南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Signals and Systems (Lab)

Lab 5: System, Convolution and Filter

Dr. Wu Guang
wug@sustc.edu.cn
Electrical & Electronic Engineering
Southern University of Science and Technology

Low-Complexity method

- Analysis in **time-domain**

$$x(t) \Rightarrow \boxed{h(t)} \Rightarrow y(t)$$

$$x(t) * h(t) = y(t)$$

Convolution

VS

- Analysis in **frequency-domain**

$$X(j\omega) \Rightarrow \boxed{H(j\omega)} \Rightarrow Y(j\omega)$$

$$X(j\omega) \cdot H(j\omega) = Y(j\omega)$$

Multiplication

Summary

- Analysis in **time-domain**

$$x(t) \Rightarrow \boxed{h(t)} \Rightarrow y(t)$$

- DT LTI system: Difference Equation
- CT LTI system: Differential Equation
- FIR: Finite Impulse Response
- IIR: Infinite Impulse Response
- Linear convolution
- Signal Auto-Correlation
- Complex Exponentials
- Eigenvalue, eigenvector and eigenfunction

VS

- Analysis in **frequency-domain**

$$X(j\omega) \Rightarrow \boxed{H(j\omega)} \Rightarrow Y(j\omega)$$

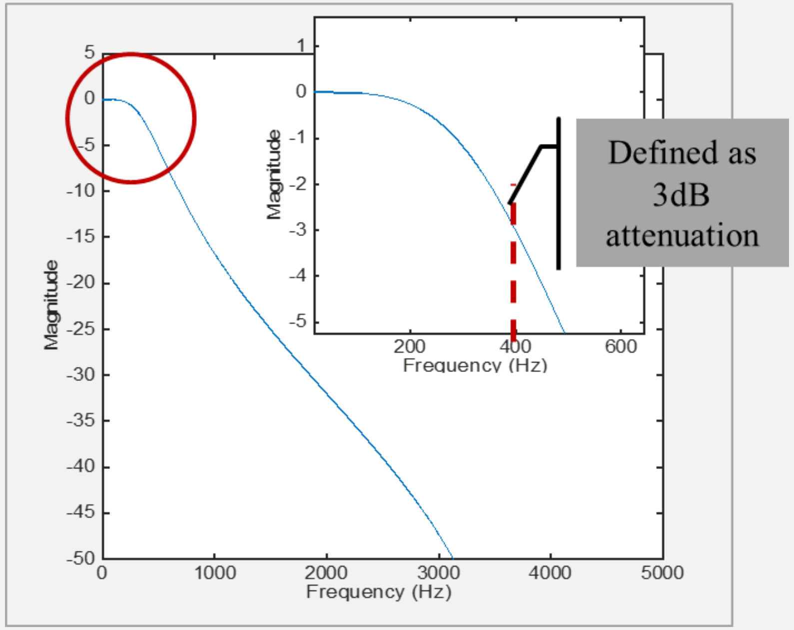
- FT: Fourier Transform
- FS: Fourier series
- DTFT: Discrete time Fourier Transform
- DFT: Discrete Fourier Transform
- FFT: Fast Fourier Transform
- Periodic convolution
- Frequency Response
- The system transfer function

Overview

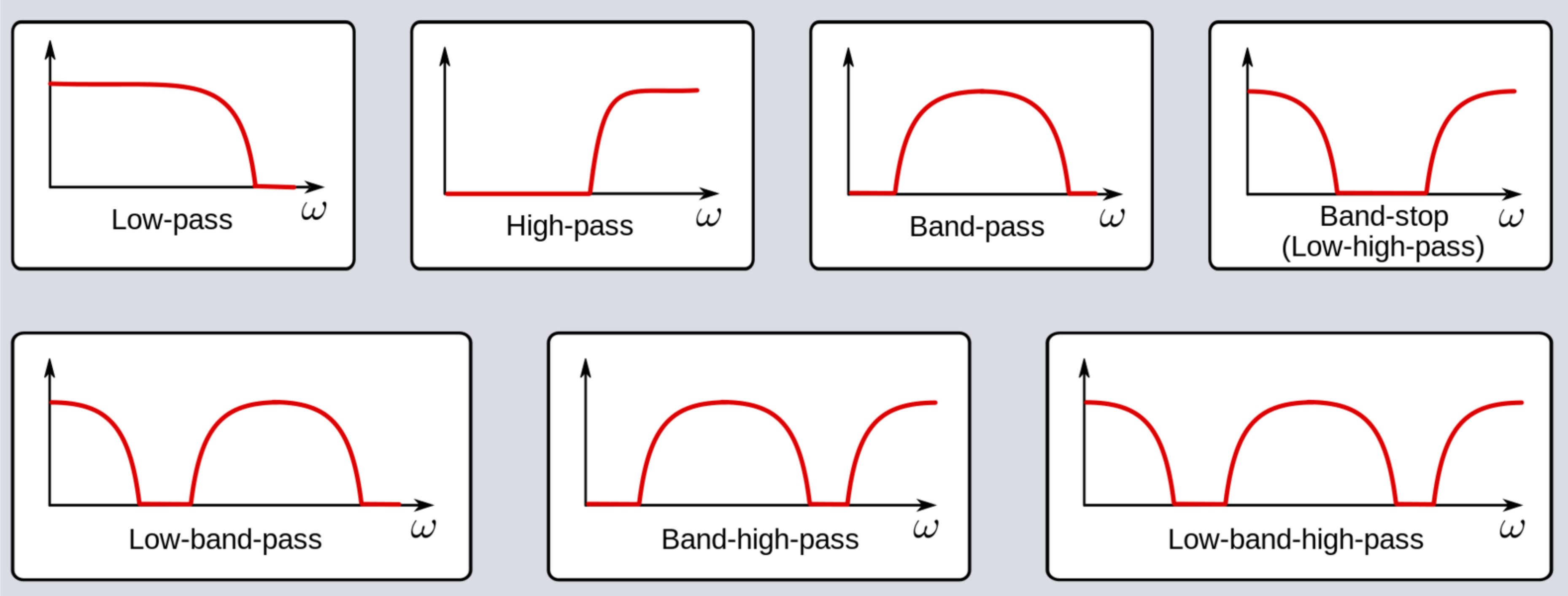
1. Design basic filters via Matlab

2. Use filtering to
-> generate speech shaped noise
-> and extract signal envelope

3. Adjust the signal intensity
-> adjusting signal-to-noise ratio (SNR) level
-> normalizing signal energy



Filter type



Low-pass
ω
High-pass
ω
Band-pass
ω
Band-stop
(Low-high-pass)
ω
Low-band-pass
ω
Band-high-pass
ω
Low-band-high-pass
ω

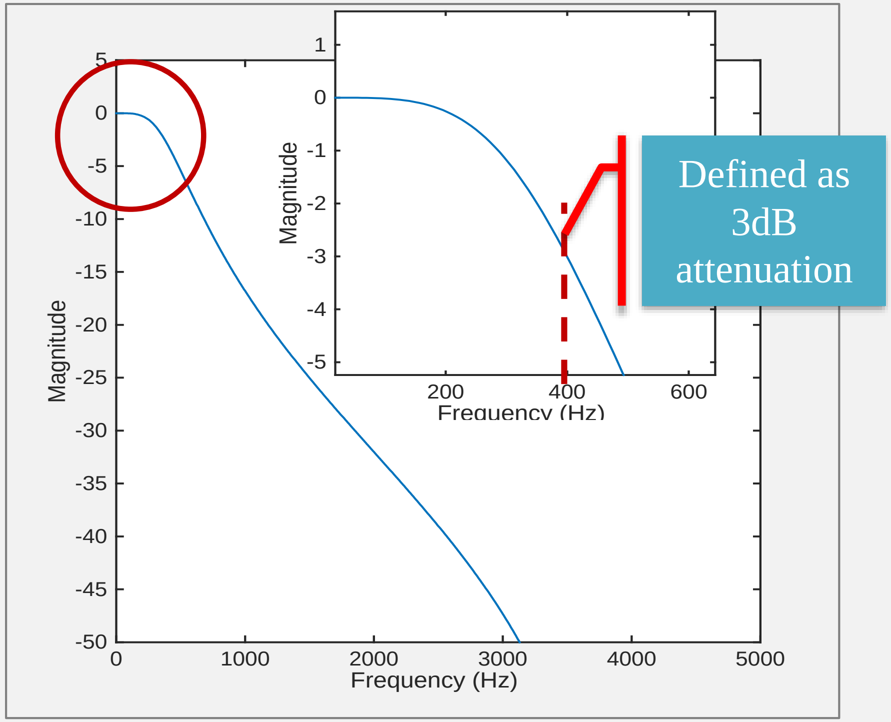
Part 1.Butterworth filter design

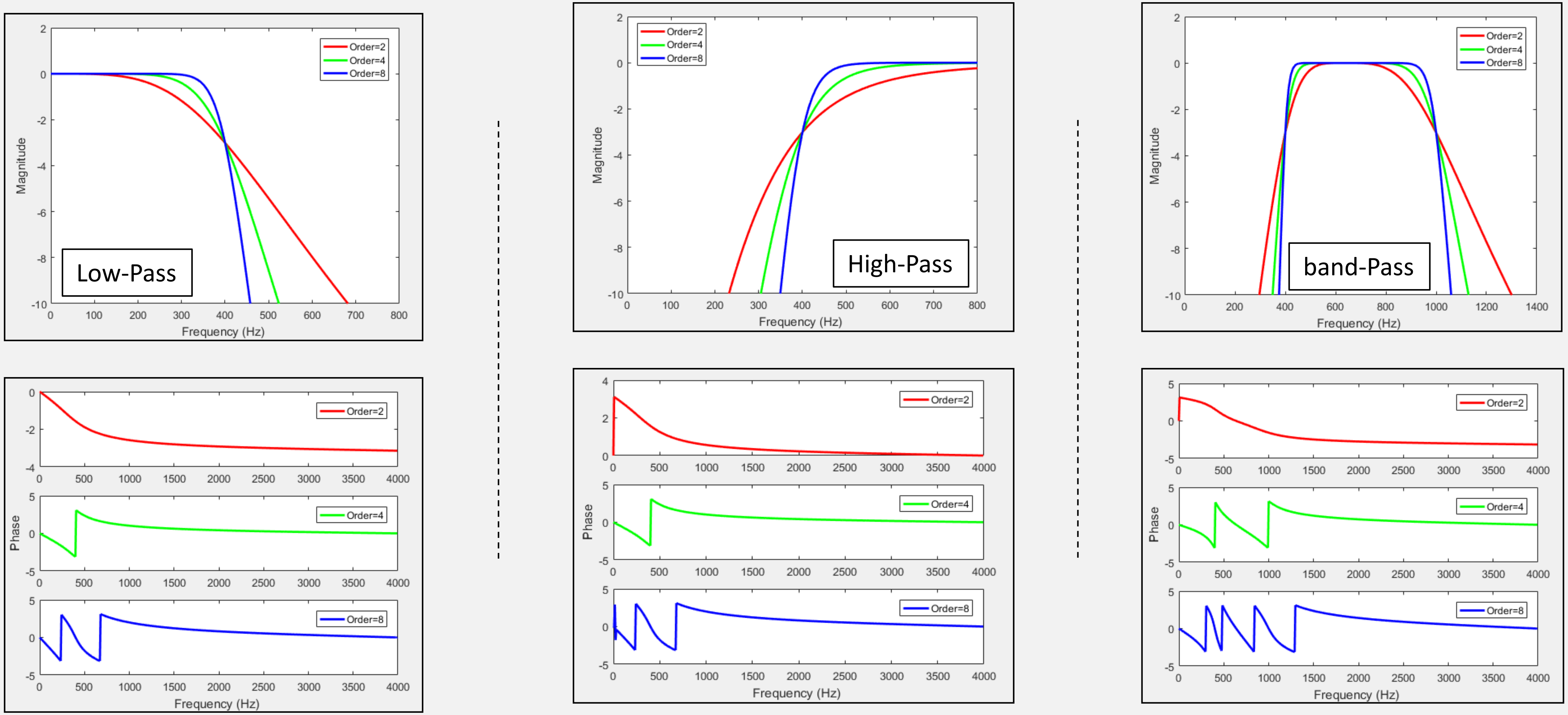
- **[b,a]=butter(n,Wn)**
  - Design an order n low-pass digital Butterworth filter with ***normalized cutoff frequency*** Wn. Wn must be 0.0 < Wn < 1.0, with 1.0 corresponding to half the sample rate.
  - Return the filter coefficients in length n+1 vectors b and a.
- **[b,a]=butter(n,Wn,'ftype')**
  - Design a highpass, lowpass, or bandstop filter, where the string ‘ftype’ is ‘high’, ‘low’, or ‘stop’,
  - e.g.,

```
fs=8000; [b, a]=butter(2,400/(fs/2));  % low-pass
[b, a]=butter(2,400/(fs/2),’high’); % high-pass
[b, a]=butter(2,[400 1000]/(fs/2));  % band-pass
```

- fs=8000; % sampling rate for discrete signal
- [b, a]=butter(2,400/(fs/2));
- [h,f]=freqz(b,a,512,fs); % Digital filter frequency response
- plot(f,20*log10(abs(h))); % in dB scale
- axis([0 5000 -50 5]);
- xlabel('Frequency (Hz) ');
- ylabel('Magnitude');

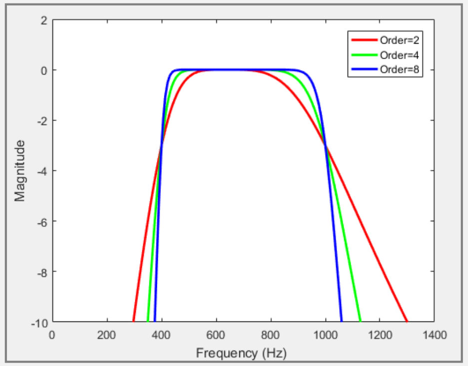
How about 'high-pass' and 'band-pass'?
How about high-order n?



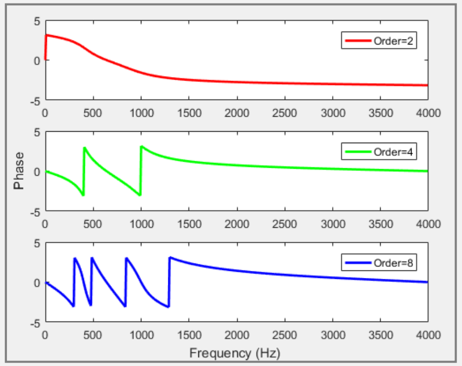


Low-Pass
High-Pass
band-Pass
Magnitude
Phase
Frequency (Hz)
Order=2
Order=4
Order=8

Quiz



Order=2
Order=4
Order=8
Magnitude
Frequency (Hz)

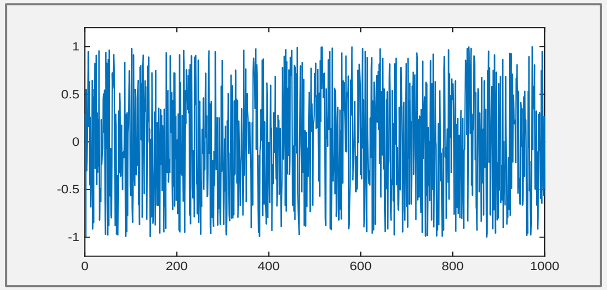


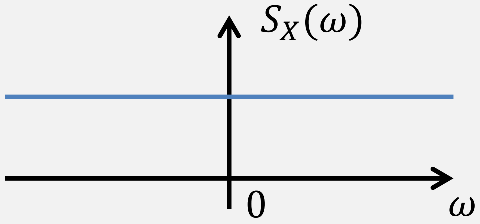
Order=2
Order=4
Order=8
Phase
Frequency (Hz)

Part 2. Use filter to ...

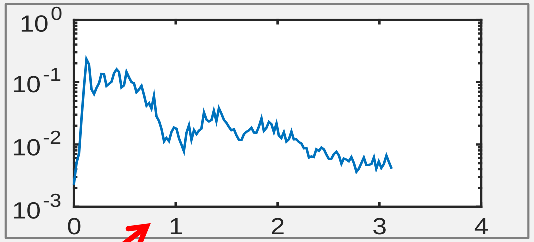
- **Generate speech-shaped noise**
  - White noise:

```
N=1000; noise = 1-2*rand(1,N);
```





  - Speech-shaped noise:



White noise → $H(j\omega)$ e.g. speech spectrum → Speech-shaped noise

Step 1

(a) Generate long-term spectrum of speech signal

- concatenate 10 speech waveforms into one signal
  - sig =[x,x,x,x,x,x,x,x,x,x]; % x is a row vector, what if x is a column vector?
  - Or, **repmat**(x,1,10)
- estimate the power spectral density of the speech signal
  - [Pxx,w] =**psd**(sig,512,fs);
  - Or [Pxx,w] =**pwelch**(sig, [], [], 512, fs)
- plot(w,Pxx)

Step 2

(b) Generate filter coefficients

- b = **fir2**(3000,w/(fs/2),sqrt(Pxx/max(Pxx))); % fir2: from frequency response to coefficients, a=1
- [h,wh] = freqz(b,1,128); %check frequency response here

(c) Perform filtering for white noise signal

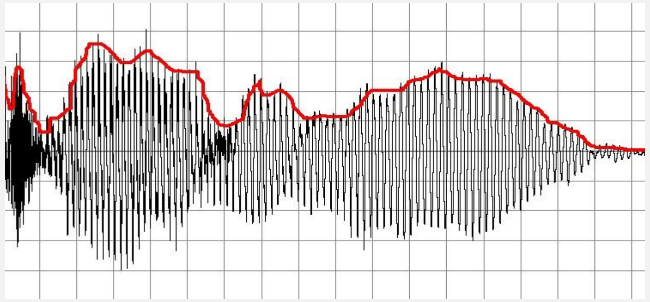
- y= filter(b,1,noise);

$$H(j\omega) = \frac{\sum_{k=0}^{N} b_k (j\omega)^k}{\sum_{k=0}^{M} a_k (j\omega)^k}$$

(d) Then you get the speech-shaped noise

Then, we use filter to ...

- **Extract speech envelope waveform**



Original waveform

Envelope

https://commons.wikimedia.org/wiki/File%3AC_Envelope_follower.jpg

Signal → abs(·) → Low-pass filtering → envelope

rectification
1) full wave: abs(·)
2) half wave: if s<0, s=0

Part 3. Adjust Signal Intensity

- **Signal-to-noise ratio (SNR)**
  - $SNR = 10log_{10}\left(E_{signal}/E_{noise}\right)$
  - *E* is energy, and *SNR* in dB scale
  - In Matlab: SNR=20*log10(norm(sig)/norm(noise)) % sig and noise must have the same length

n = norm(X) returns the 2-norm of input X and is equivalent to norm(X,2). If X is a vector, norm(X) is the **Euclidean distance**.

$\sqrt{E}$

- **Energy normalization**
  - y'=y*norm(x)/norm(y); % then y has the same energy as x

- **How to adjust SNR when adding noise to a signal?**
  - Adjust the energy of noise first
  - $E_{signal}/E_{noise} = 10^{\frac{SNR}{10}} \Rightarrow E_{noise} = f(SNR, E_{signal})$
  - norm(sig)/norm(noise) = ?
  - After adjusting the noise energy, test if the SNR is the desired value.

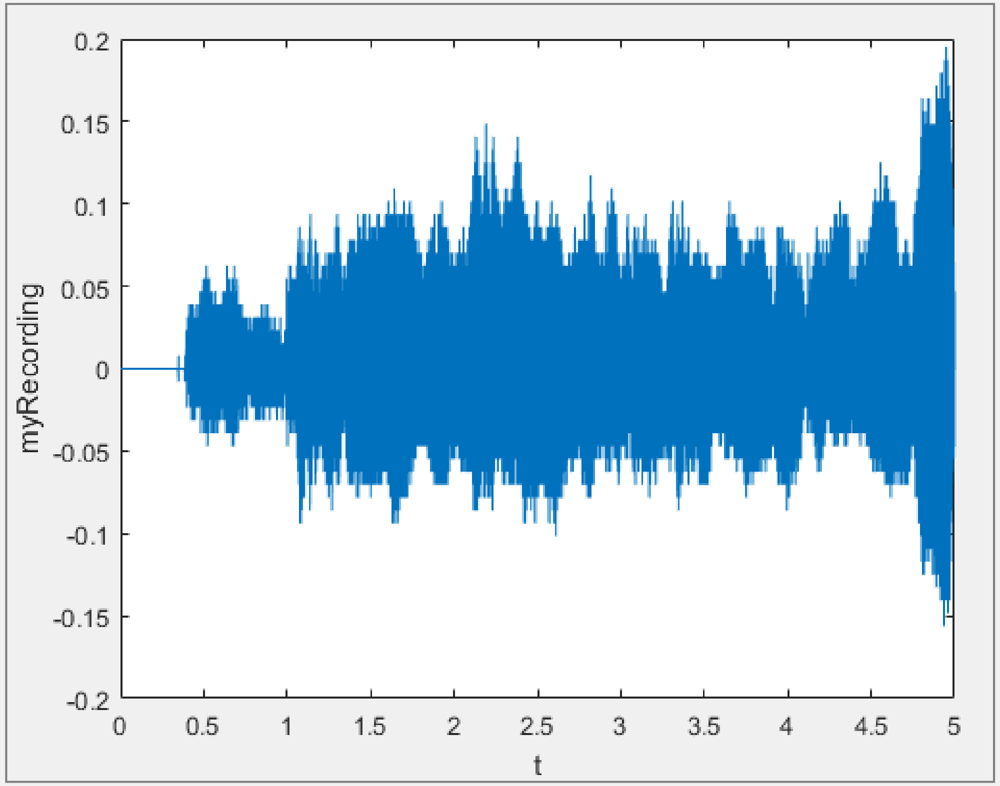
More on wav file ...

- **Read wave file (‘*.wav’)**
  - [y, fs, nbits]=wavread(filename) % before 2014
    - y: signal, fs: sampling rate in Hertz, nbits: the number of bits per sample used to encode the data (e.g., 8, 16, 32)
  - Or, [y, fs] = audioread(filename)
- **Play wave file**
  - wavplay(y, fs), or sound(y, fs)
  - Or,
    - player = audioplayer(y, fs)
    - play(player)

- **Save wave file**
  - wavwrite(y,fs,nbits,filename)
    - writes data y to a Windows WAVE file specified by the file name filename, with a sample rate of fs Hz and with nbits number of bits. nbits must be 8, 16, 24, or 32.
  - Or, audiowrite(filename,y,fs)
- **Record sound using PC-based audio input device**
  - y = wavrecord(n,Fs)
  - Or, ‘audiorecorder’

- **An example**

```
% Record your voice for 5 seconds.
recObj = audiorecorder(11025,8,1);
disp('Start speaking.')
recordblocking(recObj,5);
disp('End of Recording.');
% Play back the recording.
play(recObj);
% Store data in double-precision array
myRecording = getaudiodata(recObj);
plot(myRecording); % Plot the waveform.
```



See Matlab documentation for more details

Assignment

1. Generate a speech-shaped noise (SSN), and plot the spectra of the speech signal and SSN (e.g., use Matlab function ‘psd’ or ‘pwelch’, or other power spectral density estimation functions).
2. Read a speech signal x(t), adjust the SNR (x(t) to the above SSN) to -5 dB, let $y = x + SSN$, and normalize the energy of y in relative to x(t), i.e., modify the energy of y so that it equals to the energy of x.

Assignment (cont.)

3. Extract speech envelope
   1) with 2^{nd}-order low-pass filter and cutoff frequency f_{cut}= 100, 200, and 300 Hz. Plot these three envelope waveforms in one plot, and describe the difference among them.
   2) with 2^{nd} and 6^{th}-order low-pass filter and cutoff frequency 200 Hz. Plot these two envelope waveforms in one plot, and describe the difference between them.

Assignment (cont.)

- Two *.wav files for lab 5 and the coming lab project 1
  - ‘C_01_01.wav’ & ‘C_01_02.wav’
- For lab 5, use the files above or record speech signal by yourself via Matlab

- **Question ?**