,'XTick',[0:90:axisx(end)]);
```

Frequencies of a Sine Wave: `sinfreqdemo` output

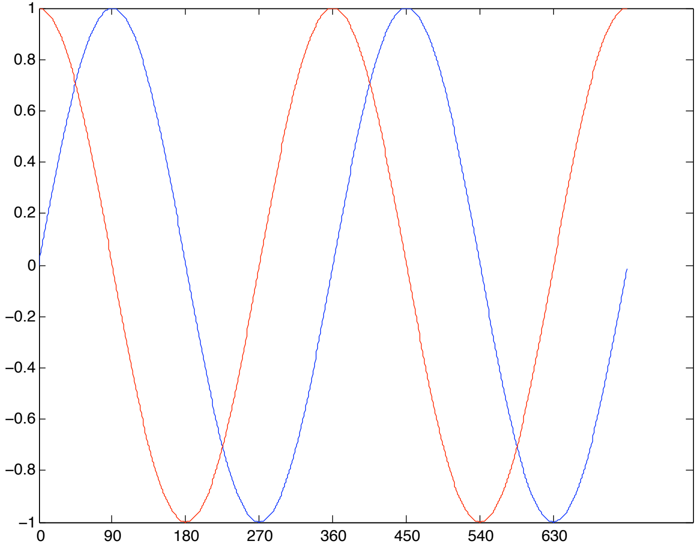


Phase of a Sine Wave

sinphasedemo.m

```
% Simple Sin Phase Demo

samp_freq = 400;
dur = 800; % 2 seconds
amp = 1; phase = 0; freq = 1;
s1 = mysin(amp,freq,phase,dur,samp_freq);

axisx = (1:dur)*360/samp_freq; % x axis in degrees
plot(axisx,s1);
set(gca,'XTick',[0:90:axisx(end)]);

fprintf('Initial Wave: \t Amplitude = ...\n', amp, freq, phase,...

% change amplitude
phase = input('\nEnter Phase:\n\n');

s2 = mysin(amp,freq,phase,dur,samp_freq);
hold on;
plot(axisx, s2,'r');
set(gca,'XTick',[0:90:axisx(end)]);
```

Phase of a Sine Wave: `sinphasedemo` output



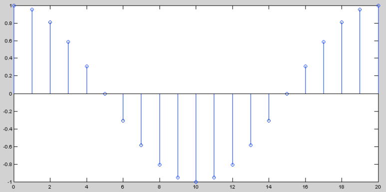
MATLAB Square and Sawtooth Waveforms

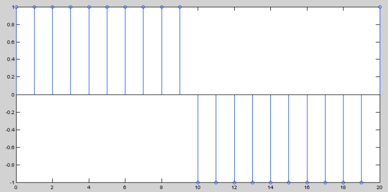
MATLAB Square and Sawtooth Waveforms

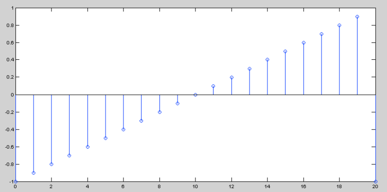
```
% Square and Sawtooth Waveforms created using Radians

ysq = amp*square(2*pi*n*F_w/F_s);
ysaw = amp*sawtooth(2*pi*n*F_w/F_s);

figure(6);
hold on
plot(ysq,'b');
plot(ysaw,'r');
title('Square (Blue)/Sawtooth (Red) Waveform Plots');
hold off;
```

Cosine, Square and Sawtooth Waveforms

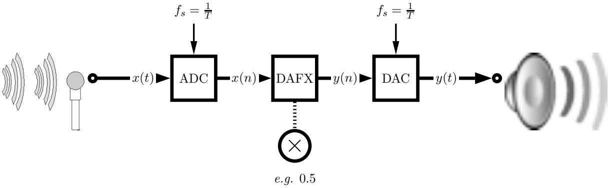
MATLAB functions `cos()` (cosine), `square()` and `sawtooth()` similar.

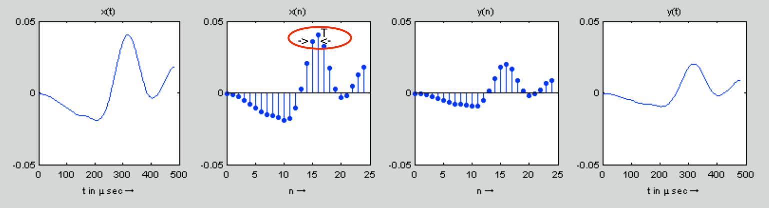






Digital Audio Effects (DAFX) Example





DAFX: Sample Interval and Sample Frequency

- An **analog signal,** $x(t)$ with signal amplitude continuous over time, t.
- Following **ADC** the signal is converted into a **a discrete-time and quantised amplitude signal,** $x(n)$ — a stream of samples over discrete time index, n
  - The time distance between two consecutive samples, **the sample interval,** T (or sampling period).
  - The **the sampling frequency** is $f_s = \frac{1}{T}$ — the number of samples per second measured in Hertz (Hz).
- Next we apply some simple **DAFX** — *E.g* here we multiply the signal by a factor of 0.5 to produce $y(n) = 0.5.x(n)$.
- The signal $y(n)$ is then forwarded to the **DAC** which reconstruct an analog signal $y(t)$

Basic DSP Concepts and Definitions: The Decibel (dB)

When referring to measurements of power or intensity, we express these in decibels (dB):

$$X_{dB} = 10 \log_{10} \left(\frac{X}{X_0} \right)$$

where:

- X is the actual value of the quantity being measured,
- X_0 is a specified or implied reference level,
- X_{dB} is the quantity expressed in units of decibels, relative to X_0.
- X and X_0 **must** have the same dimensions — they must measure the same type of quantity in the the same units.
- The reference level itself is **always at 0 dB** — as shown by setting $X = X_0$ (**note:** $\log_{10}(1) = 0$).

Why Use Decibel Scales?

- When there is a large range in frequency or magnitude, logarithm units often used.
- If X is greater than X_0 then X_{dB} is positive (Power Increase)
- If X is less than X_0 then X_{dB} is negative (Power decrease).
- Power Magnitude $= |X(i)|^2|$ so (with respect to reference level)

$$
\begin{aligned}
X_{dB} &= 10\log_{10}(|X(i)^2|) \\
&= 20\log_{10}(|X(i)|)
\end{aligned}
$$

which is an expression of dB we often come across.

Decibel and acoustics

- dB is commonly used to quantify sound levels relative to some 0 dB reference.
- The reference level is typically set at the *threshold of human perception*
- Human ear is capable of detecting a very large range of sound pressures.

CARDIFF UNIVERSITY PRIFYSGOL CAERDYD

Examples of dB measurement in Sound

Threshold of Pain

The ratio of sound pressure that causes **permanent** damage from short exposure to the limit that (undamaged) ears can hear is above a million:

- The ratio of the maximum power to the minimum power is above one (short scale) trillion (10^{12}).
- The log of a trillion is 12, so this ratio represents a **difference of 120 dB**.
- **120 dB** is the quoted **Threshold of Pain** for Humans.

Examples of dB measurement in Sound (cont.)

Speech Sensitivity

Human ear is not equally sensitive to all the frequencies of sound within the entire spectrum:

- Maximum human sensitivity at noise levels at between 2 and 4 kHz (Speech)
  - These are factored more heavily into sound descriptions using a process called **frequency weighting**.
  - Filter (Partition) into frequency bands concentrated in this range.
  - Used for Speech Analysis
  - Mathematical Modelling of Human Hearing
  - Audio Compression (E.g. **MPEG Audio**)

Examples of dB measurement in Sound (cont.)

Digital Noise increases by 6dB per bit

In digital audio sample representation (**linear pulse-code modulation (PCM)**),

- The first bit (least significant bit, or LSB) produces residual quantization noise (bearing little resemblance to the source signal)
- Each subsequent bit offered by the system **doubles** the resolution, corresponding to a 6 $(= 10 * \log_{10}(4))$ dB.
- So a 16-bit (linear) audio format offers 15 bits beyond the first, for a dynamic range (between quantization noise and clipping) of (15 x 6) = 90 dB, meaning that the maximum signal is 90 dB above the theoretical peak(s) of quantisation noise.
- 8-bit linear PCM similarly gives (7 x 6) = 42 dB.
- 48 dB difference between 8- and 16-bit which is (48/6 (dB)) 8 times as noisy.

Signal to Noise

Signal-to-noise ratio is a term for the power ratio between a signal (meaningful information) and the background noise:

$$SNR = \frac{P_{signal}}{P_{noise}} = \left(\frac{A_{signal}}{A_{noise}}\right)^2$$

where P is average power and A is RMS amplitude.

- Both signal and noise power (or amplitude) must be measured at the same or equivalent points in a system, and within the same system bandwidth.

Because many signals have a very wide dynamic range, SNRs are usually expressed in terms of the logarithmic decibel scale:

$$SNR_{dB} = 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) = 20\log_{10}\left(\frac{A_{signal}}{A_{noise}}\right)$$