Time-frequency analysis of nonstationary signals

INSELSPITAL
UNIVERSITÄTSSPITAL BERN
HOPITAL UNIVERSITAIRE DE BERNE
BERN UNIVERSITY HOSPITAL

BENESCO Lecture series on signal analysis

Mojtaba Bandarabadi

Center for Experimental Neurology, Inselspital, Hospital University of Bern
mojtaba.bandarabadi@insel.ch

Bern, 6 October 2017

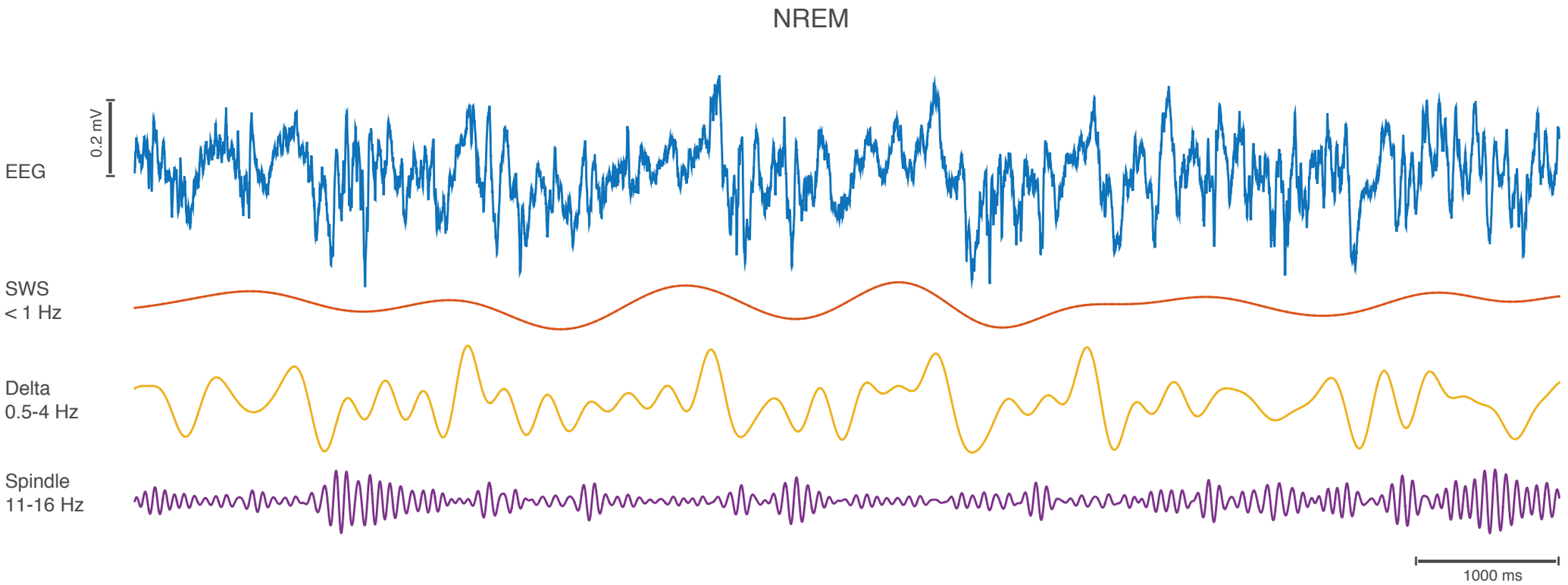
Outline

- Introduction
- Discrete Fourier Transform (DFT)
  - Spectral leakage
  - Windowing
  - Stationarity
- Time-frequency signal analysis
  - Short-time Fourier transform
  - Wavelet analysis
- Spectral analysis of two time series

Introduction

- Signals can be treated in
  - Time domain
    - Biological signals (EEG, ECG, EMG, MEG, …) are function of time
    - Amplitude, variance, autocorrelation, cross-correlation, AR coefficients, zero-crossing, …
  - Frequency domain
    - Mostly distinguished information is hidden in frequency contents
    - Spectral density estimation, Fourier transform, …
  - Time-frequency domain
    - Time-varying signals, such as brain signals
    - Short-time Fourier transform, wavelet analysis

Frequency domain analysis

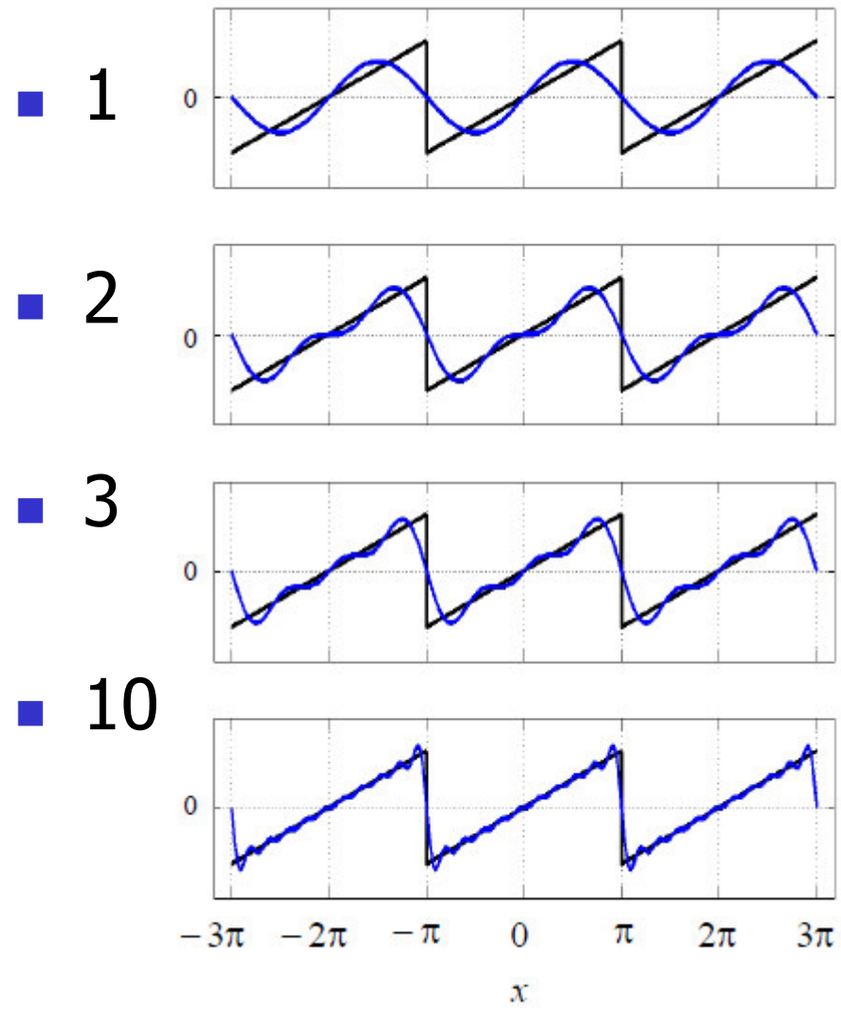


10 s of EEG signal of mouse during non-REM sleep.
Filtered for SWS (Slow Wave Sleep), Delta activity and Spindling

Discrete Fourier Transform (DFT)

- Any waveform can be expressed as a weighted sum of sine/cosine waves!
- DFT provides magnitude and phase at each frequency.

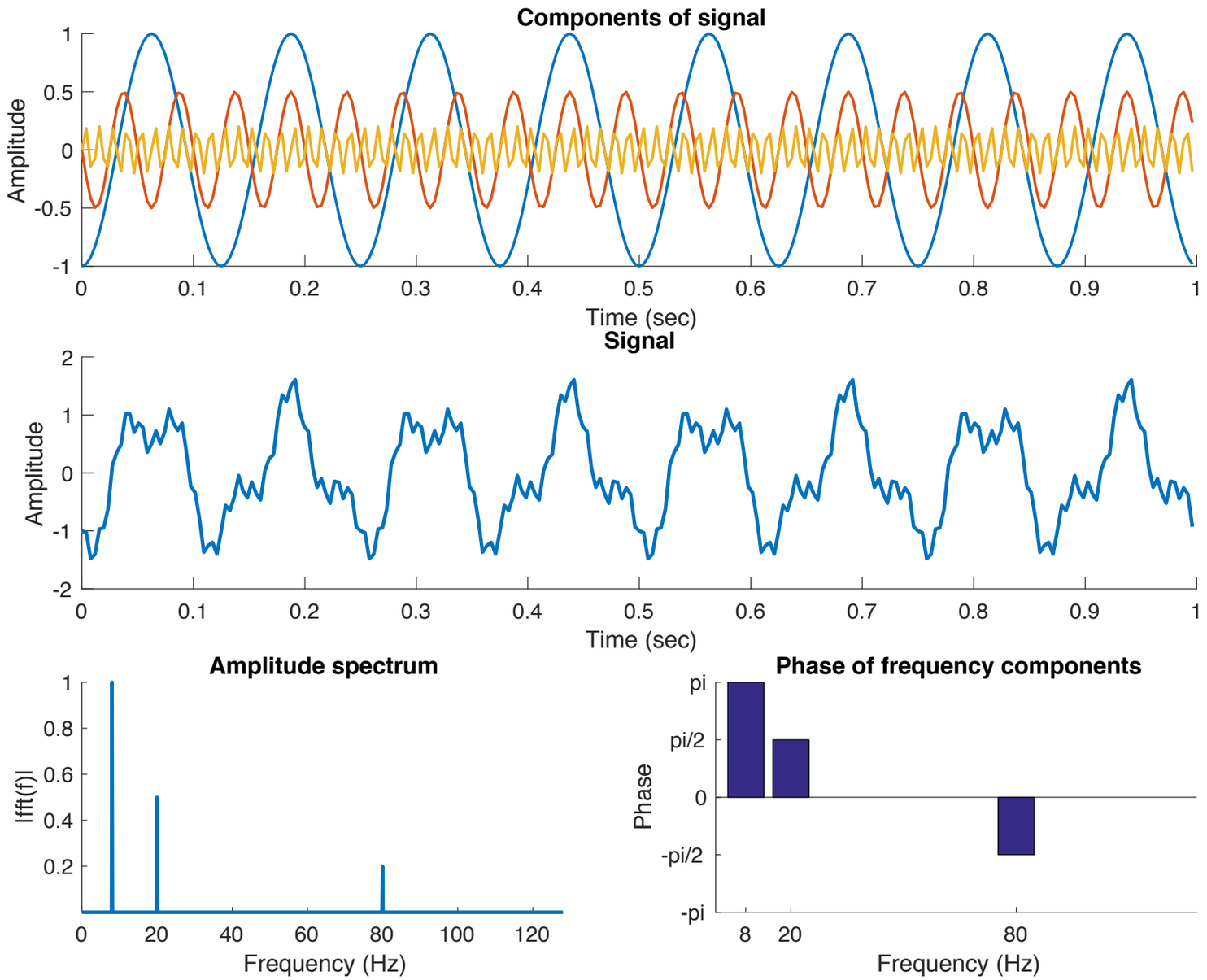
$$X_k \stackrel{\text{def}}{=} \sum_{n=0}^{N-1} x_n \cdot e^{-2\pi i k n / N}, \quad k \in \mathbb{Z}$$



Representing a sawtooth as sum of 10 sine waves.
Source: Wikipedia

Discrete Fourier Transform (DFT)

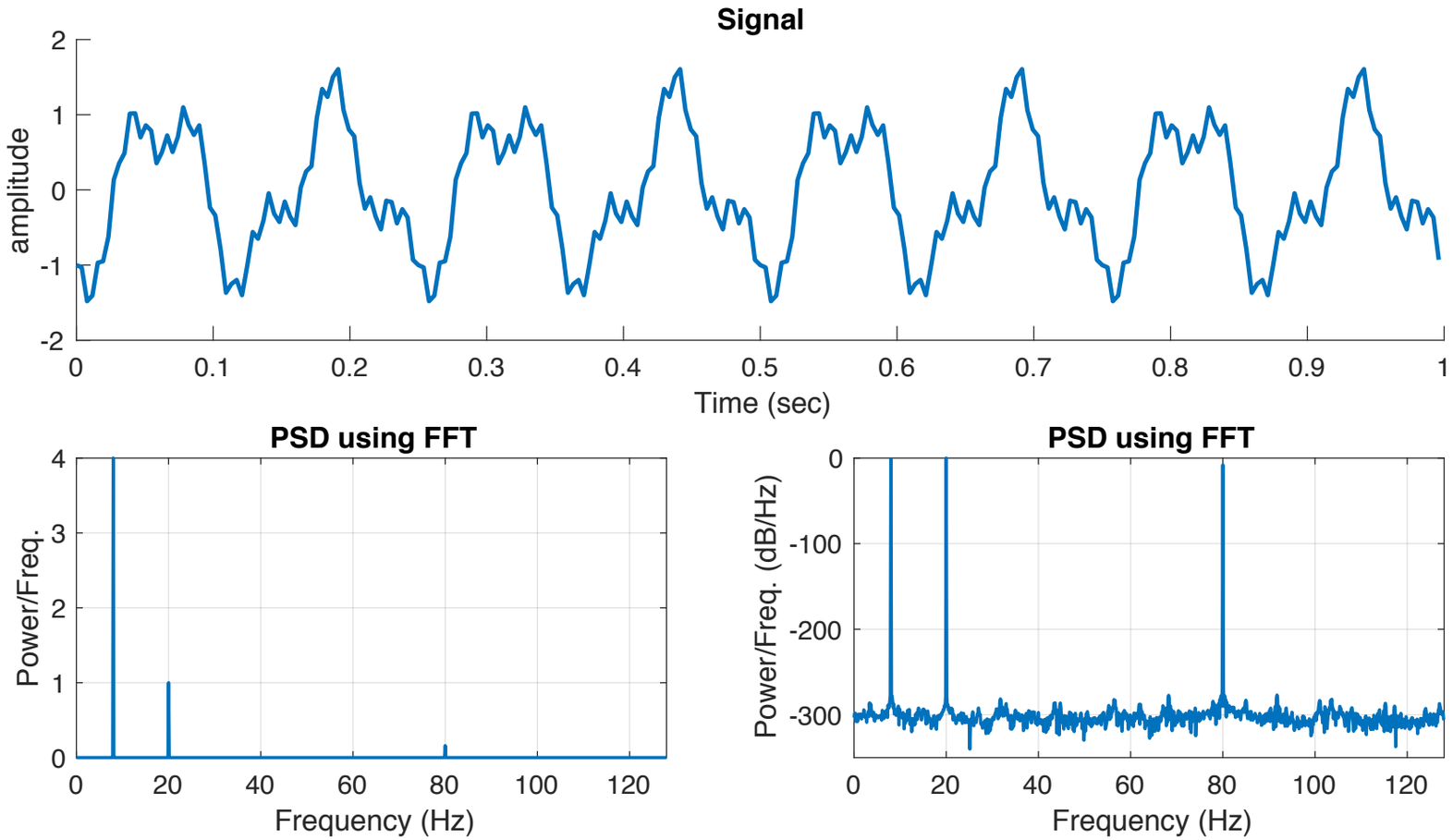
```
% MATLAB

% 8 Hz with pi phase
comp1 = 1.0*cos(2*pi*8*time+pi);
% 20 Hz with +pi/2 phase
comp2 = 0.5*cos(2*pi*20*time+pi/2);
% 80 Hz with -pi/2 phase
comp3 = 0.2*cos(2*pi*80*time-pi/2);
% Merge components
signal = comp1 + comp2 + comp3;

% and now Fast Fourier Transform (FFT)
fftSig = fft(signal);
fftSig = fftSig(1:lenSig/2+1);  % single-sided
freq = 0:sampFreq/lenSig:sampFreq/2;
absFFT = abs(fftSig/lenSig);  % absolute values

absFFT(2:end-1) = 2*absFFT(2:end-1);  %
multiply single-sided spectrum by 2

phaseFFT = angle(fftSig); % phases of freqs.
```



Power spectral density (PSD)

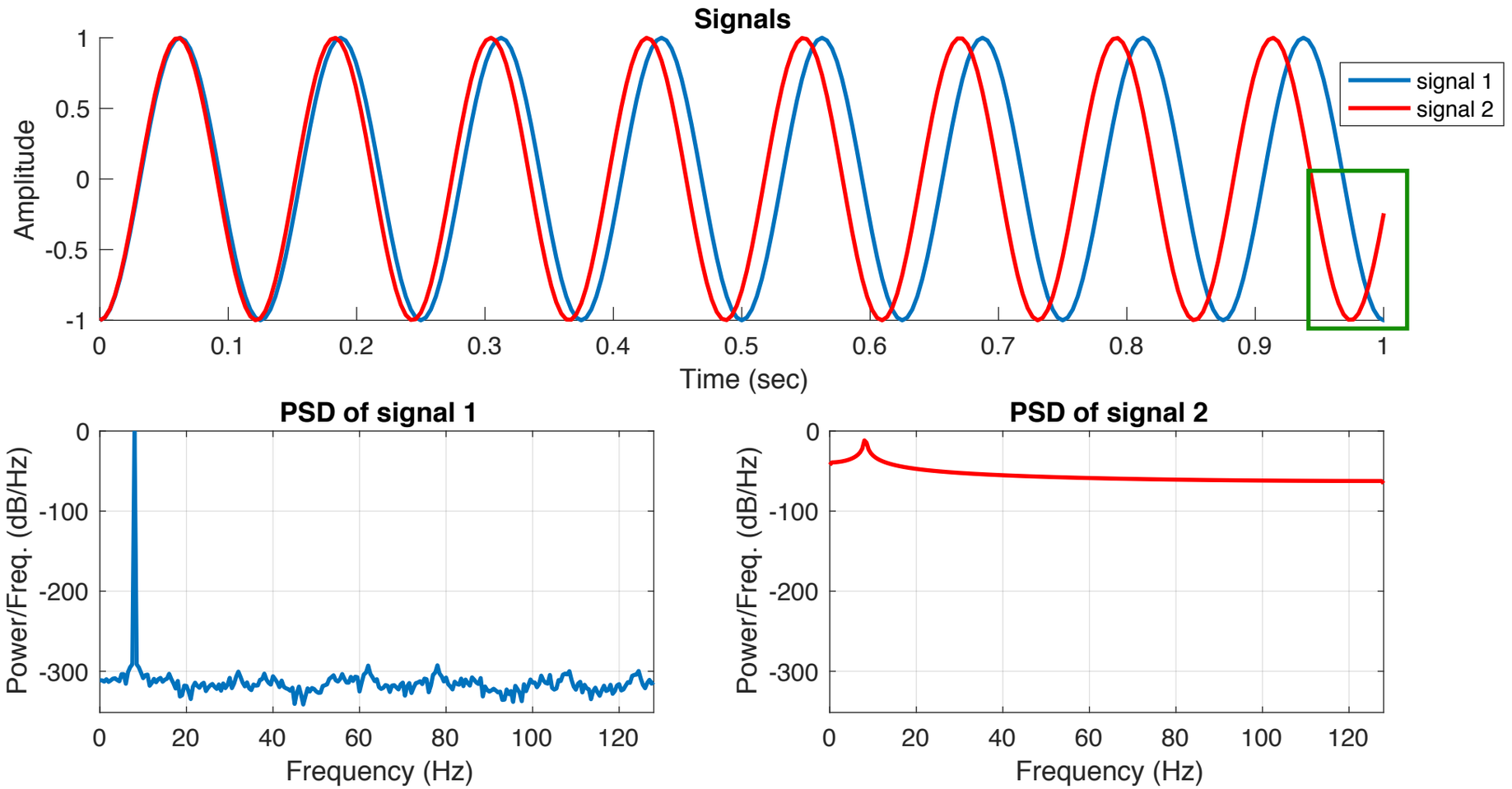
- FFT provides **amplitude** and **phase** of each contributing frequency
- PSD describes how **power** of a signal is distributed over frequencies

```
% MATLAB

% 8 Hz with pi phase
comp1 = 1.0*cos(2*pi*8*time+pi);
% 20 Hz with +pi/2 phase
comp2 = 0.5*cos(2*pi*20*time+pi/2);
% 80 Hz with -pi/2 phase
comp3 = 0.2*cos(2*pi*80*time-pi/2);
% Merge components
signal = comp1 + comp2 + comp3;

% FFT
fftSig = fft(signal);
fftSig = fftSig(1:lenSig/2+1);  % single-sided
freq = 0:sampFreq/lenSig:sampFreq/2;

absFFT = abs(fftSig/lenSig);  % absolute
absFFT(2:end-1) = 2*absFFT(2:end-1);  %
multiply single-sided spectrum by 2

% PSD from FFT
psdFFT = 1/(sampFreq*lenSig) * abs(fftSig).^2;
psdFFT(2:end-1) = 2*psdFFT(2:end-1);
```



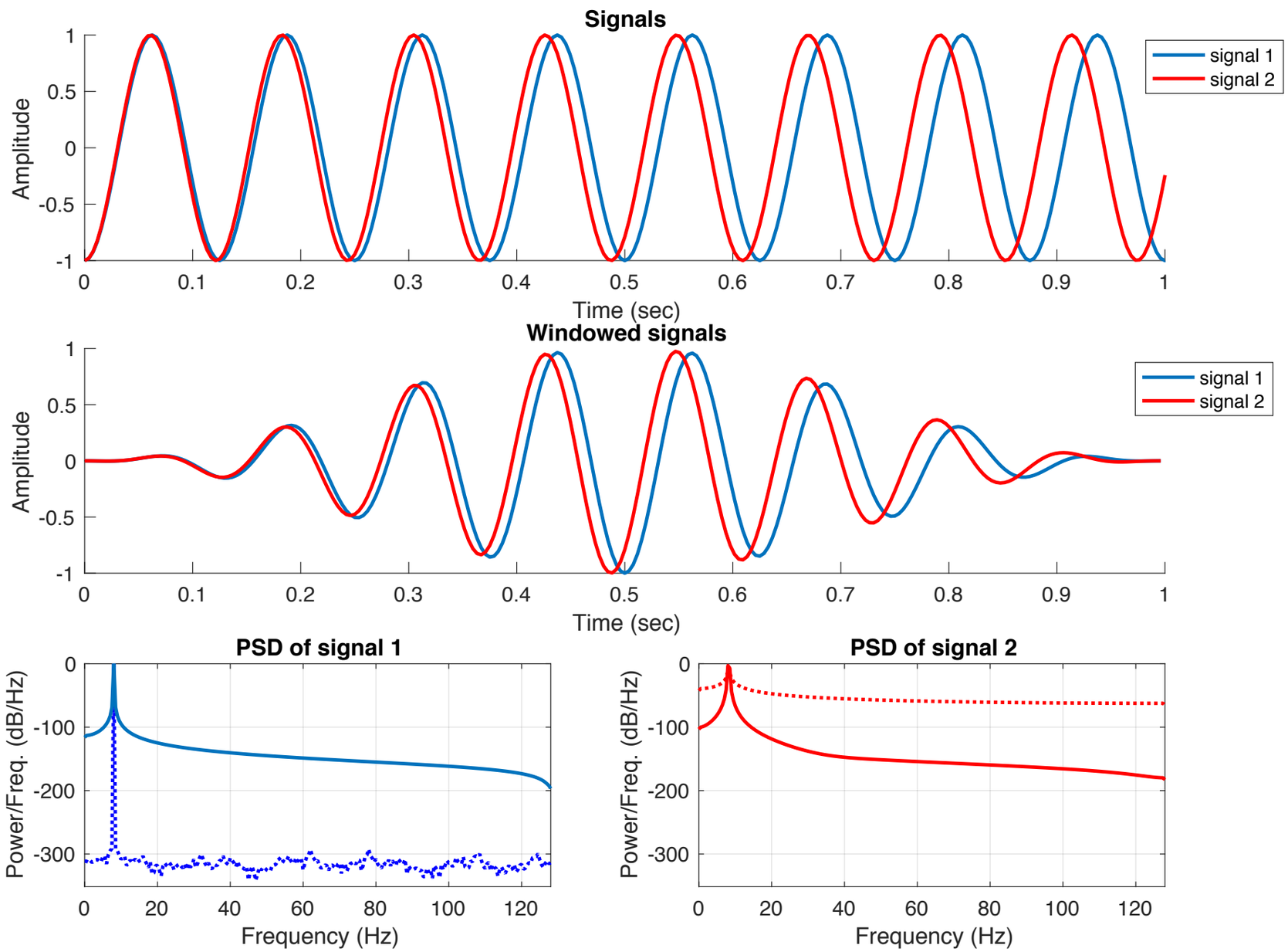
Spectral leakage

- If the number of cycles of a frequency is not an integer (in a segment of data), endpoints are discontinuous.
- These artificial discontinuities show up in the FFT as wide range of frequency components not present in the original signal.



Spectral leakage and windowing

- This is unavoidable for DFT, but can be improved using windowing.

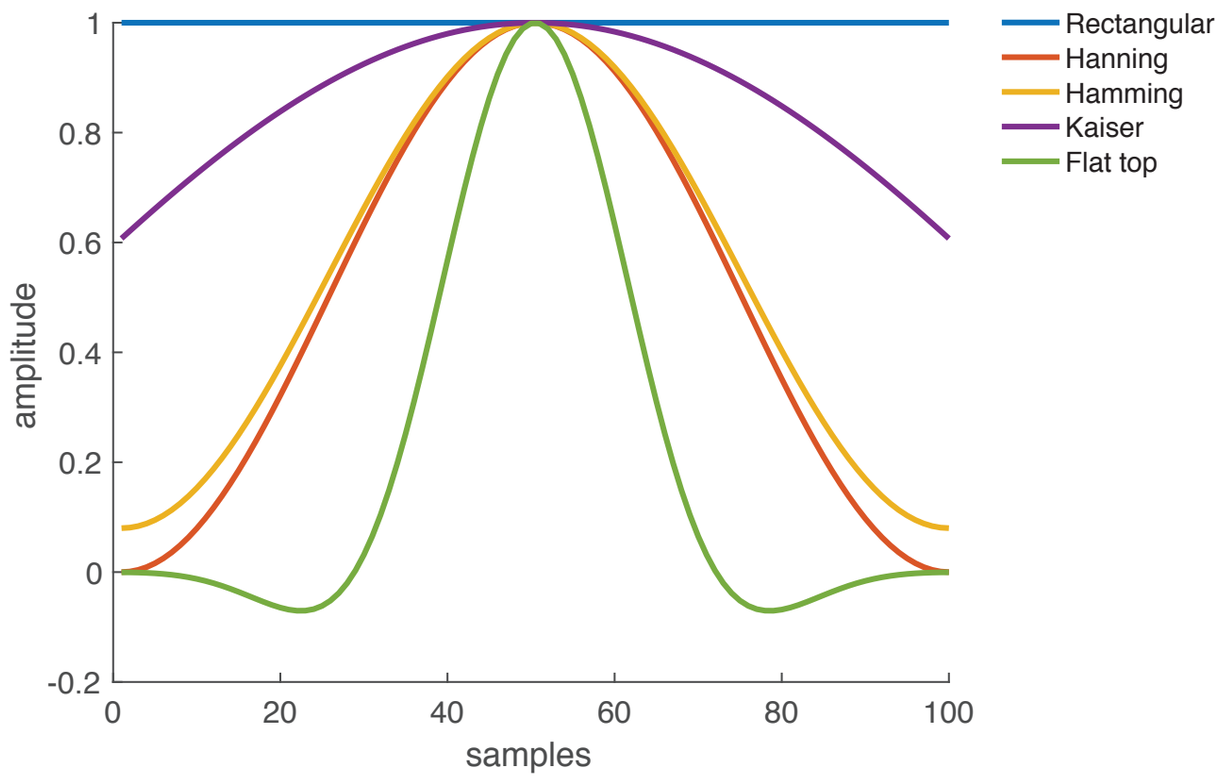


Windowing functions

- There are several windowing functions for different situations/applications
  - Hanning, Hamming, Kaiser, … (see http://ch.mathworks.com/help/signal/windows.html)
  - In general, the Hanning window is satisfactory in 95 percent of cases.

Hanning window: $w_j = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi j}{N-1}\right)\right] \quad j \in [0 \ldots N-1]$

```
% MATLAB

winLen = 100; % length of window in sample

rectWin = window(@rectwin,winLen); % Rectangular

hannWin = window(@hann,winLen); % Hanning

hammWin = window(@hamming,winLen); % Hamming

kaiserWin = window(@kaiser,winLen,1.5); % Kaiser

flattopWin = window(@flattopwin,winLen); % Flat top
```



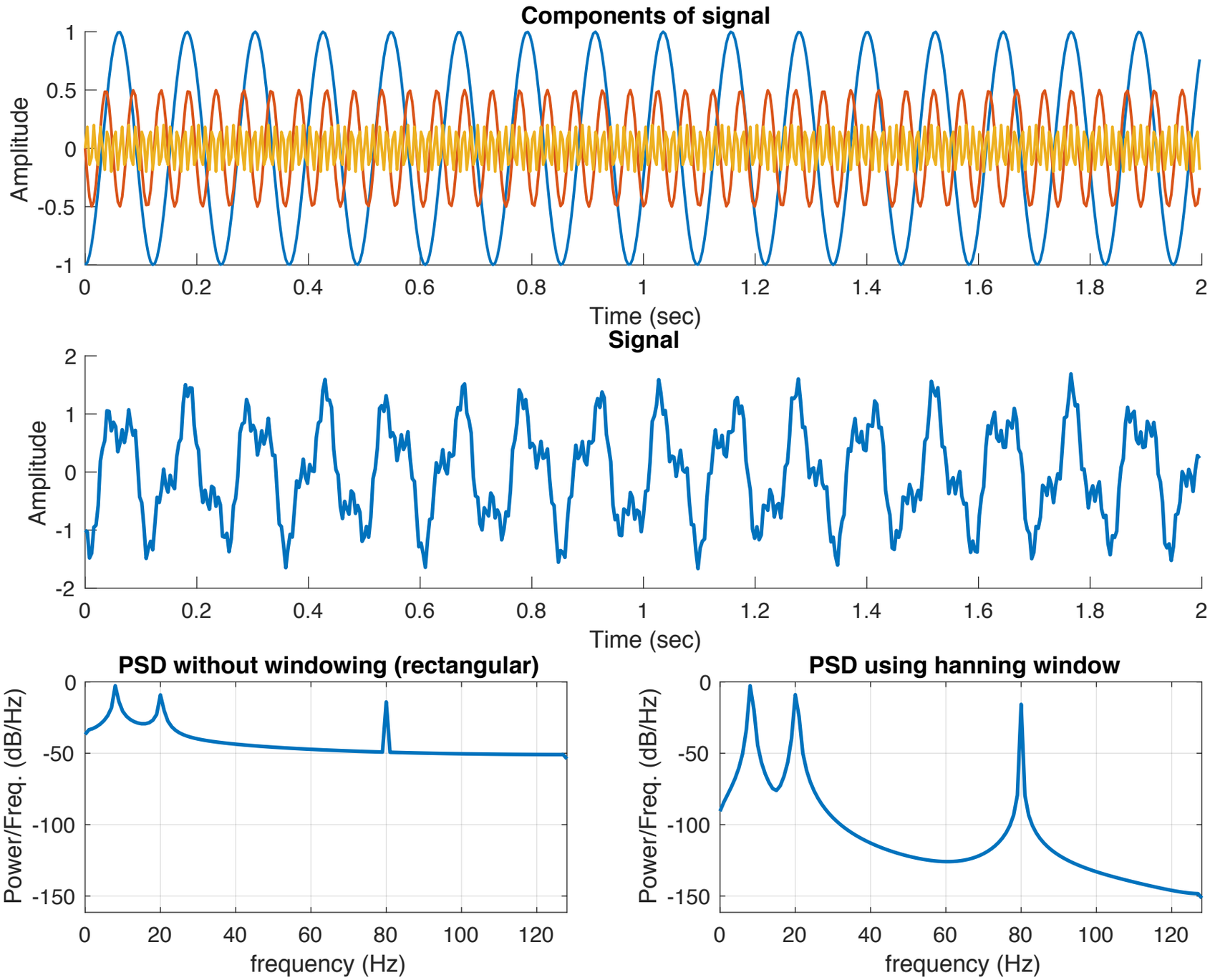
Some of famous windowing functions of length 100 samples

Choice of window

Type of Signal	Window function
Transients whose duration is shorter than the length of the window	Rectangular
Transients whose duration is longer than the length of the window	Exponential, Hanning
General-purpose applications	Hanning
Spectral analysis (frequency-response measurements)	Hanning (for random excitation), Rectangular (for pseudorandom excitation)
Separation of two tones with frequencies very close to each other but with widely differing amplitudes	Kaiser-Bessel
Separation of two tones with frequencies very close to each other but with almost equal amplitudes	Rectangular
Accurate single-tone amplitude measurements	Flat top
Sine wave or combination of sine waves	Hanning
Sine wave and amplitude accuracy is important	Flat top
Narrowband random signal (vibration data)	Hanning
Broadband random (white noise)	Rectangular
Closely spaced sine waves	Rectangular, Hamming
Excitation signals (hammer blow)	Force
Response signals	Exponential
Unknown content	Hanning

http://zone.ni.com/reference/en-XX/help/372614J-01/lvanlsconcepts/choosing_smoothing_window/

Spectral leakage

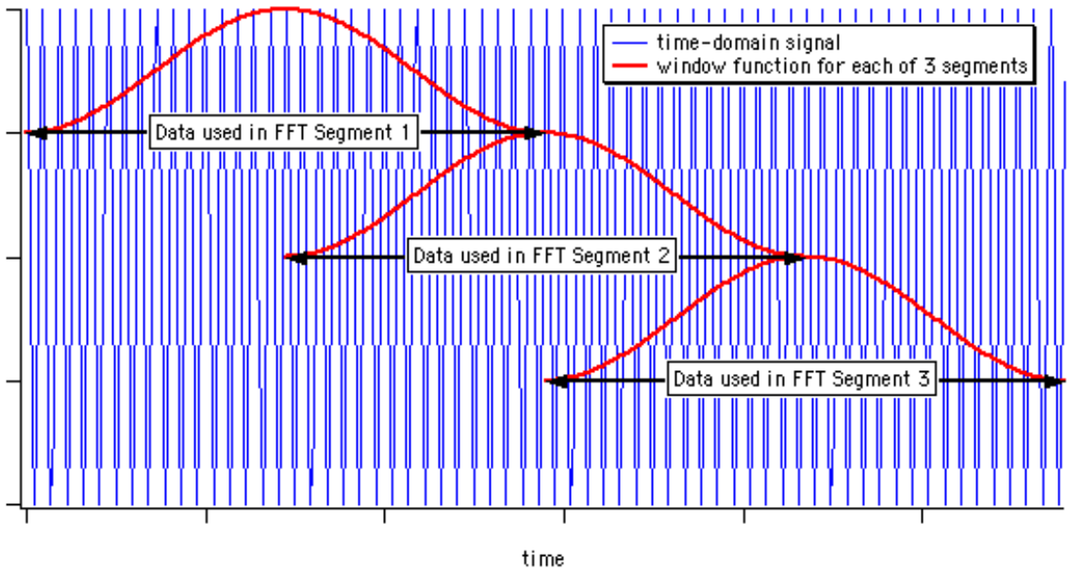


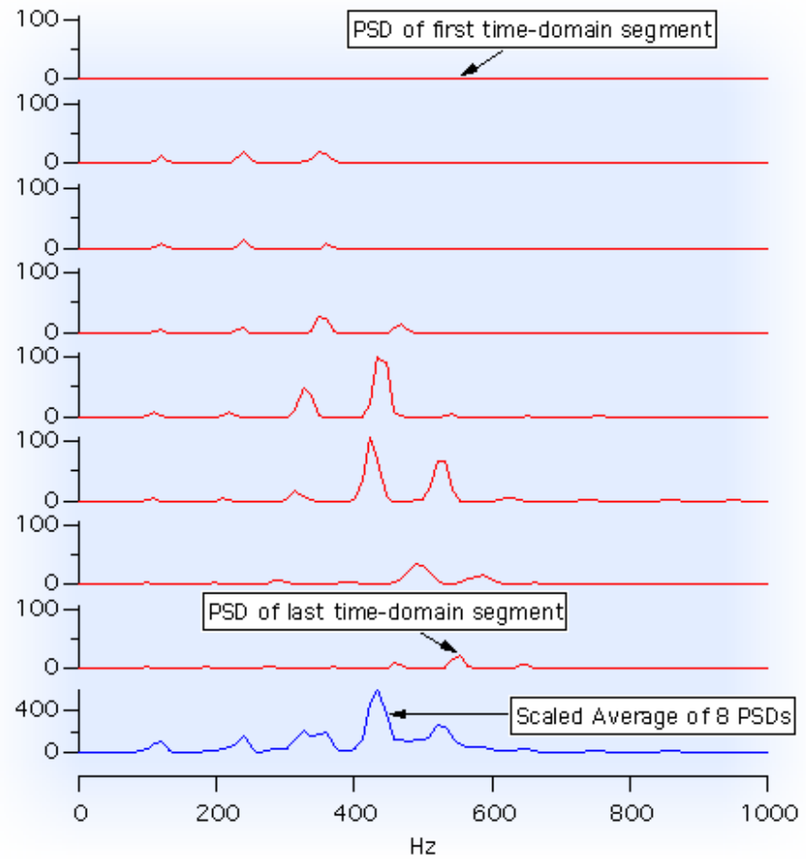
Stationarity

- A stationary time series has constant statistical properties over time, such as mean, variance, and autocorrelation.
  - For stationary signals, frequency domain analysis works well
- Brain is a dynamic complex system
  - Neural signals are non-stationary
    - Frequency content of neural signals changes with time
- How to deal with time-varying signals?
  - Averaging
  - Time-frequency representations (TFRs)

PSD of non-stationary and long signals

- **Welch's method** is an improved PSD estimator, reduces noise by averaging
  - Split up signal into overlapping segments
  - A window function (such as hamming) is applied on segments
  - Squared magnitude of DFT is calculated for each segment
  - Individual PSDs are averaged



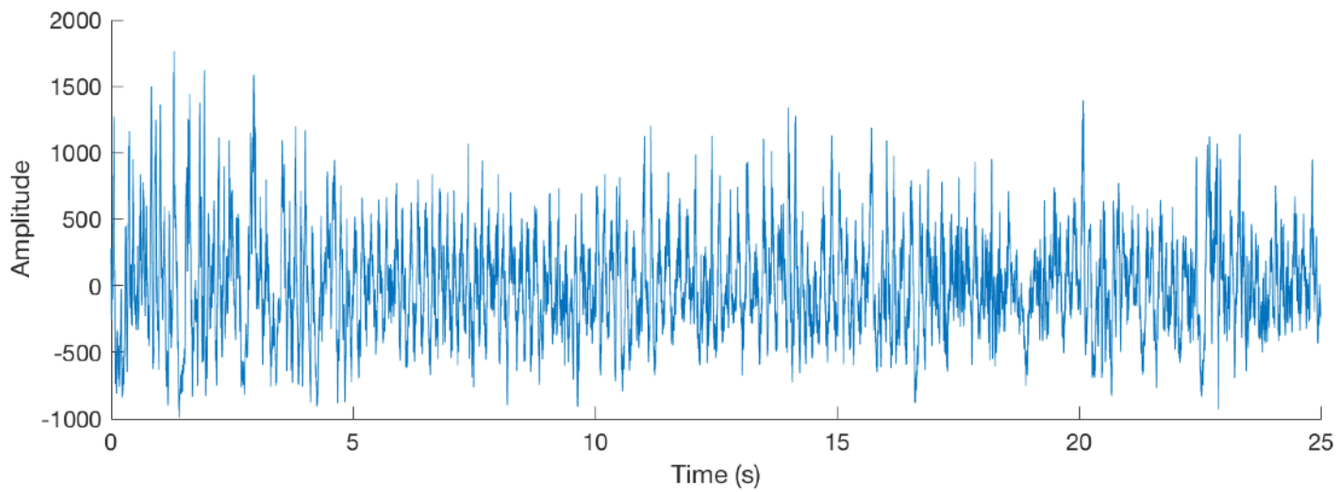


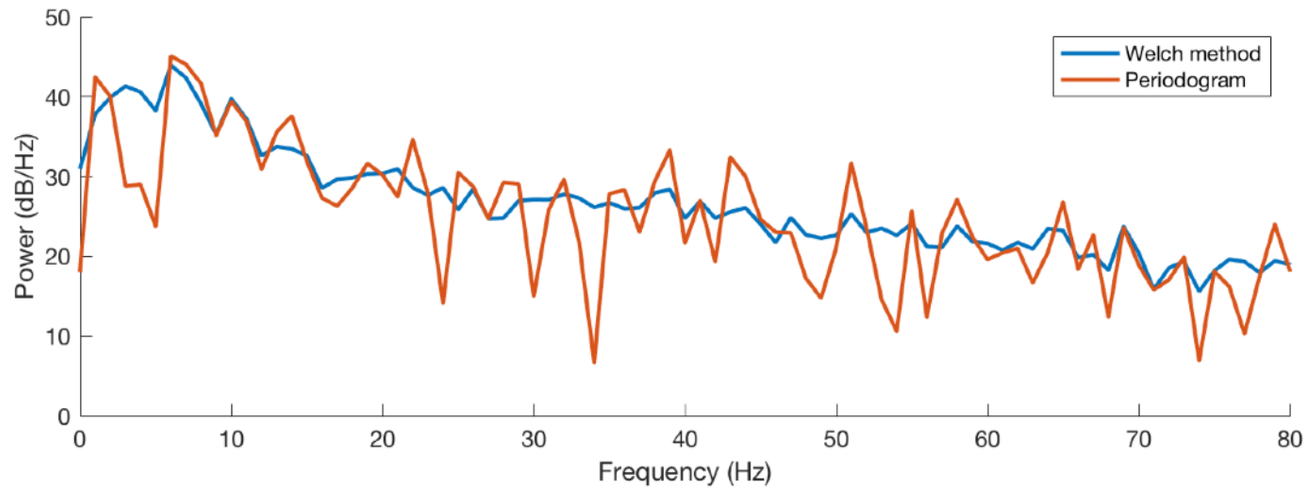
https://www.wavemetrics.com/products/igorpro/dataanalysis/signalprocessing/powerspectra.htm

PSD using Welch's method

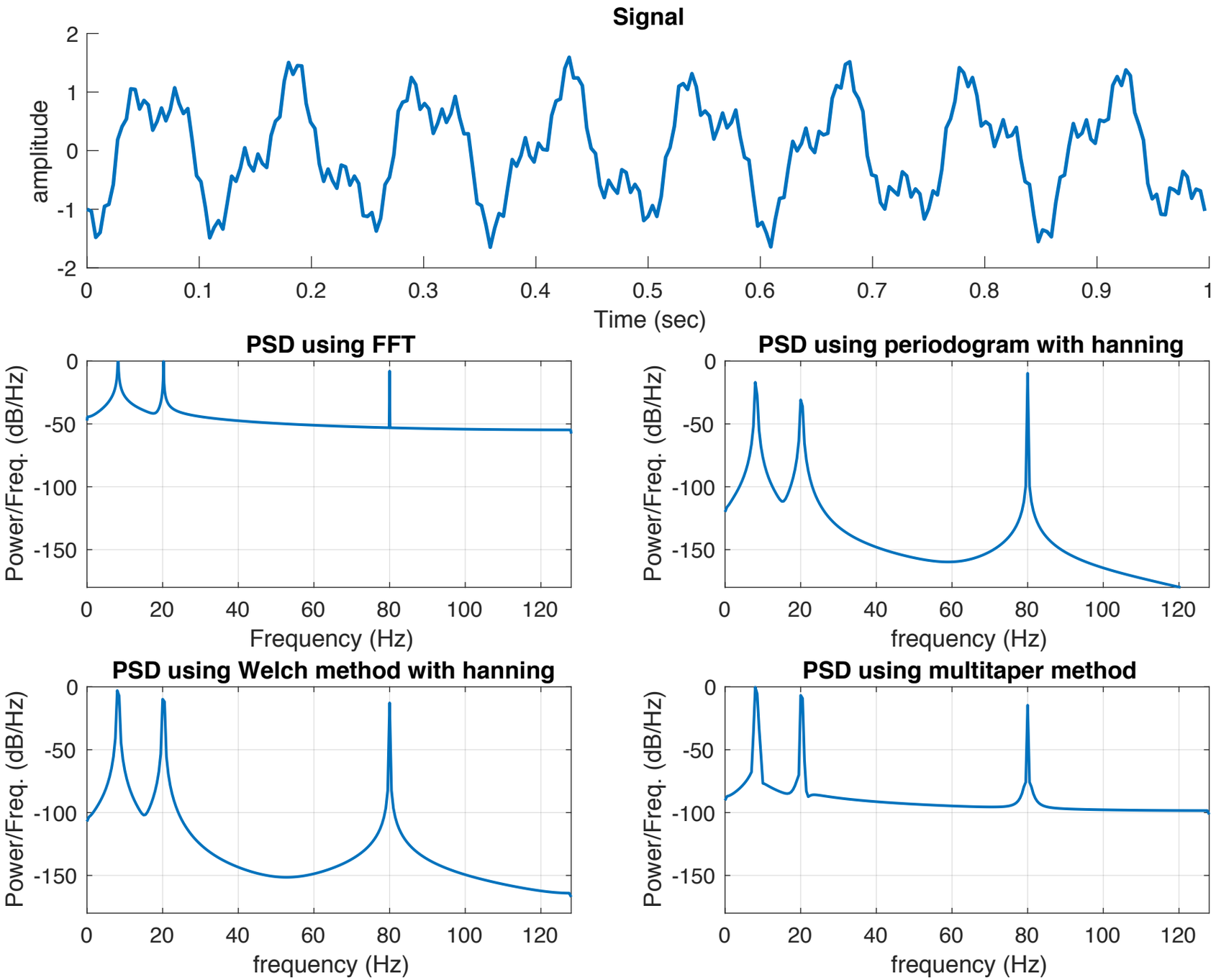
- "pwelch.m" function [pxx,f] = pwelch(x,window,noverlap,f,fs)
- Compare welch's method with periodogram (no windowing)

```
% MATLAB code
% rawSig and fs are provided in lecture notes
% rawSig is hippocampal LFP of mouse (fs = 1000 Hz)
t = 0:1/fs:(length(rawSig)-1)/fs;
subplot(211); plot(t,rawSig); box off
xlabel('Time (s)')
ylabel('Amplitude')

window = 4*fs; % 4-sec windows (resolution 0.25Hz)
overlap = round(0.5*window); % 50% overlap
[pxx,f] = pwelch(rawSig,window,overlap,fs,fs);
subplot(212); hold on
plot(f,10*log10(pxx),'linewidth',2); box off
xlim([0 80])
xlabel('Frequency (Hz)')
ylabel('Power (dB/Hz)')

[pxx1,f1] = periodogram(rawSig,...
    rectwin(length(rawSig)),fs,fs);
plot(f1,10*log10(pxx1),'linewidth',2); hold off
xlim([0 80])
xlabel('Frequency (Hz)')
ylabel('Power (dB/Hz)')
legend({'Welch method','Periodogram'})
```





PSD - comparing some approaches



What to do with the PSD?

- Finding possible periodicities in a signal
- Quantifying power of frequency bands in different conditions (health vs disease)
  - e.g. delta, theta, alpha, beta, gamma, …, for brain signals
- Spectral edge frequency and spectral edge power
  - Spectral edge frequency stands for the frequency below which x percent of the overall power of the signal is located.
  - Power covered under spectral edge frequency is called spectral edge power
- Looking for interferences
- …

Conclusion - frequency domain analysis

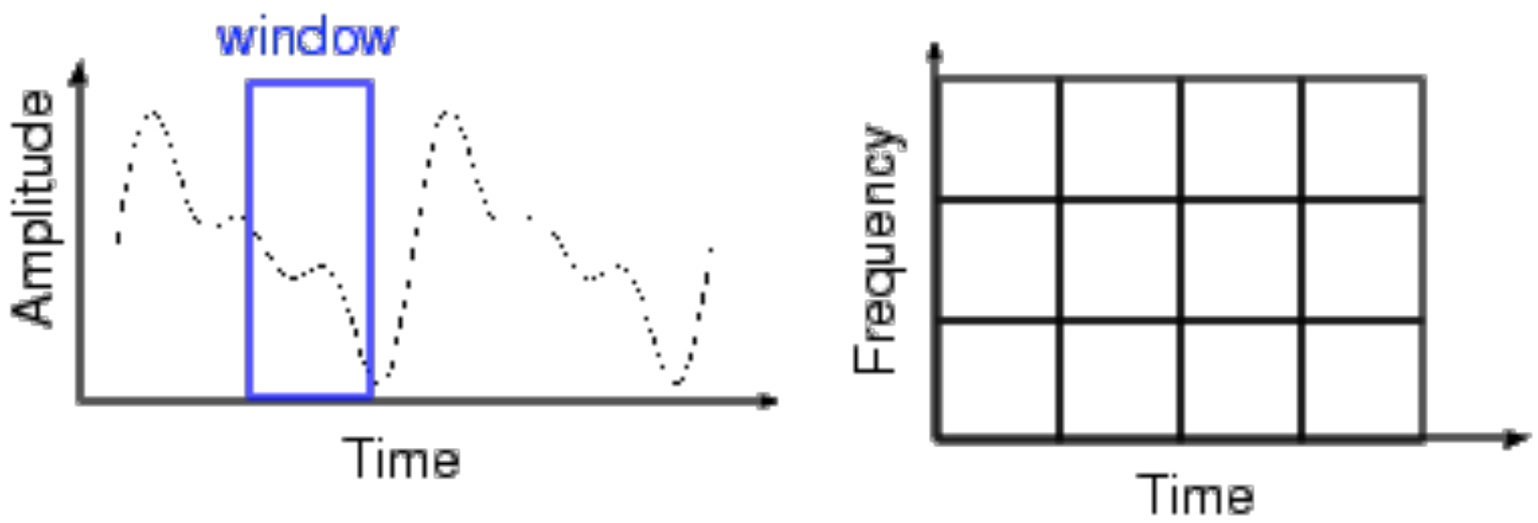
- Useful to have a global view of frequency components of a signal
- Suitable for stationary signals
  - The frequency content doesn’t change with time (time-invariant)
- Loosing time information
  - When do transient changes occur?
  - What frequencies are exist at a particular time?

Time-frequency representation (TFR)

- TFR provides information in time-frequency plane
- Some applications in neural signal processing
  - Analysis of event-related potential (ERP)
  - Neurostimulation response analysis (Electrical / Optogenetics, TMS, …)
  - Studying of epileptic seizures
  - Recalling a specific memory
- Some approaches
  - **Short-time Fourier transform (spectrogram)**
  - **Wavelet transform (scalogram), wavelet coherence**
  - Wigner distribution
  - Choi-Williams distribution
  - Matching pursuit

Short-Time Fourier Transform (STFT)

- Signals can be assumed quasi-stationary in short times
  - Short time definition depends on frequency content of signal
  - Window length should cover few cycles of main frequency component

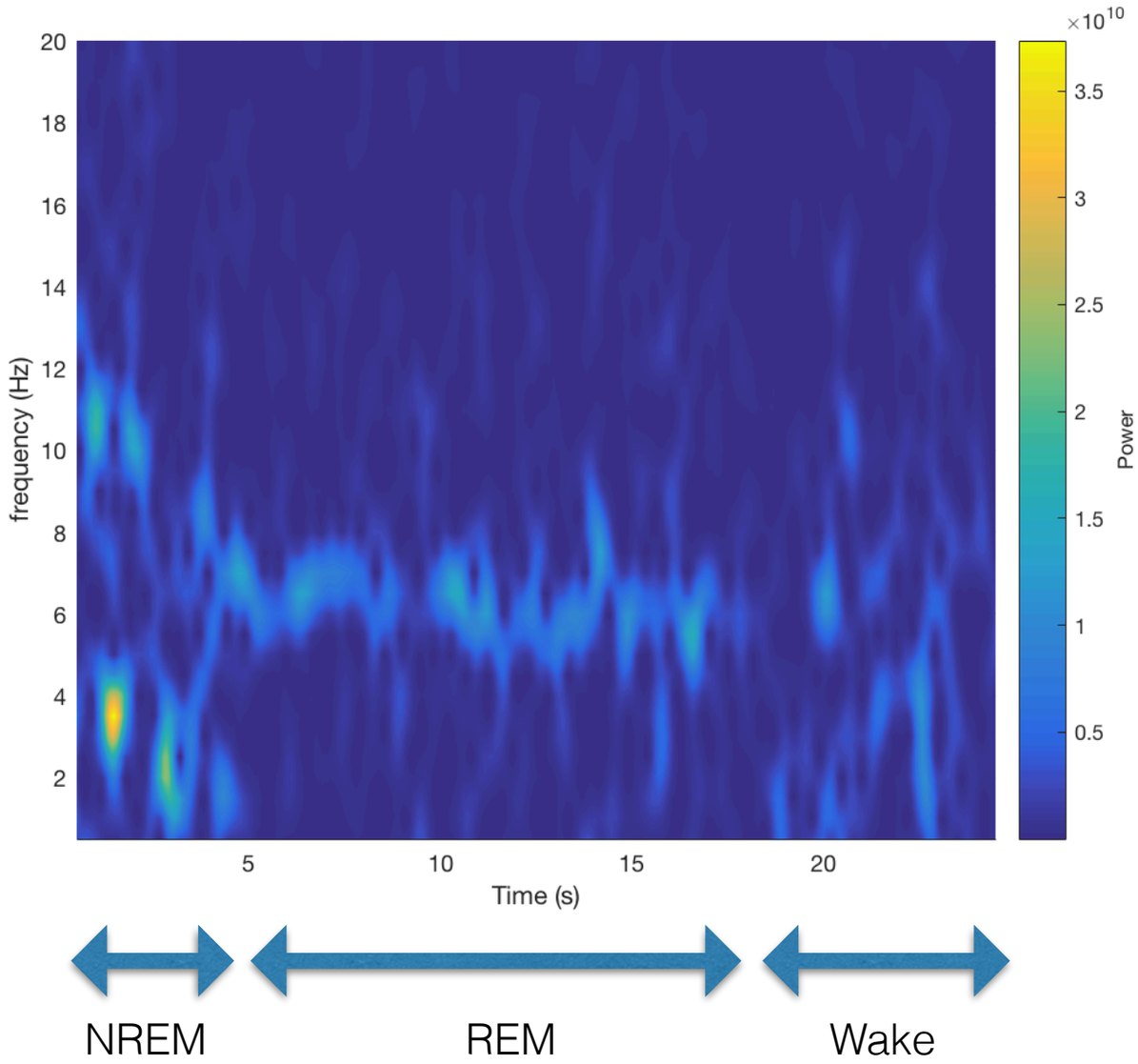


STFT - MATLAB

- “spectrogram.m” function

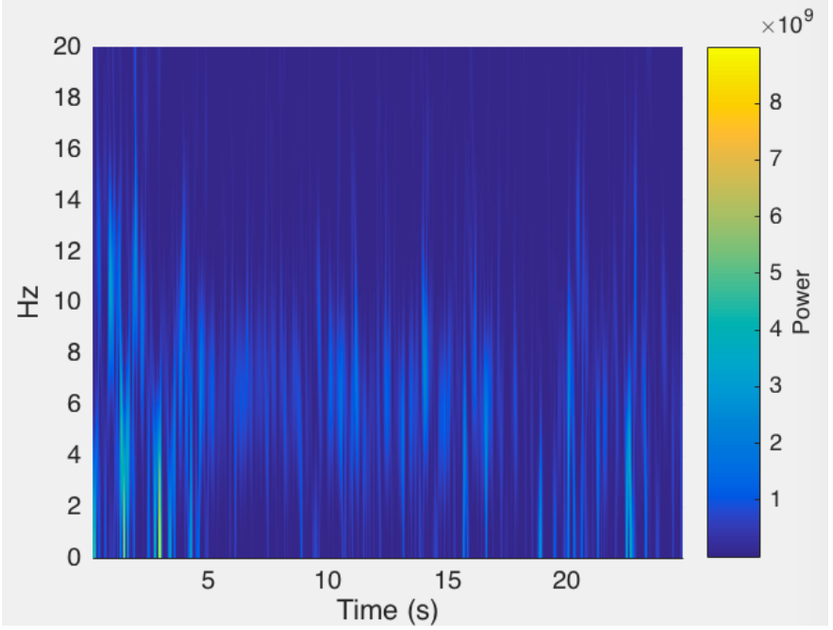
[s,f,t] = spectrogram(x,window,noverlap,f,fs)

```
% MATLAB code
% rawSig and fs are provided in lecture notes
% rawSig is hippocampal LFP from mouse (fs = 1000 Hz)

window = 1*fs; % window size in sample
overlap = round(0.96*window); % overlap is an integer value

[SC,freq,time] = spectrogram(rawSig,window,overlap,2*fs,fs);
SC = abs(SC(freq>=minfreq&freq<=maxfreq,:));
freq = freq(freq>=minfreq&freq<=maxfreq);
args = {time,freq,SC.^2};

figure('Color',[1 1 1]);
surf(args{:},'edgecolor','none');
view(0,90);
axis tight;
shading interp;
colormap(parula(128));
h = colorbar;
h.Label.String = 'Power';
xlabel('Time (s)');
ylabel('frequency (Hz)');
```

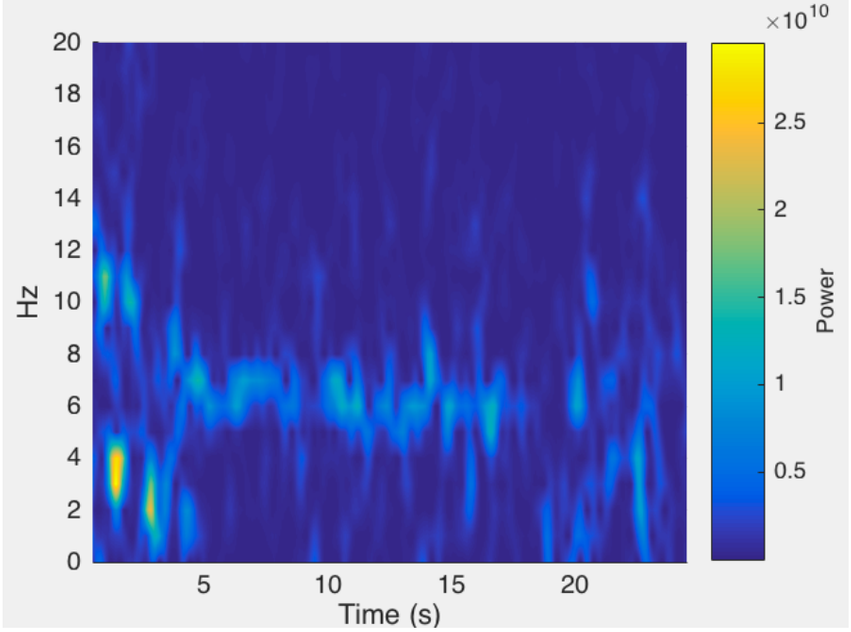


STFT - effect of window width

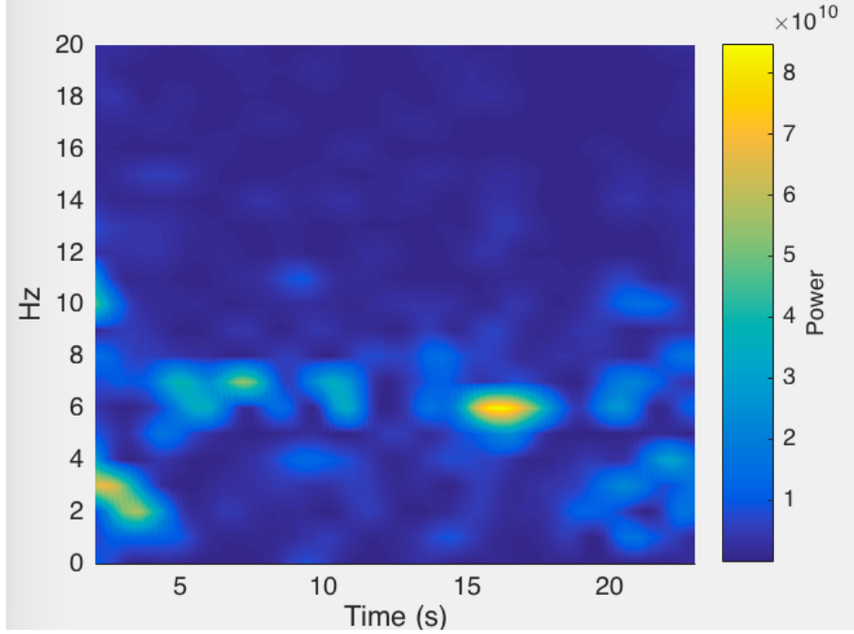
- Control the time and frequency resolutions
  - Narrow window provides better time resolution, degrade freq. resolution
  - Wide window provides better freq. resolution, degrade time resolution



Window length = 0.25 sec



Window length = 1 sec



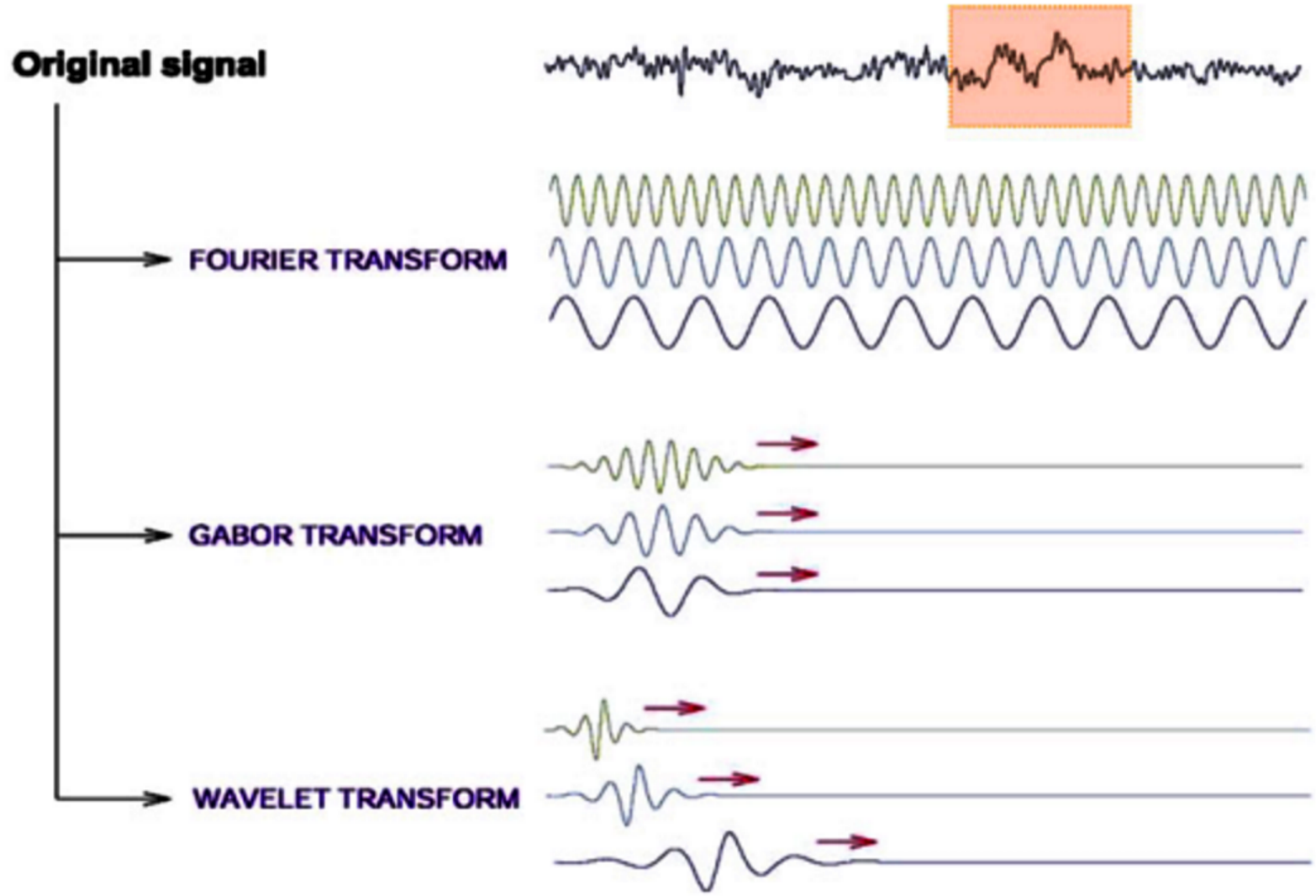
Window length = 4 sec

Conclusion - STFT

- STFT representation is useful for narrowband analysis
  - Lower and higher frequency components are close
- Problem in wideband signals
  - Higher frequency oscillations are usually shorter than lower ones.
  - A fixed window for all frequencies (as in STFT) would not work well.
- Solution
  - Multi-resolution time-frequency representation using wavelets

Wavelet transform

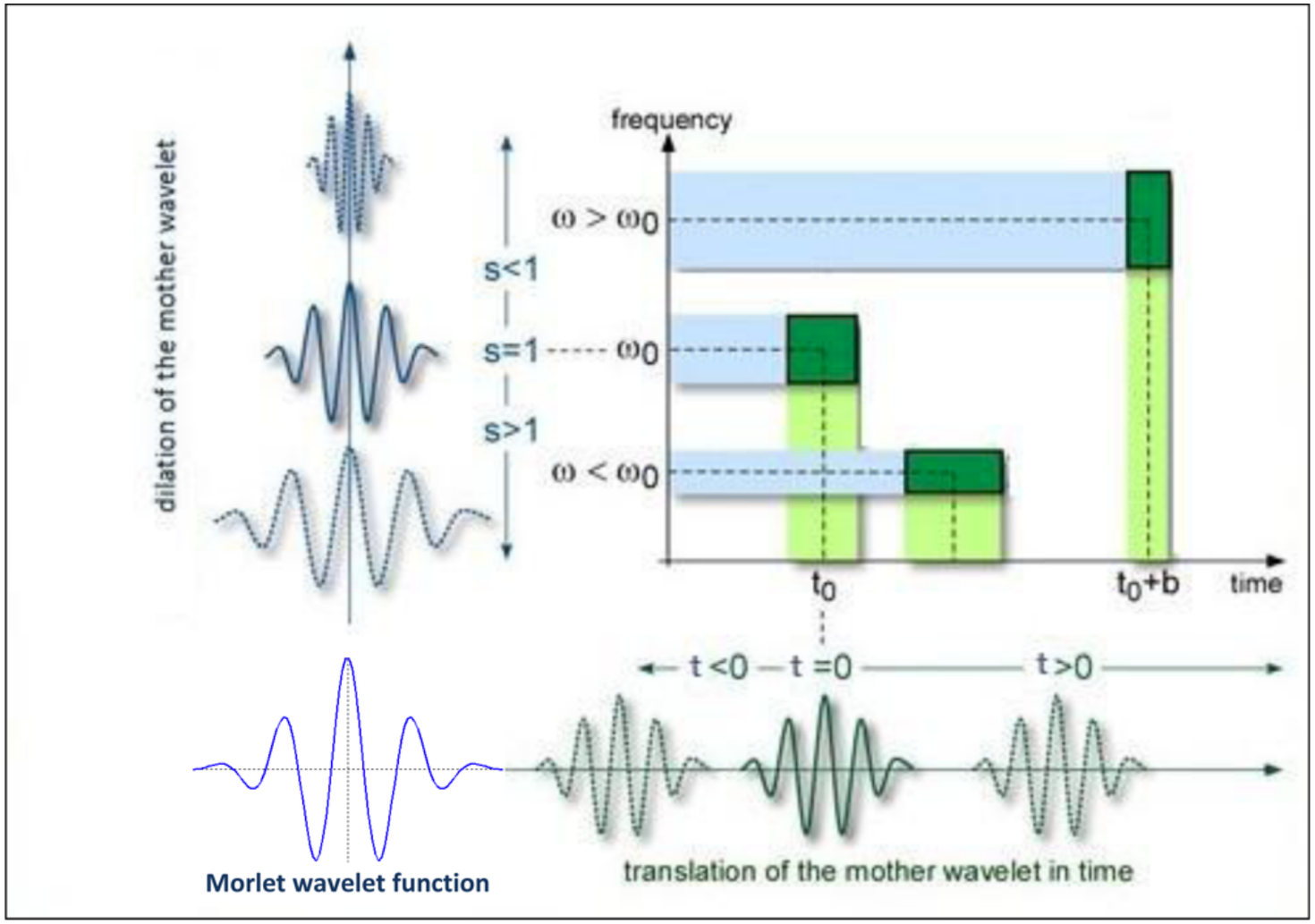
- Convolution between signal and scaled versions of a wavelet function.



Freeman W., Quiroga R. Q., Imaging Brain Function With EEG, Springer, 2013

Wavelet transform

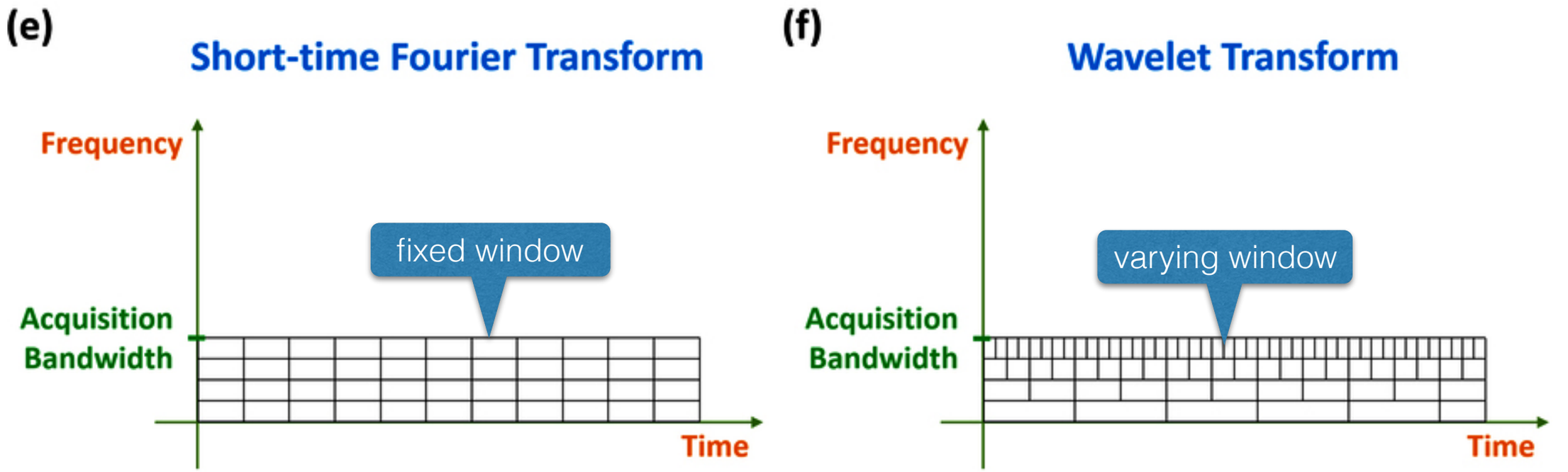
- Multi-resolution nature of wavelet transform
  - Shorter windows of signal are considered for higher frequencies



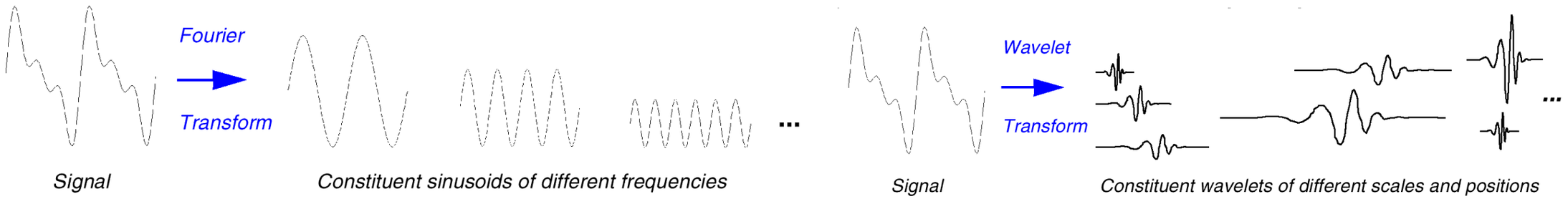
Erol S., Time-Frequency Analyses of Tide-Gauge Sensor Data, Sensors, 2011

Wavelet transform vs. STFT

- STFT uses fixed window size for all frequencies
- Wavelet transform has multi-resolution nature



Mahjoubfar et al., Design of Warped Stretch Transform, Nat. Sci. Rep., 2015



Wavelet transform using MATLAB

- There are many wavelet functions in the wavelet toolbox of MATLAB.
- “waveinfo” lists supported wavelets
- Some functions
  - Continuous wavelet transform (CWT)

    coefs = cwt(x,scales,’wname’)
  - CWT using FFT algorithm (faster than cwt function)

    cwtstruct = cwtft(sig,Name,Value)
  - Discrete wavelet transform

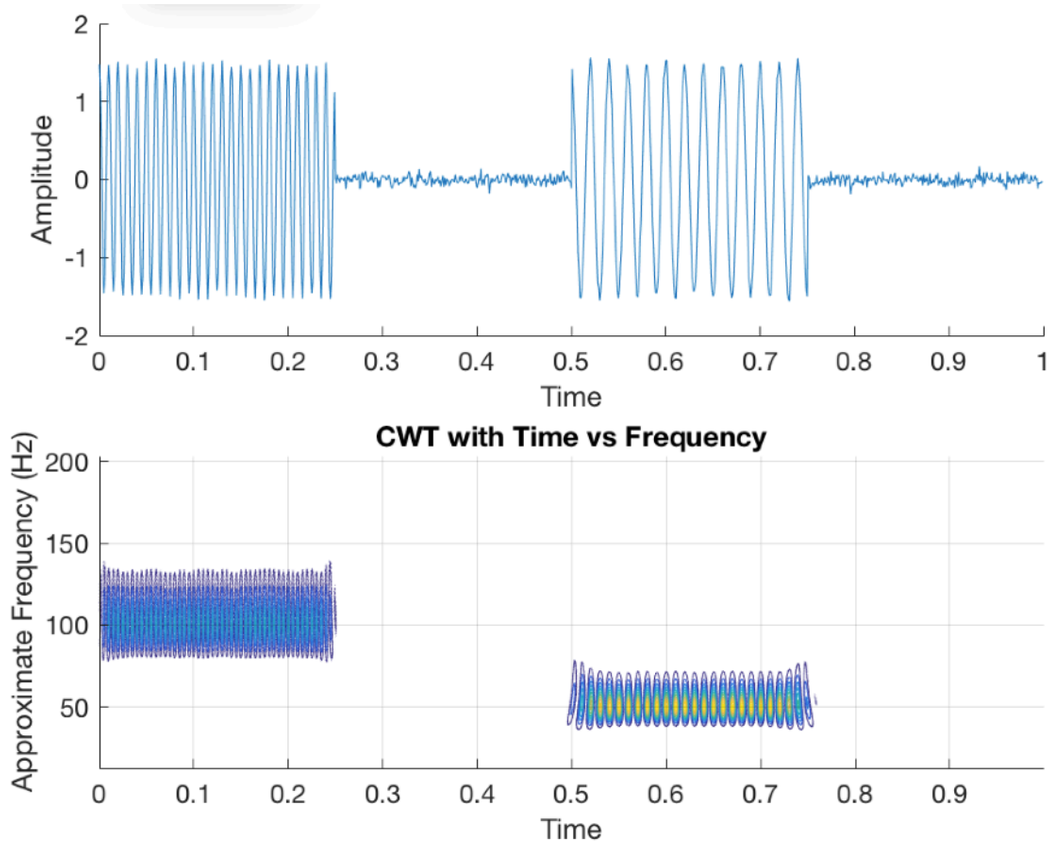
    [C,L] = wavedec(X,N,’wname’)

Scale to frequency

- Each scale of wavelet function has a pseudo-frequency
  - An approximate relationship between scale and frequency)

F = scal2frq(A,'wname',DELTA)

```
% MATLAB code
Fs = 1000; % sampling frequency of signal
t = 0:1/Fs:1-1/Fs; % time axis
x = 1.5*cos(2*pi*100*t).*(t<0.25)+1.5*cos(2*pi*50*t).*(t>0.5 & t<=0.75);
x = x+0.05*randn(size(t)); % add noise to signal
figure('Color',[1 1 1]);
subplot(211); plot(t, x)
xlabel('Time');
ylabel('Amplitude');
set(gca,'fontsize',12); box off

 numvoices = 32;
a0 = 2^(1/numvoices);
scales = a0.^(2*numvoices:1/numvoices:6*numvoices); % scales
cfs = cwt(x,scales,'morl'); % apply cwt
pfreq = scal2frq(scales,'morl',1/Fs); % match scale to frequency
subplot(212);
contour(t,pfreq,abs(cfs).^2);
axis tight;
grid on;
xlabel('Time');
ylabel('Approximate Frequency (Hz)');
title('CWT with Time vs Frequency');
set(gca,'fontsize',12); box off
```

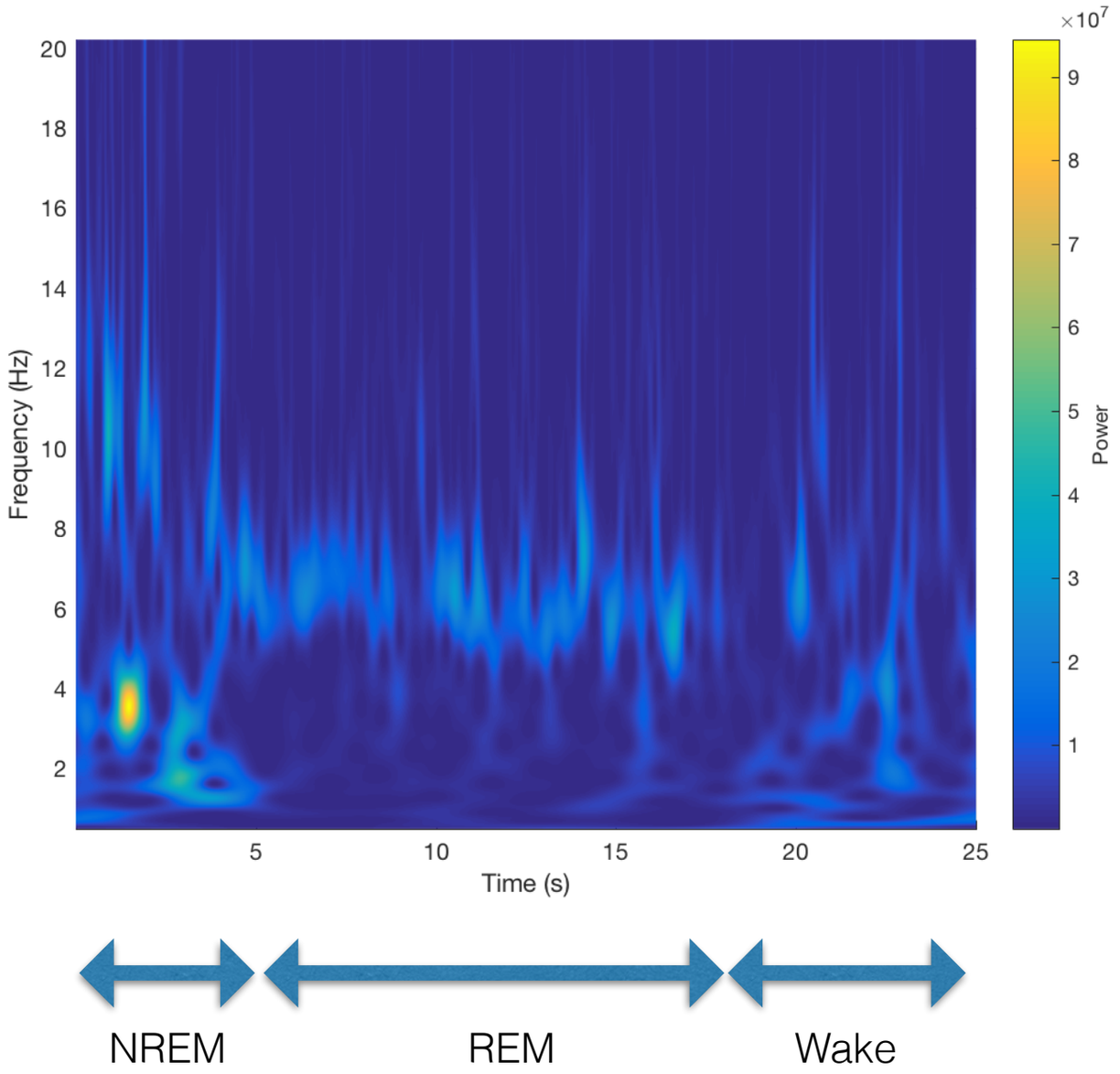


MATLAB help files

Wavelet transform - application

- Scalogram of hippocampal LFP (NREM, REM, Wake) using Morlet wavelet

```
% MATLAB code
% rawSig and fs are provided in lecture notes)
% rawSig is hippocampal LFP from mouse (fs = 1000 Hz)

% define min/max frequencies (Hz) for analysis
minfreq = 0.5; % min frequency
maxfreq = 20;  % max frequency

% finding corresponding scales for min/max frequencies
dt = 1/fs;   % time between two samples (1/sampling freq.)
f0 = 6/(2*pi); % central freq. of Morlet function ('morl')
NumVoices = 32;
a0 = 2^(1/NumVoices);
minscale = floor(NumVoices*log2(f0/(maxfreq*dt)));
maxscale = ceil(NumVoices*log2(f0/(minfreq*dt)));
scales = a0.^(minscale:maxscale).*dt;

% apply wavelet transform
cwtx = cwtft({rawSig,dt},'wavelet','morl','scales',scales);
```



Frequency analysis of two signals

- Cross power spectral density (CPSD) Provides information on
  - the power shared by a given frequency for the two signals
  - the phase shift information between the two signals at a frequency.
- Is obtained using PSD (Welch's method) of cross-correlation of two signals

$$P_{xy}(\omega) = \sum_{m=-\infty}^{\infty} R_{xy}(m) e^{-j\omega m}.$$

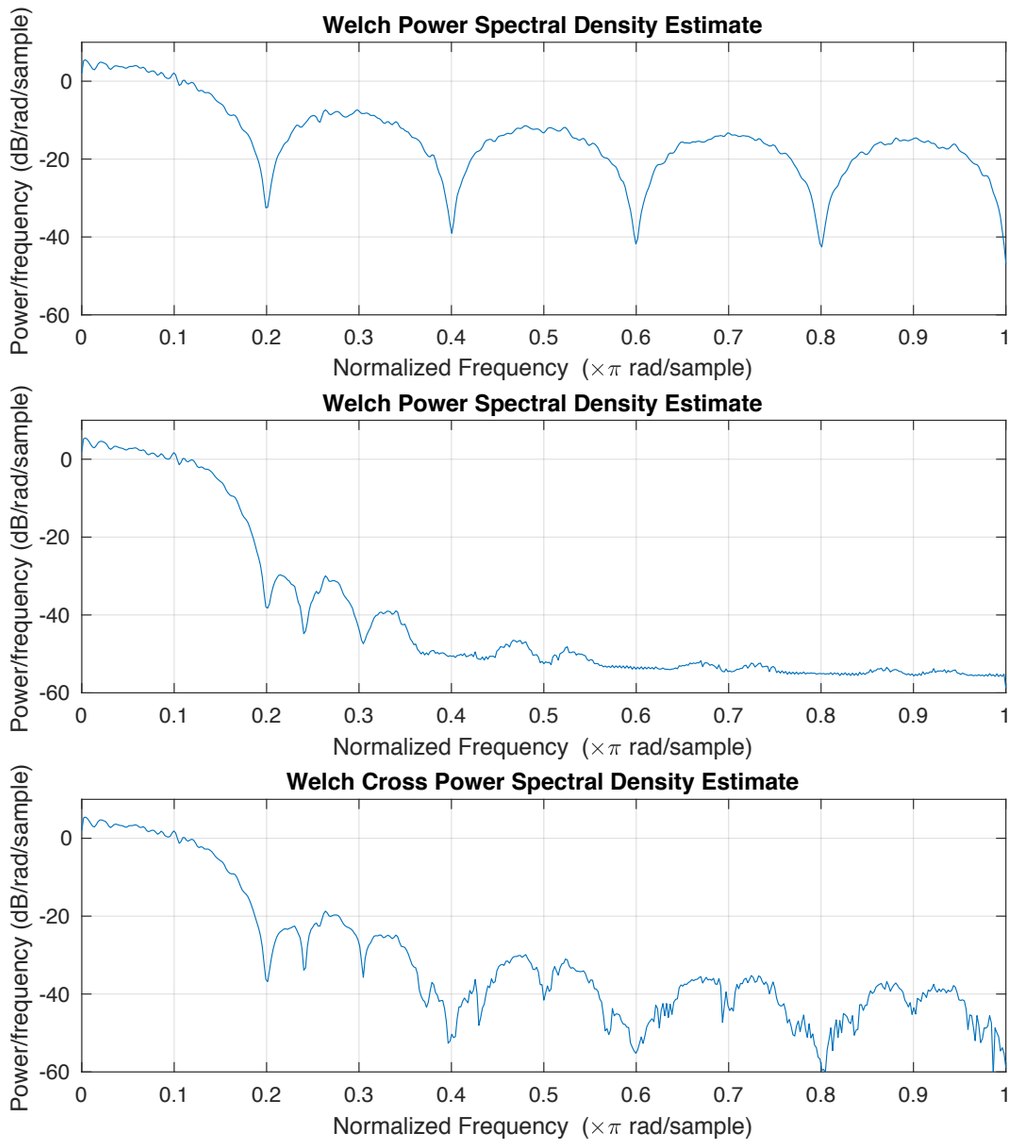
Cross-correlation → $R_{xy}(m) = E\left\{x_{n+m} y_n^*\right\} = E\left\{x_n y_{n-m}^*\right\}$

Cross power spectral density

- “cpsd.m” function

[Pxy,F] = cpsd(x,y,window,noverlap,nfft,fs)

```
% MATLAB code
h1 = ones(1,10)/sqrt(10);
h2 = fir1(30,0.2,rectwin(31));
r = randn(16384,1);
x = filter(h1,1,r);
y = filter(h2,1,x);

figure('Color',[1 1 1]);
subplot(311); pwelch(x,500,250,1024); ylim([-60 10])
subplot(312); pwelch(y,500,250,1024); ylim([-60 10])
subplot(313); cpsd(x,y,500,250,1024); ylim([-60 10])
```



MATLAB help files

Time-frequency analysis of two signals

- The cross examination of the two CWT decompositions obtained from two signals
  - can reveal localized similarities in time-scale plane.
  - can reveals common time-varying patterns
  - Wavelet cross spectrum and coherence are two measures

Wavelet cross spectrum

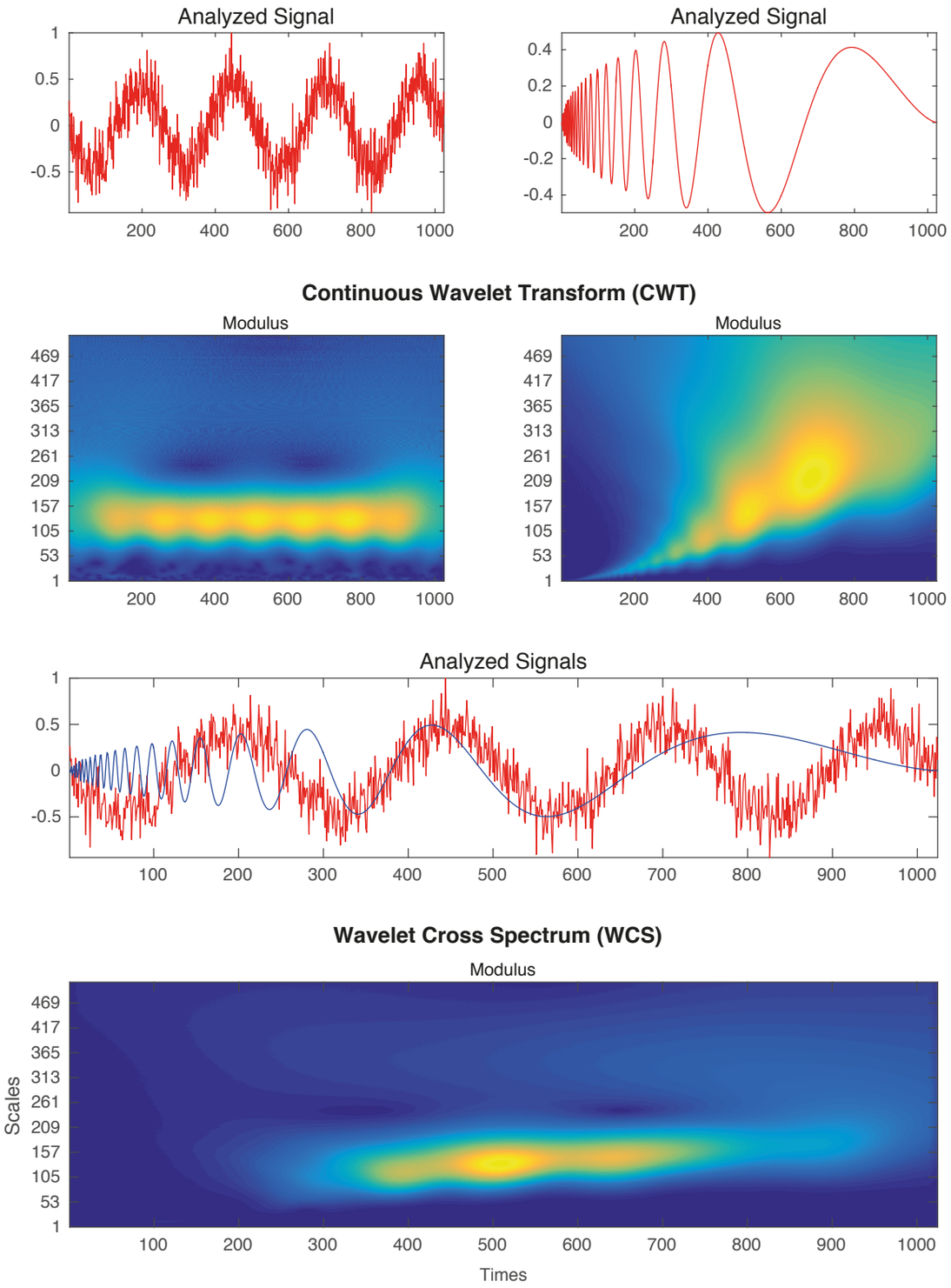
- The cross spectrum of two signals (x,y) denotes the fraction of covariance at each scale a and time b.

$$C_{xy}(a,b) = S(C_x^*(a,b)C_y(a,b))$$

- The phase spectrum denotes the phase difference between the signals at each scale a and time b.

Wavelet cross spectrum - MATLAB

```
% MATLAB code

wname  = 'cgau2'; % wavelet function
scales = 1:512;   % scales to plot
ntw = 21;         % length of moving average

t = linspace(0,1,1024); % time
x = -sin(8*pi*t) + 0.4*randn(1,1024);
x = x/max(abs(x));        % first signal
y = wnoise('doppler',10); % second signal

wcoher(x,y,scales,wname,'ntw',ntw,'plot','cwt');

figure;
wcoher(x,y,scales,wname,'ntw',ntw,'nsw',1,'plot','wcs');
```



MATLAB help files

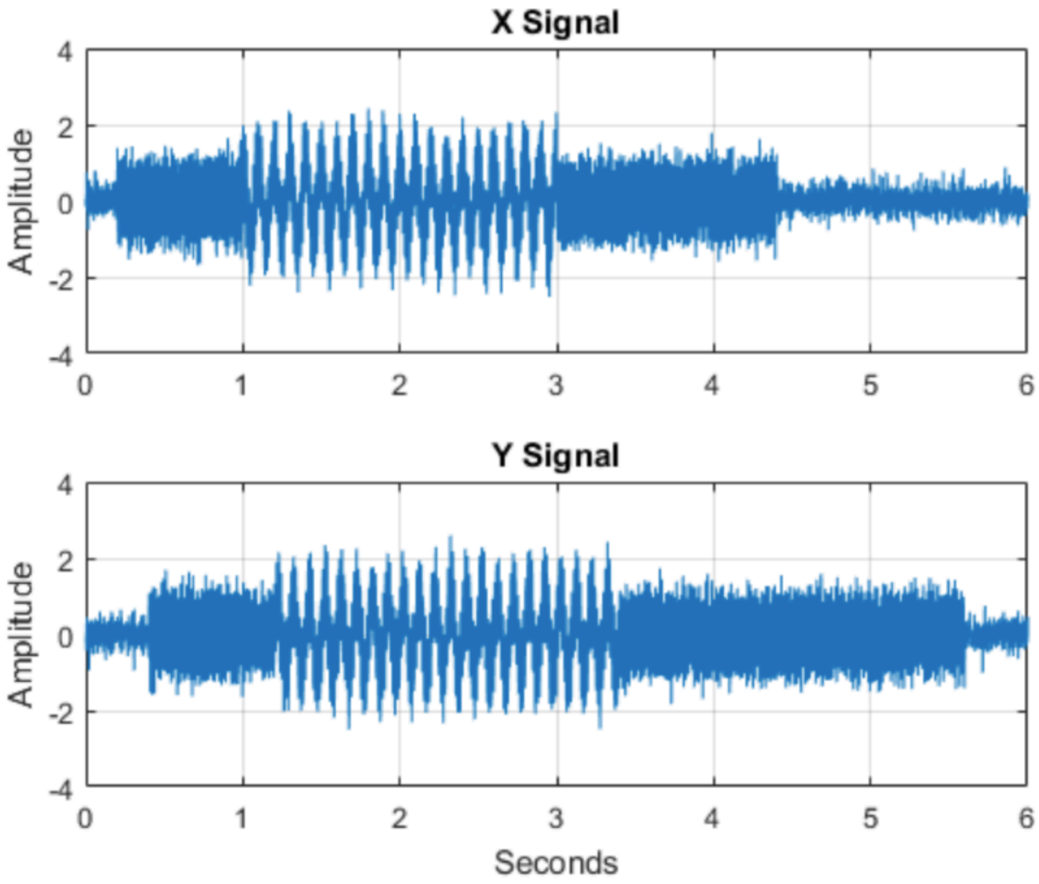
Wavelet coherence

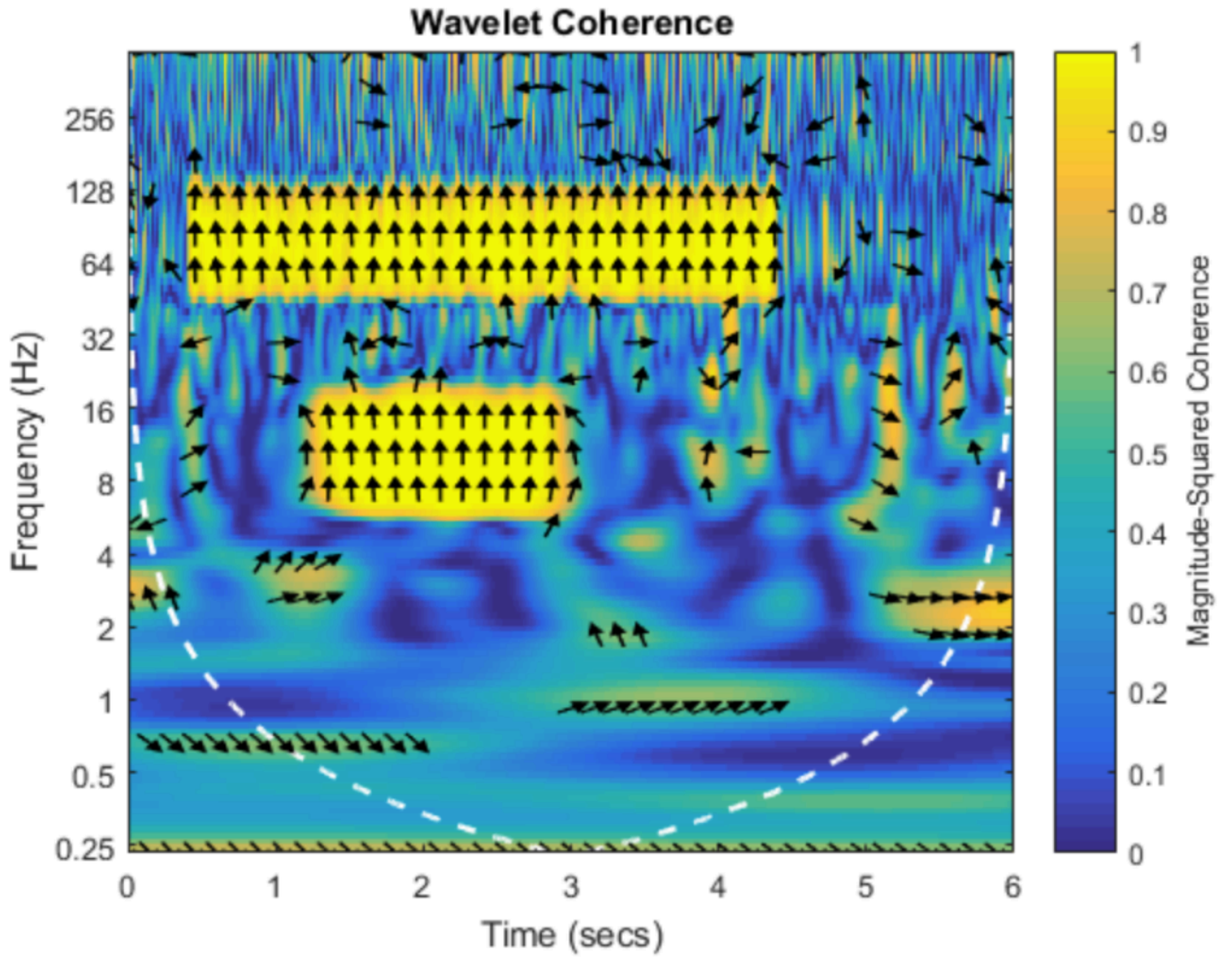
- Time-varying coherence is a powerful tool for revealing functional dynamics between different regions in the brain.
- Measure of the correlation between signals x and y in the time-frequency plane
- Is the normalized cross-spectrum with respect to the spectrum of each signal (a value between 0 and 1)

$$\frac{\left|S(C_x^*(a,b)C_y(a,b))\right|^2}{S(\left|C_x(a,b)\right|^2 S(\left|C_y(a,b)\right|^2}$$

Wavelet coherence

- “wcoherence.m” function of MATLAB provides





MATLAB help files

Questions?