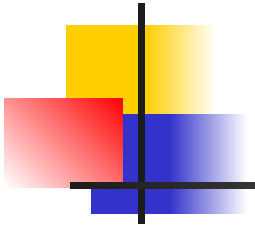
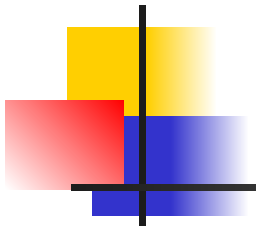
MATLAB 程式設計入門篇
音訊讀寫、錄製與播放
(Audio Reading, Writing, Recording, and Playback)

張智星 張智星 (Roger Jang)
台大資訊系 多媒體檢索實驗室
CSIE/NTU, MIR Lab



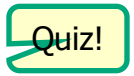
音訊的基本介紹

- 聲音訊號(Audio Signals)簡稱音訊，泛指由人耳聽到的各種聲音的訊號。
- 音訊的基本聲學特徵
  - 音量(Volume)：聲音的大小稱為音量，又稱為力度、強度(Intensity)或是能量(Energy)。音量越大，代表音訊波形的震幅越大。
  - 音高(Pitch)：聲音的基本頻率(Fundamental Frequency)越高，代表音高越高(例如女高音的歌聲)；反之，聲音的基本頻率越低，代表音高越低(例如男低音的歌聲)。
  - 音色(Timbre)：音訊波形在每個週期內的變化，就形成了此音訊的音色。不同的音色即代表不同的音訊內容，例如不同的字有不同的發音，或是不同的歌手有不同的特色，這些都是由於音色不同而產生。
- Demo via CoolEdit



Basics about Audio Signals

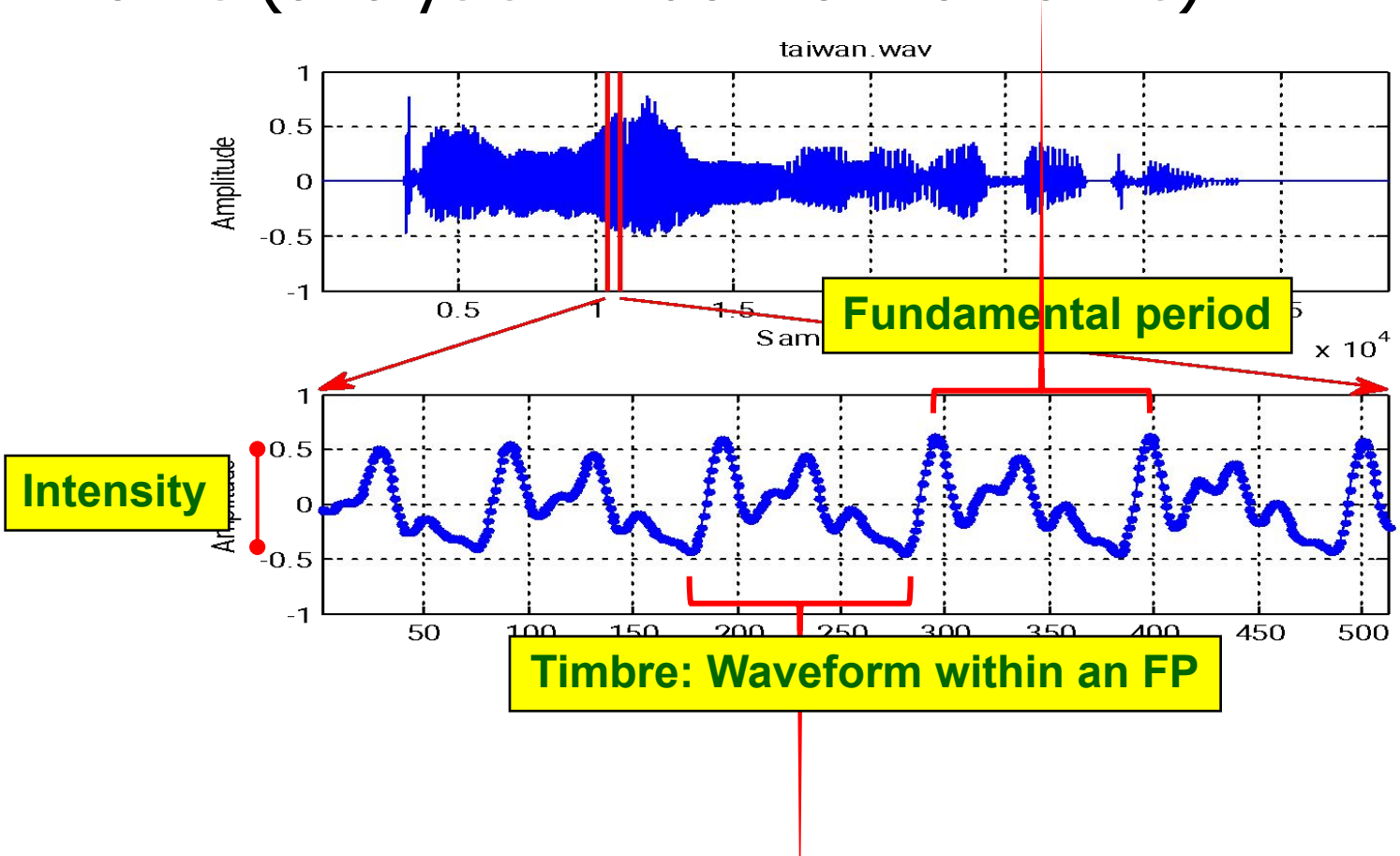
- Audio signals: signals that are audible to human
- Basic perceptible acoustic features of speech



  - Volume (音量): the amplitude of audio signals
    - Also known as intensity, or energy.
  - Pitch (音高): Fundamental frequency (the number of fundamental periods in a second) in audio signals.
    - Usually males have a lower pitch while females have a higher one
  - Timbre (音色): Waveform inside a fundamental period.
    - Different vowels have different timbres
    - Different singers also have different timbres.
- Demo via CoolEdit

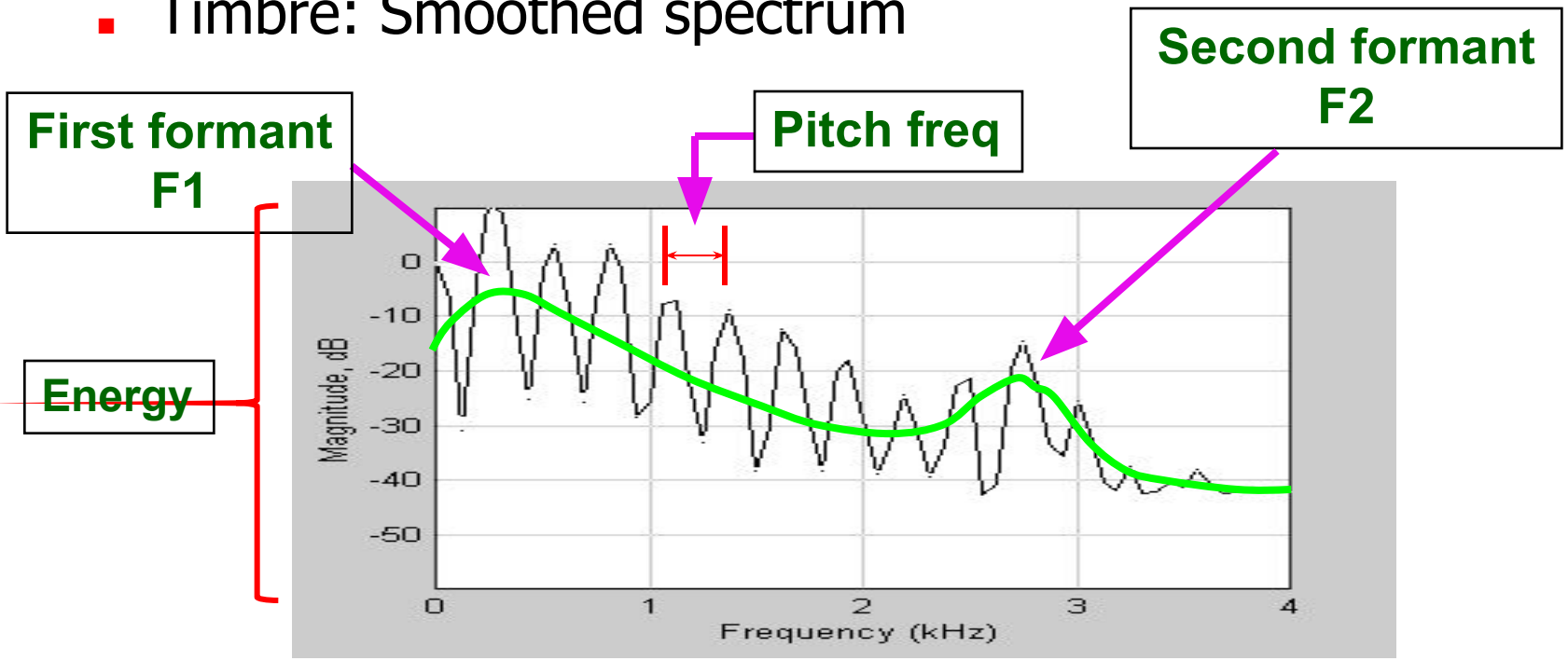
Time-domain Features

- Time-domain audio features presented in a frame (analysis window of 20-40 ms)



Frequency-domain Features

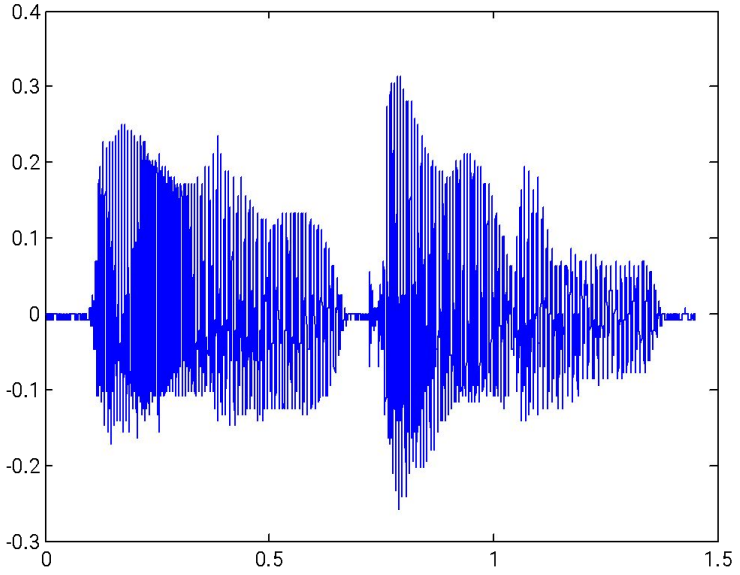
- Frequency-domain audio features in a frame
  - Energy: Sum of power spectrum
  - Pitch: Distance between harmonics
  - Timbre: Smoothed spectrum



音檔的讀取、寫檔與播放

- 使用audioread讀取wav檔案，畫出音訊的波形並播放此音訊：
  - audioRead01.m

```
[y, fs]=audioread('welcome.wav');
sound(y, fs);          % 播放此音訊
time=(1:length(y))/fs;     % 時間軸的向量
plot(time, y);         % 畫出時間軸上的波形
```

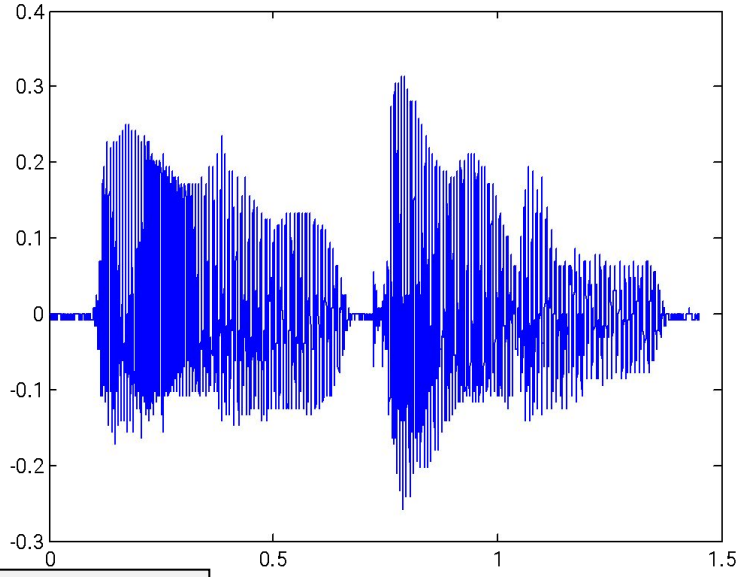


放大波形後即可看到基本週期！

Read, Write and Playback

- Use “audioread” to read a .wav file, plot its waveform and play the sound.
  - audioRead01.m

```
[y, fs]=audioread('welcome.wav');
sound(y, fs);            % Playback
time=(1:length(y))/fs;   % Time vector
plot(time, y);           % Waveform display
```



Enlarge to see fundamental periods!

中介資料

Read Metadata from .wav Files

- Reading metadata
  - info=audioInfo('file');
  - Different types of audio files may return different fields of info.
- Two types of reading data from audio files
  - Read audio signals
    - y= audioread('file')
  - Read metadata
    - info=audioinfo('file')

- Metadata of a .wav file
  - audioInfo01.m

```
fileName='flanger.wav';
info=audioinfo(fileName);
fprintf('檔案名稱 = %s\n', info.Filename);
fprintf('壓縮方式 = %s\n',
info.CompressionMethod);
fprintf('通道個數 = %g 個\n',
info.NumChannels);
fprintf('取樣頻率 = %g Hz\n', info.SampleRate);
fprintf('取樣點總個數 = %g 個\n',
info.TotalSamples);
fprintf('音訊長度 = %g 秒\n', info.Duration);
fprintf('取樣點解析度 = %g 位元/取樣點\n',
info.BitsPerSample);
```

Metadata of Other Audio Files

- *.aif
  - audioInfo02.m

```
fileName='whale.aif';
info=audioinfo(fileName);
disp(info);
```

```
          Filename: 'D:\users\...
CompressionMethod: 'Uncompressed'
      NumChannels: 1
       SampleRate: 2000
     TotalSamples: 4000
         Duration: 2
            Title: []
          Comment: []
           Artist: []
     BitsPerSample: 16
```

- *.mp3
  - audioInfo03.m

```
fileName='youAtLeast.mp3';
info=audioinfo(fileName);
disp(info);
```

```
          Filename: 'D:\users\...
CompressionMethod: 'MP3'
      NumChannels: 2
       SampleRate: 44100
     TotalSamples: 317953
         Duration: 7.2098
            Title: '02_至少還有你'
          Comment: []
           Artist: '林憶蓮'
          BitRate: 128
```

Scaling of Audio Signals by "audioread"

- Internal data types of audio signals in a file (音訊檔案內部儲存方式)
  - 8 bits ☐ uint8, [0, 2^8-1]
  - 16 bits ☐ int16, [-2^{15}, 2^{15}-1]
- MATLAB's method to scale raw audio signals to the range [-1, 1]
  - 8 bits ☐ (y-128)/128
  - 16 bits ☐ y/3276[illegible]

Original audio signals in integer

- Verification of MATLABs' scaling
  - audioRead03.m

```
fileName='welcome.wav';
[y, fs]=audioread(fileName);
info=audioinfo(fileName);
nbits=info.BitsPerSample;
% y0 是原先儲存在音訊檔案中的 值
y0=y*(2^nbits/2)+(2^nbits/2);
difference=sum(abs(y0-round(y0)))
```

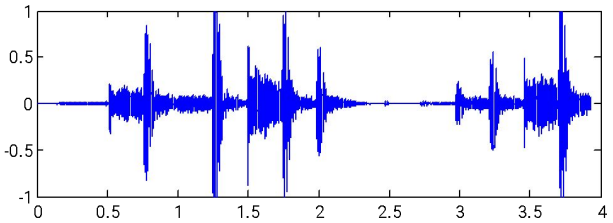
  - difference = 0

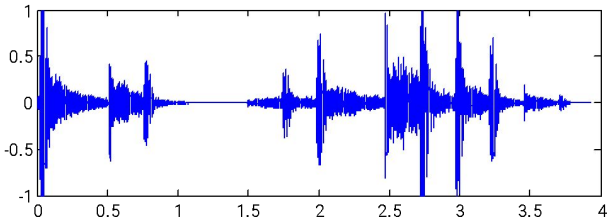
讀取雙聲道檔案

- audioread 可以讀取雙聲道或立體聲(Stereo)的音檔，此時傳回的變數具有兩直行，每一直行代表一個聲道的音訊。

- audioRead04.m

```
fileName='flanger.wav';
[y, fs]=audioread(fileName); % 讀取音訊檔
sound(y, fs);                % 播放音訊
left=y(:,1);                 % 左聲道音訊
right=y(:,2);                % 右聲道音訊
subplot(2,1,1), plot((1:length(left))/fs, left);
subplot(2,1,2), plot((1:length(right))/fs, right);
```





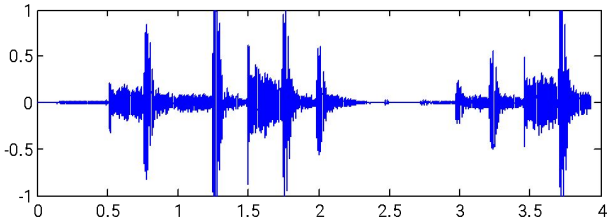
聲音會在左右喇叭游移！

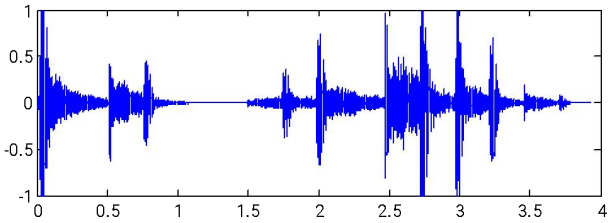
Read Stereo Audio Files

- "audioread" can also read stereo audio files. The returned variable has two columns representing two channels of audio signals.

- audioRead04.m

```
fileName='flanger.wav';
[y, fs]=audioread(fileName); % Read
sound(y, fs);              % Playback
left=y(:,1);               % Left channel
right=y(:,2);              % Right channel
subplot(2,1,1), plot((1:length(left))/fs, left);
subplot(2,1,2), plot((1:length(right))/fs, right);
```



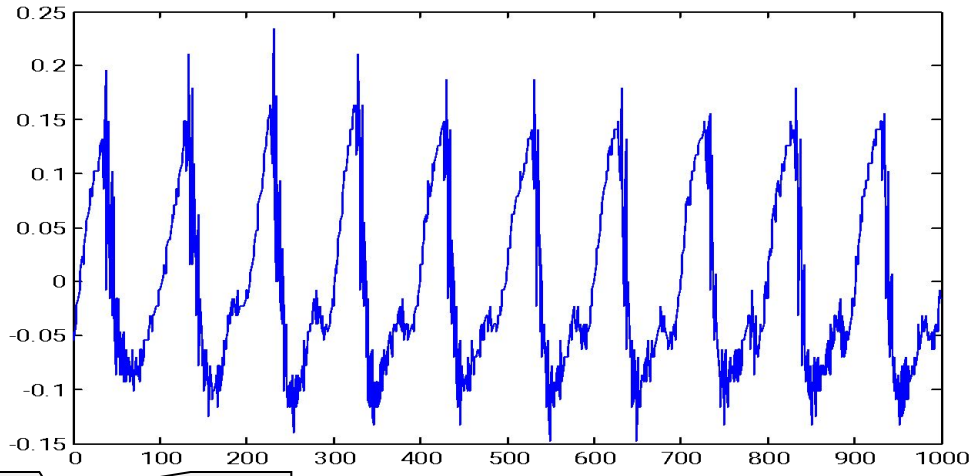


Moving sound source between two speakers!

讀取部分音檔

- 如果音檔很大，無法一次讀入記憶體，我們也可以使用 audioread 來讀出音檔的一部份，例如：
- audioRead05.m

```
[y,fs]=audioread('welcome.wav', [4001 5000]);     % 讀取第4001至5000點
figure; plot(y)
```

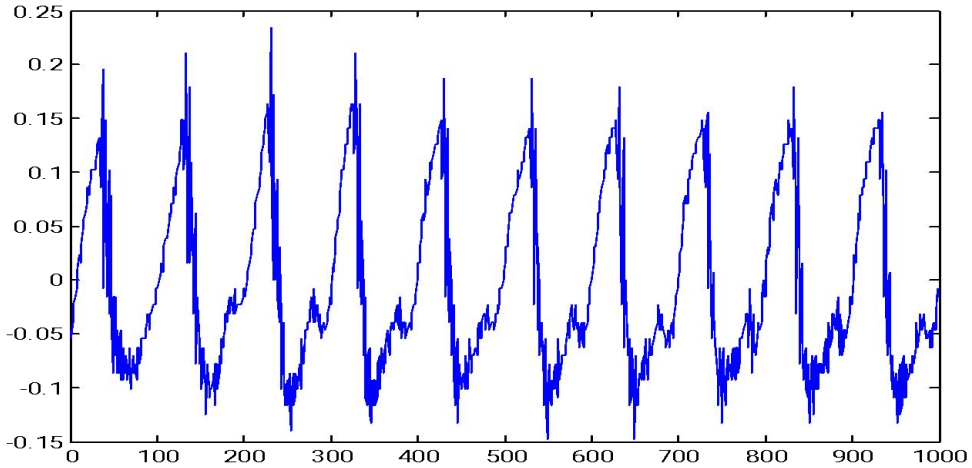


可看到明顯的基本週

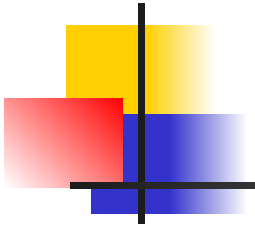
Read a Portion Only

- If the audio file is too big, we can read a portion from the whole file:
- audioRead05.m

```
[y,fs]=audioread('welcome.wav', [4001 5000]);     % Read data points 4001 to 5000
figure; plot(y)
```

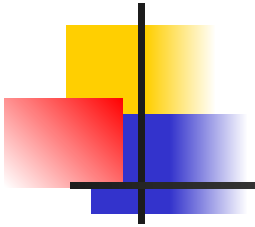


Obvious fundamental periods!



聲音訊號的播放

- 一旦我們可以讀入音訊檔案，就可以對聲音訊號進行各種處理，例如增大或減小音量、提高或降低音高、消除雜訊等。
- 要確認處理後的聲音訊號是否符合所需，就要能夠把音訊直接透過接到電腦的喇叭播放出來，本節就是要介紹如何使用 MATLAB 來進行音訊的播放。



Playback of Audio Signals

- Once we have read audio signals, we can perform all kinds of processing, such as volume modification, pitch scaling, noise reduction, etc.
- To verify the result, you need to play the audio via speakers, as shown in the following examples.

播放聲音 (1/2)

- 一旦 MATLAB 讀入音訊資料，並將之設定成工作空間中的變數後，我們就可以使用 sound 指令來直接播放此變數。

- 播放單一聲音
  - audioPlay01.m

```
load handel.mat      % 載入音訊
sound(y, Fs);   % 播放音訊
```

- 同時播放兩種聲音
  - audioPlay02.m

```
[y, fs]=audioread('welcome.wav');  % 載入音訊
sound(5*y, fs);% 播放音訊
load handel.mat      % 載入音訊
sound(y, Fs);   % 播放音訊
```

sound的播放模式為「非同步播放」！

Playback (1/2)

- We can use "sound" to play audio signals that has been read and stored as a variable in MATLAB's workspace.

- Audio playback
  - audioPlay01.m

```
load handel.mat      % Load audio
sound(y, Fs);   % Playback
```

- Simultaneous playback
  - audioPlay02.m

```
[y, fs]=audioread('welcome.wav');  % Read audio
sound(5*y, fs);% Playback
load handel.mat      % Load audio
sound(y, Fs);   % Playback
```

Playback mode of "sound" is asynchronous!

播放聲音 (2/2)

- 若要控制聲音的播放模式, 則必須採用功能較為強大的指令：
  - audioplayer
  - play
  - playblocking

- 播放單一聲音
  - audioPlay03.m

```
load handel.mat      % 載入音訊
p=audioplayer(y, Fs);% 產生播放物件
play(p);             % 播放音訊
```

- 循序播放兩種聲音
  - audioPlay04.m

```
[y, fs]=audioread('welcome.wav');  % 讀入音訊
p=audioplayer(y, fs);      % 產生播放物件
playblocking(p);           % 播放音訊
load handel.mat            % 載入音訊
p=audioplayer(y, Fs);      % 產生播放物件
playblocking(p);           % 播放音訊
```

playblocking 的播放模式為「同步播放」！

Playback (2/2)

- If you want to control the playback mode, you need to invoke other commands:
  - audioplayer
  - play
  - playblocking

- Single playback
  - audioPlay03.m

```
load handel.mat      % Load audio
p=audioplayer(y, Fs);% Player object
play(p);             % Playback
```

- Sequential playback
  - audioPlay04.m

```
[y, fs]=audioread('welcome.wav');  % Read audio
p=audioplayer(y, fs);      % Player object
playblocking(p);           % Playback
load handel.mat            % Load audio
p=audioplayer(y, Fs);      % Player object
playblocking(p);           % Playback
```

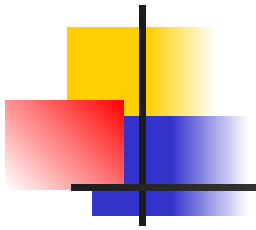
Playback mode of "playblocking" is synchronous!

改變音訊的震幅

- 我們在第一節提到過，聲音的音量是由聲波的震幅來決定，因此我們可藉由震幅的大小來改變音量，例如:
- playVolume01.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(1*y, fs); playblocking(p);      % 播放 1 倍震幅的音訊
p=audioplayer(3*y, fs); playblocking(p);      % 播放 3 倍震幅的音訊
p=audioplayer(5*y, fs); playblocking(p);      % 播放 5 倍震幅的音訊
```

聲音聽起來並沒有變成3或5倍大聲，為什麼？



Change of Audio Amplitude

- Volume of audio signals is determine by their amplitude. Here are modify amplitude to change the volume:
- playVolume01.m

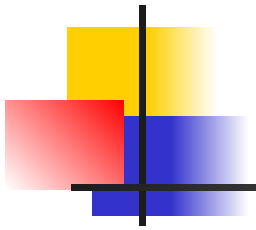
```
[y, fs]=audioread('welcome.wav');
p=audioplayer(1*y, fs); playblocking(p);      % Original audio
p=audioplayer(3*y, fs); playblocking(p);      % Audio of 3x amplitude
p=audioplayer(5*y, fs); playblocking(p);      % Audio of 5x amplitude
```

The playback doesn't sound like 5-times louder, why?

改變音訊播放的取樣率 (1/2)

- 如果在播放時，改變取樣頻率，就會改變整個音訊的時間長度，進而影響到音高。
- 在下例中，我們漸漸提高播放時的取樣頻率，聽到的聲音就會越來越快、越來越高，最後出現像唐老鴨的聲音。為什麼？
- playFs01.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs);
p.SampleRate=1.0*fs; playblocking(p);   % 播放 1.0 倍速度的音訊
p.SampleRate=1.2*fs; playblocking(p);   % 播放 1.2 倍速度的音訊
p.SampleRate=1.5*fs; playblocking(p);   % 播放 1.5 倍速度的音訊
p.SampleRate=2.0*fs; playblocking(p);   % 播放 2.0 倍速度的音訊
```



Change of Sample Rates (1/2)

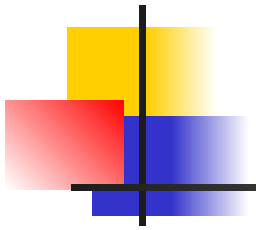
- Change of sample rate during playback ☐ Change of duration ☐ Change of the perceived pitch
- Increase the sample rate during playback, and you'll hear Donald Duck (唐老鴨). Why?
- playFs01.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs);
p.SampleRate=1.0*fs; playblocking(p);   % Duration ratio: 1
p.SampleRate=1.2*fs; playblocking(p);   % Duration ratio: 1/1.2
p.SampleRate=1.5*fs; playblocking(p);   % Duration ratio: 1/1.5
p.SampleRate=2.0*fs; playblocking(p);   % Duration ratio: 1/2
```

改變音訊播放的取樣率 (2/2)

- 反之，如果漸漸降低播放的頻率，聽到的聲音就會越來越慢、越來越低，最後出現像牛叫的聲音。
- playFs02.m

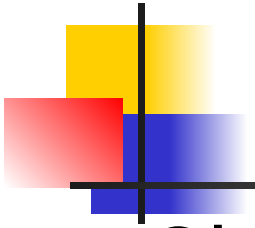
```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs);
p.SampleRate=1.0*fs; playblocking(p);   % 播放 1.0 倍速度的音訊
p.SampleRate=0.9*fs; playblocking(p);   % 播放 0.9 倍速度的音訊
p.SampleRate=0.8*fs; playblocking(p);   % 播放 0.8 倍速度的音訊
p.SampleRate=0.6*fs; playblocking(p);   % 播放 0.6 倍速度的音訊
```



Change of Sample Rates (2/2)

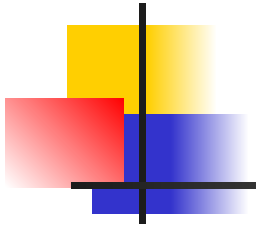
- On the other hand, decrease the sample rate during playback, and you'll hear cow moo...
- playFs02.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs);
p.SampleRate=1.0*fs; playblocking(p);   % Duration ratio: 1
p.SampleRate=0.9*fs; playblocking(p);   % Duration ratio: 1/0.9
p.SampleRate=0.8*fs; playblocking(p);   % Duration ratio: 1/0.8
p.SampleRate=0.6*fs; playblocking(p);   % Duration ratio: 1/0.6
```



Observations

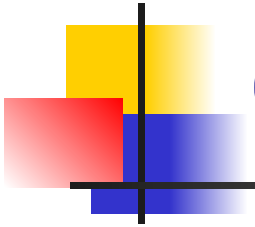
- Observations Quiz!
  - Larger sample rate for playback leads to…
    - Shorter duration and higher pitch
  - Smaller sample rate for playback leads to...
    - Longer duration and lower pitch
- How to…
  - Generate higher pitch without duration change?
    - ☐ Pitch modification
  - Create longer duration without pitch change?
    - ☐ Duration modification
  - Demo



改變符號及改變時序

- 如果我們將聲波訊號上下顛倒，聽到的聲音基本上是一樣的，但是如果前後顛倒，聽到的聲音就如同錄音帶「倒放」的聲音，聽起來很像是某種外國語音，請試試下列範例：
- 範例20-11：playReverse01.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs); playblocking(p);          % 播放正常的音訊波形
p=audioplayer(-y, fs); playblocking(p);         % 播放上下顛倒的音訊波形
p=audioplayer(flipud(y), fs); playblocking(p); % 播放前後顛倒的音訊波形
```



Change of Sign & Time Sequence

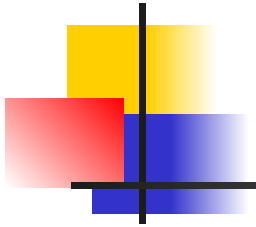
- Change of sign ☐ No perceptional difference
- Reverse sequence ☐ Sounds like another spoken language?
- 範例20-11：playReverse01.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs); playblocking(p);          % Normal playback
p=audioplayer(-y, fs); playblocking(p);         % Change of sign
p=audioplayer(flipud(y), fs); playblocking(p); % Reverse the sequence
```

同步及非同步播放

- 通常在播放音訊時，MATLAB 停止進行其他動作，直到音訊播放完畢後，才會再進行其他指令的運算，此種運作方式稱為「同步式」(Synchronous)。若需要一邊播放、一邊進行其他運算，就必須使用「非同步式」(Asynchronous)的播放方式。
- 範例20-12：playSync01.m

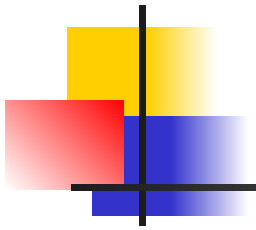
```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs);
playblocking(p);      % 同步播放 1.0 倍速度的音訊
sound(y, 0.8*fs);     % 非同步播放 0.8 倍速度的音訊
sound(y, 0.6*fs);     % 非同步播放 0.6 倍速度的音訊
```



非同步播放

- 在此例中，我們會聽到類似男女兩部合唱，一快一慢，這是因為 sound 指令的預設播放方式就是「非同步」。
- 範例20-13：playSync02.m

```
load handel.mat
sound(y, Fs);
sound(y, 1.2*Fs);
```



Playback Modes

- There are two playback modes
  - Synchronous mode: Block everything till the end of playback.
  - Asychronous mode: Nonblocking
- playSync01.m

```
[y, fs]=audioread('welcome.wav');
p=audioplayer(y, fs);
playblocking(p);      % Synchronous
sound(y, 0.8*fs);     % Asynchronous
sound(y, 0.6*fs);     % Asynchronous
```

- playSync02.m

```
load handel.mat
sound(y, Fs);
sound(y, 1.2*Fs);
```

音量自動調整

- 另一個類似的指令是 soundsc，可針對音訊變數的數值先進行正規化(介於 –1 和 1 中間)後，再送到喇叭播放，以達到最好的播放效果。
- soundsc01.m

```
[y, fs]=audioread('welcome.wav');
sound(y, fs);
fprintf('Press any key to continue...\n'); pause
soundsc(y, fs);
```

在影像顯示方面，對應的命令是 imagesc。

Automatic Volume Adjustment

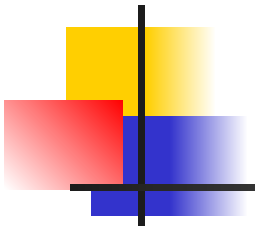
- "soundsc" adjusts the volume (by normalizing the signals to have max of 1 or min of -1) before playback
- soundsc01.m

```
[y, fs]=audioread('welcome.wav');
sound(y, fs);
fprintf('Press any key to continue...\n'); pause
soundsc(y, fs);
```

The corresponding command for image display is "imagesc".

聲音訊號的錄製

- 我們在第一節已經說明了如何讀取音訊檔案，並在第二節說明如何播放。MATLAB 也支援直接由麥克風讀取訊號，因此可以直接進行聲音的錄製，所使用的指令是
  - audiorecorder
  - recordblocking



Recording of Audio Signals

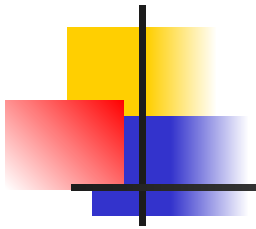
- We can use the following MATLAB commands for recording from the microphone directly:
  - audiorecorder
  - recordblocking

音訊的錄製範例 (1/2)

- 使用預設參數, 由麥克風進行3秒錄音:
  - audioRecord01.m

```
duration=3;            % 錄音時間
recObj=audiorecorder;
fprintf('按任意鍵後開始 %g 秒錄音:', duration); pause
fprintf('錄音中...');
recordblocking(recObj, duration);
fprintf('錄音結束\n');
fprintf('按任意鍵後開始播放:'); pause
play(recObj);
```

  - 預設錄音參數
    - 取樣頻率為 8000 Hz
    - 取樣點解析度為 8 bits
    - 單聲道錄音



Recording (1/2)

- Use default setting for 3-sec recording:
  - audioRecord01.m

```
duration=3;            % Duration of recording
recObj=audiorecorder;
fprintf('按任意鍵後開始 %g 秒錄音:', duration); pause   % Prompt
fprintf('錄音中...');          % During recording
recordblocking(recObj, duration);
fprintf('錄音結束\n');       % End of recording
fprintf('按任意鍵後開始播放:'); pause     % Press any key for playback
play(recObj);
```

  - Default settings for recording
    - Sample rate: 8000 Hz
    - Bit resolution: 8 bits
    - Mono

音訊的錄製範例 (2/2)

- 設定各項錄音參數來進行3秒錄音，並畫出波形：
  - audioRecord02.m

```
fs=16000;       % 取樣頻率
nBits=16;       % 取樣點解析度，必須是 8 或 16 或 24
nChannel=1;     % 聲道個數，必須是1(單聲道)或2(雙聲道或立體音)
duration=3;     % 錄音時間(秒)
recObj=audiorecorder(fs, nBits, nChannel);
fprintf('按任意鍵後開始 %g 秒錄音：', duration); pause
fprintf('錄音中...');
recordblocking(recObj, duration);
fprintf('錄音結束\n');
fprintf('按任意鍵後開始播放：'); pause
play(recObj);
y = getaudiodata(recObj, 'double'); % get data as a double array
plot((1:length(data))/fs, y);
xlabel('Time (sec)'); ylabel('Amplitude');
```

設定錄音參數

取得音訊資料

Recording (2/2)

- Set recording parameters, record, plot the waveform:
  - audioRecord02.m

```
fs=16000;       % Sample rate
nBits=16;       % Bit resolution (must be 8, 16, or 24)
nChannel=1;     % No. of channels (must be 1 or 2)
duration=3;     % Duration for recording in sec
recObj=audiorecorder(fs, nBits, nChannel);
fprintf('按任意鍵後開始 %g 秒錄音:', duration); pause
fprintf('錄音中...');
recordblocking(recObj, duration);
fprintf('錄音結束\n');
fprintf('按任意鍵後開始播放:'); pause
play(recObj);
y = getaudiodata(recObj, 'double'); % get data as a double array
plot((1:length(data))/fs, y);
xlabel('Time (sec)'); ylabel('Amplitude');
```

Set up recording parameters

Obtain audio signals

聲音訊號的寫檔 (1/2)

- 我們也可以經由 MATLAB 將音訊資料直接儲存為音訊檔案，以便直接在電腦播放。寫入音訊檔案的指令是 audiowrite，其格式為：
  - audiowrite(audioFile, y, fs)
    - audioFile 則是欲寫入資料的檔案名稱，y 是音訊變數，fs 是取樣頻率。
  - 範例：audioWrite01.m

```
load handel.mat
audioFile='handel.wav';   % 欲儲存的 wav 檔案
fprintf('Saving to %s...\n', audioFile);
audiowrite(audioFile, y, round(1.5*Fs));
fprintf('按任意鍵後開始播放 %s...\n', audioFile);
dos(['start ', audioFile]);  % 開啟與 wav 檔案對應的應用程式
```

Storing Audio Files (1/2)

- We can use “audiowrite” to save audio files, with the following I/O format:
  - audiowrite(audioFile, y, fs)
    - audioFile: file to write to, y: audio signals, fs: sample rate
  - 範例：audioWrite01.m

```
load handel.mat
audioFile='handel.wav';   % wav file to write to
fprintf('Saving to %s...\n', audioFile);
audiowrite(audioFile, y, round(1.5*Fs));
fprintf('按任意鍵後開始播放 %s...\n', audioFile);
dos(['start ', audioFile]);   % Use default application to open the wav file
```

聲音訊號的寫檔 (2/2)

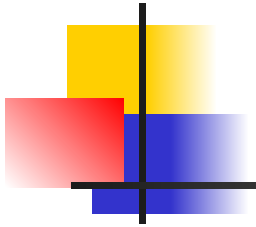
- 錄音、播放、存檔的範例：
  - 範例：audioWrite02.m

```
fs=16000;  % 取樣頻率
nBits=16;   % 取樣點解析度, 必須是 8 或 16 或 24
nChannel=1;       % 聲道個數, 必須是 1(單聲道) 或 2(雙聲道或立體音)
duration=3;       % 錄音時間(秒)
recObj=audiorecorder(fs, nBits, nChannel);
fprintf('按任意鍵後開始 %g 秒錄音：', duration); pause
fprintf('錄音中...');
recordblocking(recObj, duration);
fprintf('錄音結束\n');
fprintf('按任意鍵後開始播放：\n'); pause
y = getaudiodata(recObj, 'double');        % get data as a double array
plot((1:length(data))/fs, y); xlabel('Time (sec)'); ylabel('Amplitude');
sound(y, fs);
audioFile='myRecording.wav';       % 欲儲存的 wav 檔案
fprintf('Saving to %s...\n', audioFile);
audiowrite(audioFile, y, fs);
system(audioFile);       % Use default application to open the wav file
```

Storing Audio Files (2/2)

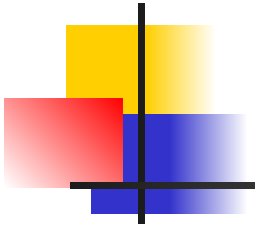
- Example of recording, playback, and saving:
  - 範例：audioWrite02.m

```
fs=16000;  % Sample rate
nBits=16;   % Bit resolution (must be 8, 16, or 24)
nChannel=1;       % No. of channels (must be 1 or 2)
duration=3;       % Recording duration in sec
recObj=audiorecorder(fs, nBits, nChannel);
fprintf('按任意鍵後開始 %g 秒錄音：', duration); pause
fprintf('錄音中...');
recordblocking(recObj, duration);
fprintf('錄音結束\n');
fprintf('按任意鍵後開始播放：\n'); pause
y = getaudiodata(recObj, 'double');       % get data as a double array
plot((1:length(data))/fs, y); xlabel('Time (sec)'); ylabel('Amplitude');
sound(y, fs);
audioFile='myRecording.wav';       % wav file to be saved
fprintf('Saving to %s...\n', audioFile);
audiowrite(audioFile, y, fs);
system(audioFile);       % 開啟與 wav 檔案對應的應用程式
```



Cross-version Issues

- File mapping for different versions of MATLAB
  - wavread ☐ audioread
  - wavwrite ☐ audiowrite
  - wavplay ☐ audioplayer, sound
  - wavrecord ☐ audiorecorder
- Other supports
  - audiodevinfo
  - playblocking
  - play
- SAP toolbox
  - Version-independent audio file reading ☐ myAudioRead.m
  - Progressive bar ☐ audioPlayWithBar.m



Supplementary Material

- Other resources
  - ASPR: Audio Signal Processing and Recognition
    - Texts for this set of slides can be found at Chapter 4.
    - Pitch tracking by visual inspection can be found at Section 4 of Chapter 5. (Slides)