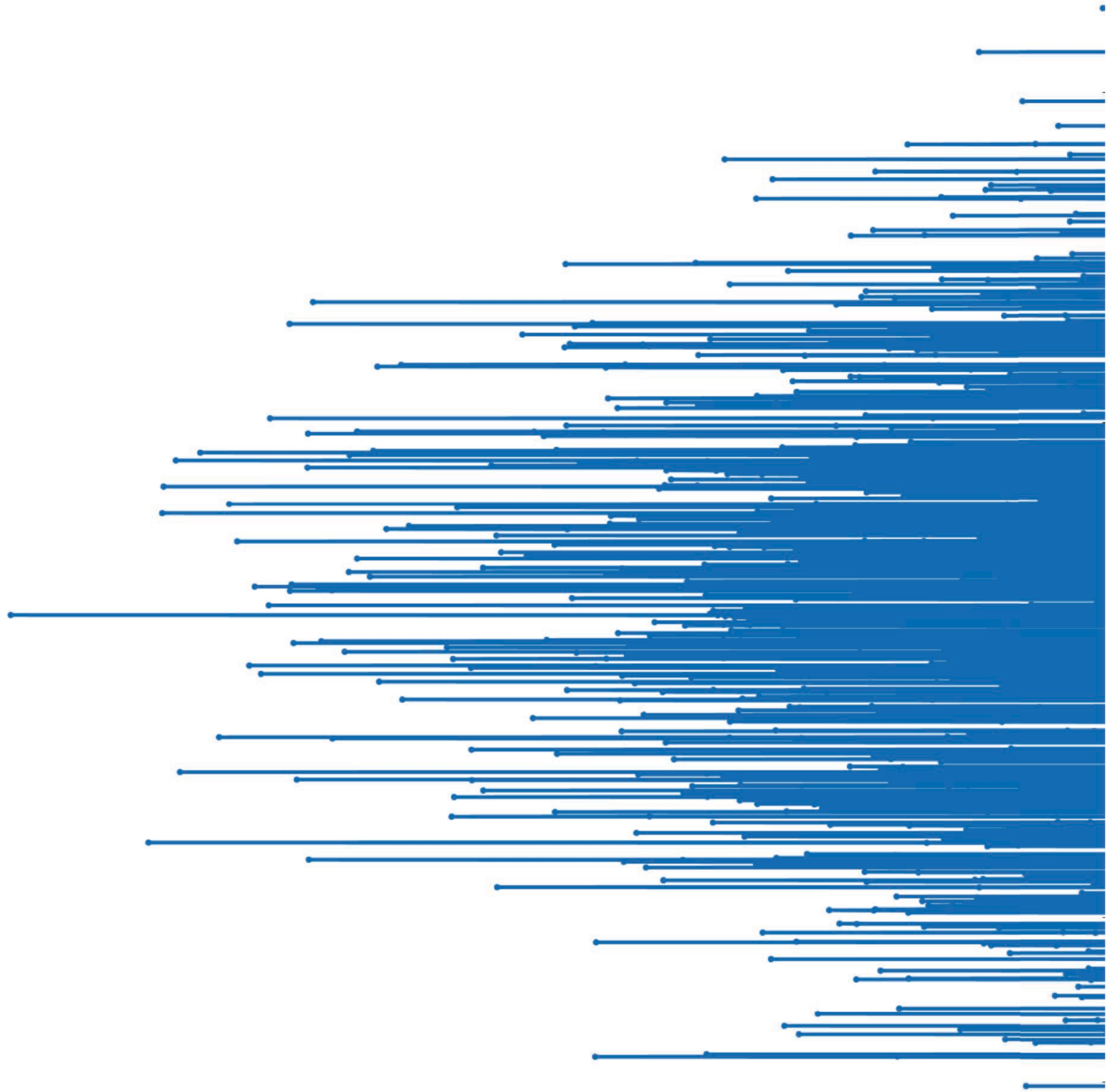
PHYS 2010 (W20)
Classical Mechanics



2020.03.05

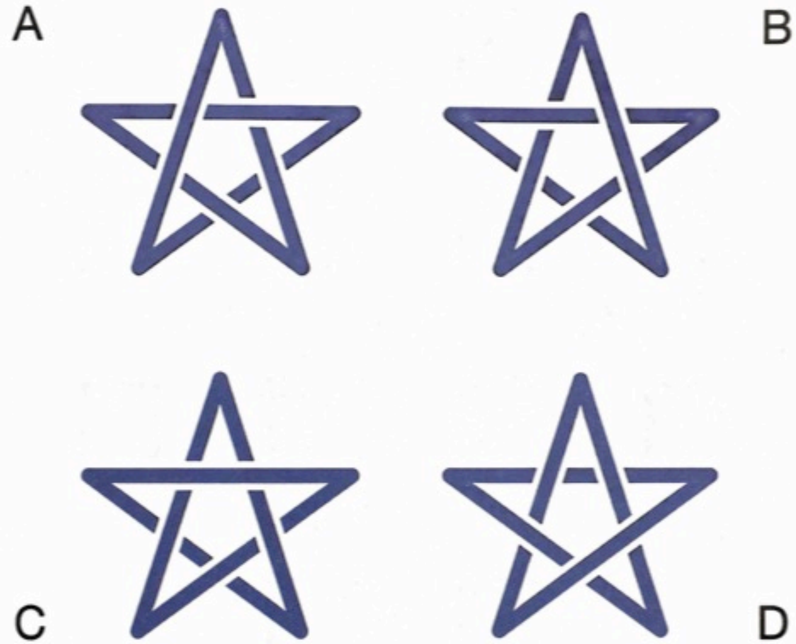
Relevant reading:

Knudsen & Hjorth: X

Christopher Bergevin
York University, Dept. of Physics & Astronomy
Office: Petrie 240   Lab: Farq 103
cberge@yorku.ca

Ref.s:
Knudsen & Hjorth (2000), Fowles & Cassidy (2005)

239. Tangled Stars

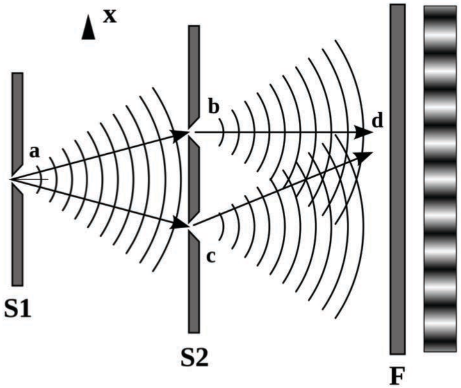


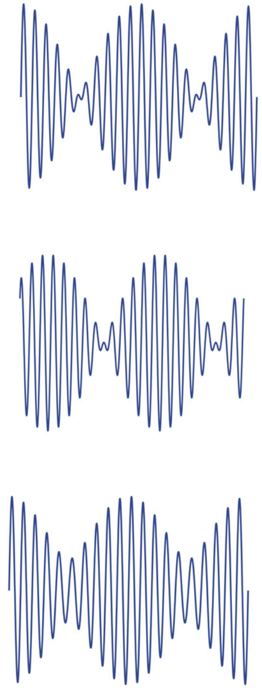
Three of the four tangled stars are identical. Which one is different?

A B C D

Looking Ahead...

Superposition





DDHO is a linear system

$$m\ddot{x}+c\dot{x}+kx=\sum_n(m\ddot{x}_n+c\dot{x}_n+kx_n)=\sum_n F_n(t)=F_{ext}$$

Periodic forcing & superposition

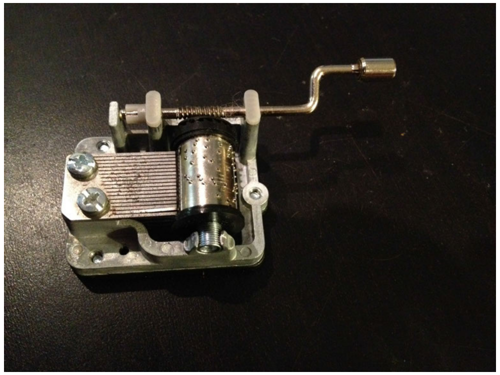
$$F_{ext}=\sum_n F_n(t)$$

Fourier analysis

$$f(t)=\tfrac{1}{2}a_0+\sum_{n=1}^{\infty}[a_n\ \cos(n\omega t)+b_n\ \sin(n\omega t)]$$

$$a_n=\frac{2}{T}\int_{-T/2}^{T/2}f(t)\cos(n\omega t)\,dt \qquad n=0,1,2,\ldots$$

$$b_n=\frac{2}{T}\int_{-T/2}^{T/2}f(t)\sin(n\omega t)\,dt \qquad n=1,2,\ldots$$



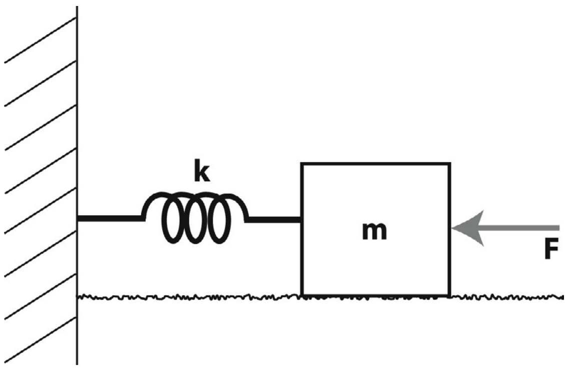
Impulse response

➢ Intuitively defined in two different (but equivalent) ways:

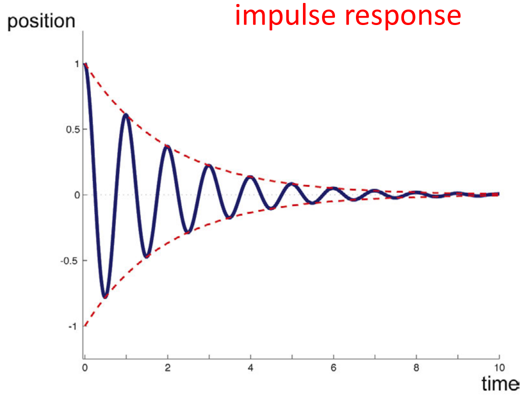
1. Time response of 'system' when subjected to an impulse
   (e.g., striking a bell w/ a hammer)

2. Fourier transform of resulting response
   (e.g., spectrum of bell ringing)

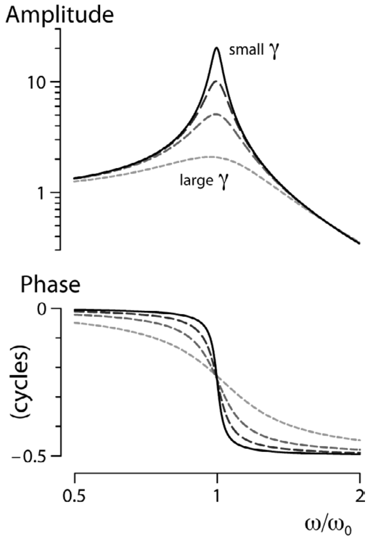
ex. Harmonic oscillator



Temporal impulse response

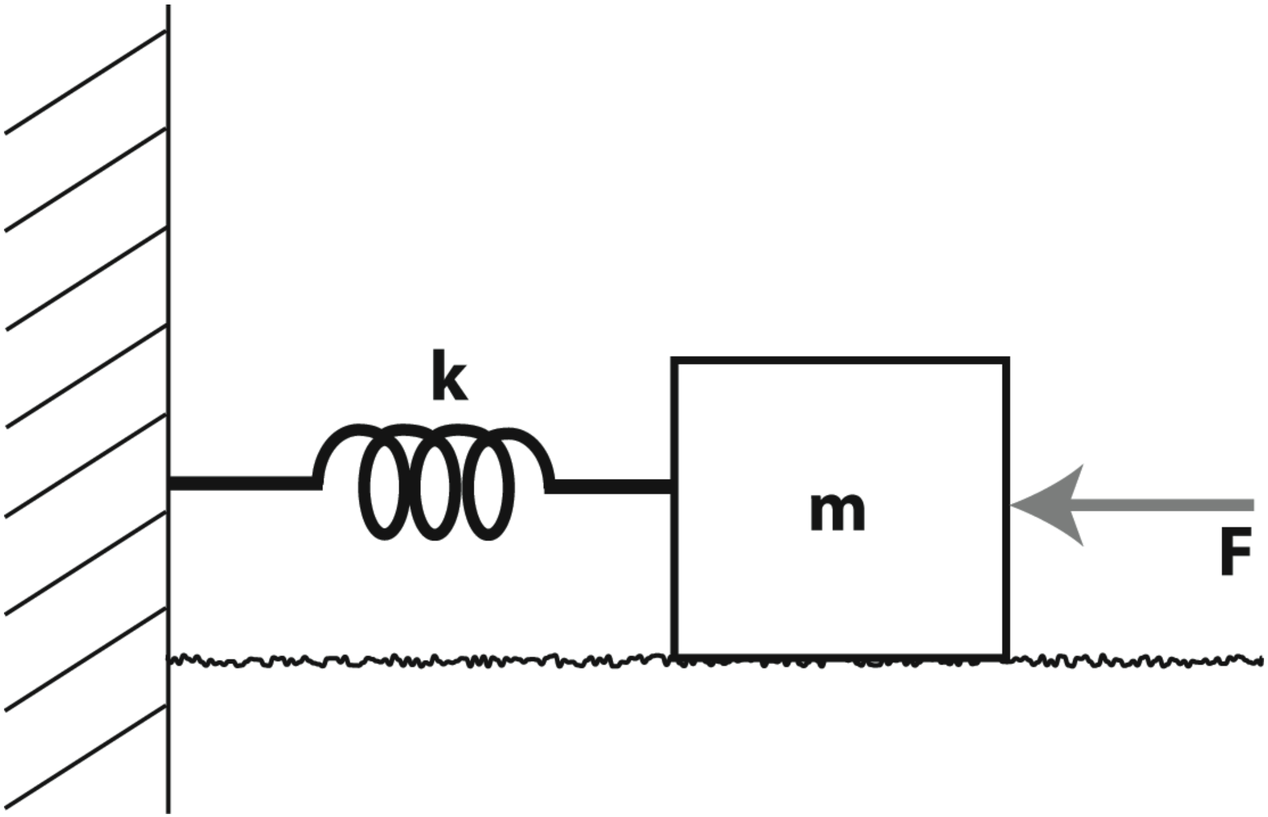


Spectral impulse response



(Important) Note: The Fourier transform of the impulse response is called the *transfer function*

<u>Superposition & Linearity</u>

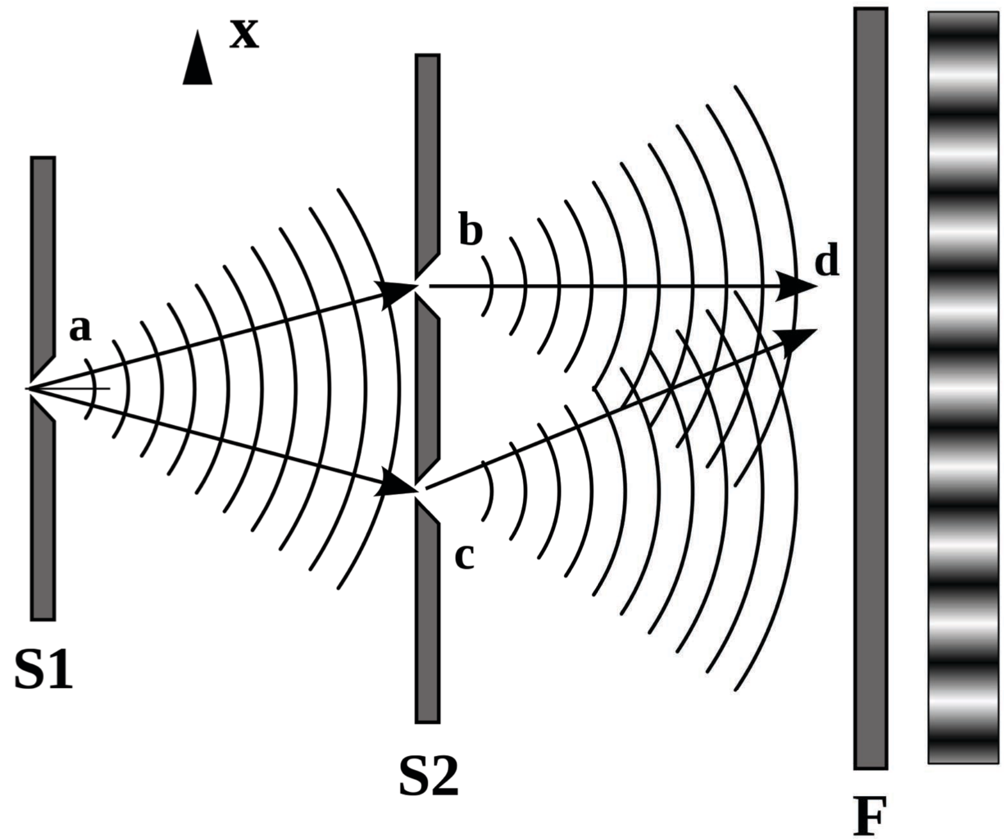


→ When dealing with linear oscillators (or linear systems in general), superposition takes a domineering position in how we approach analysis and modeling

Superposition

"Double-slit experiment"

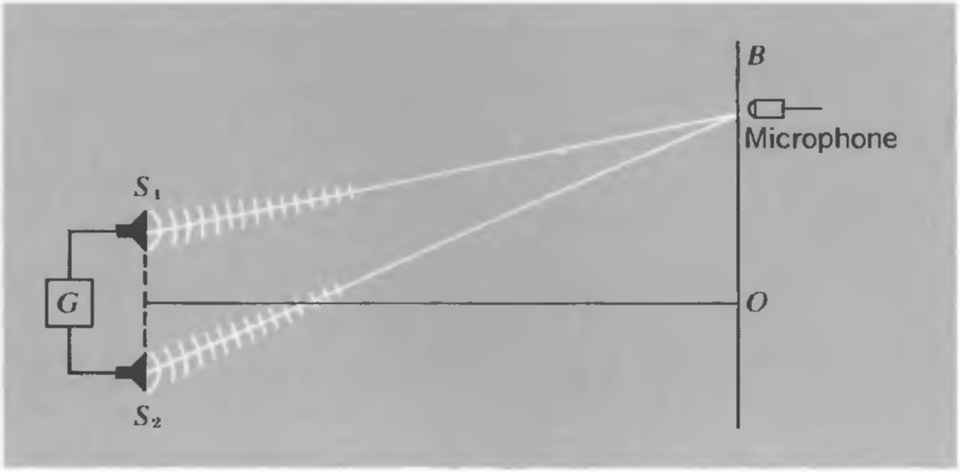
Thomas Young (1773-1829)



→ Wave theory of light
(via *phase interference*)

Wikipedia

Superposition



Each source:

$$x_1 = A_1 \cos(\omega t + \alpha_1)$$
$$x_2 = A_2 \cos(\omega t + \alpha_2)$$

Their sum (at the mic)

$$x = x_1 + x_2 = A_1 \cos(\omega t + \alpha_1) + A_2 \cos(\omega t + \alpha_2)$$

$$= A \cos(\omega t + \alpha)$$

A tad messy to solve for the constants there....

If instead we used complex exponentials:

$$z_1 = A_1 e^{j(\omega t + \alpha_1)}$$
$$z_2 = A_2 e^{j(\omega t + \alpha_2)}$$

Then:

$$z = z_1 + z_2 = A_1 e^{j(\omega t + \alpha_1)} + A_2 e^{j(\omega t + \alpha_2)}$$

$$= e^{j(\omega t + \alpha_1)}[A_1 + A_2 e^{j(\alpha_2 - \alpha_1)}]$$

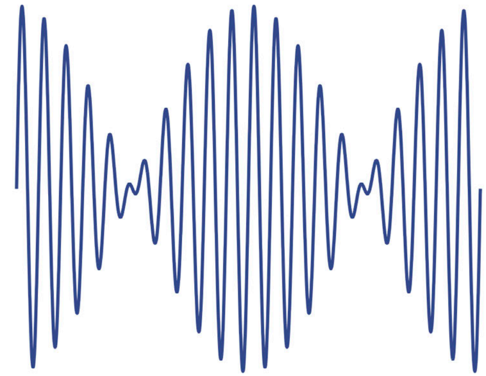
→ This latter equation, while seemingly intimidating, tells us a lot!

French (1971)

Superposition: "Beats" (adding two different frequencies)

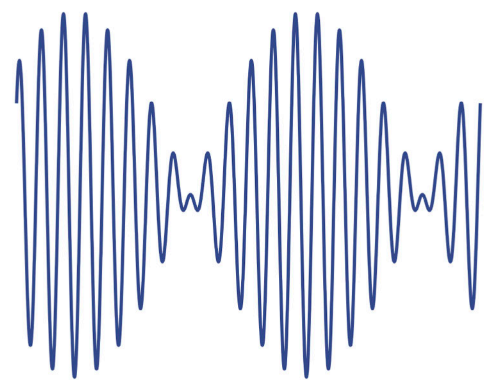
$$x(t) = A_1 \sin(2\pi f_1 t + \phi_1) + A_2 \sin(2\pi f_2 t + \phi_2)$$

$f_1=1, f_2=1.1$
$A_1=1, A_2=1$
$\phi_1=0, \phi_2=0$



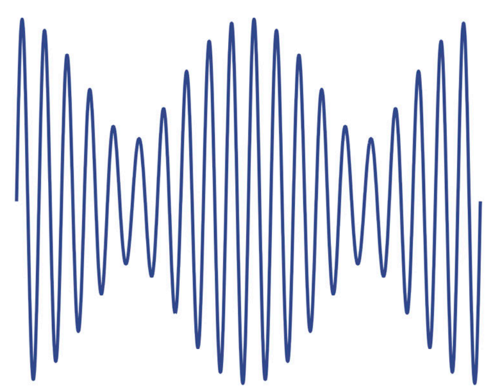
→ Changing (relative) phase affects summation

$f_1=1, f_2=1.1$
$A_1=1, A_2=1$
$\phi_1=\pi/2, \phi_2=0$



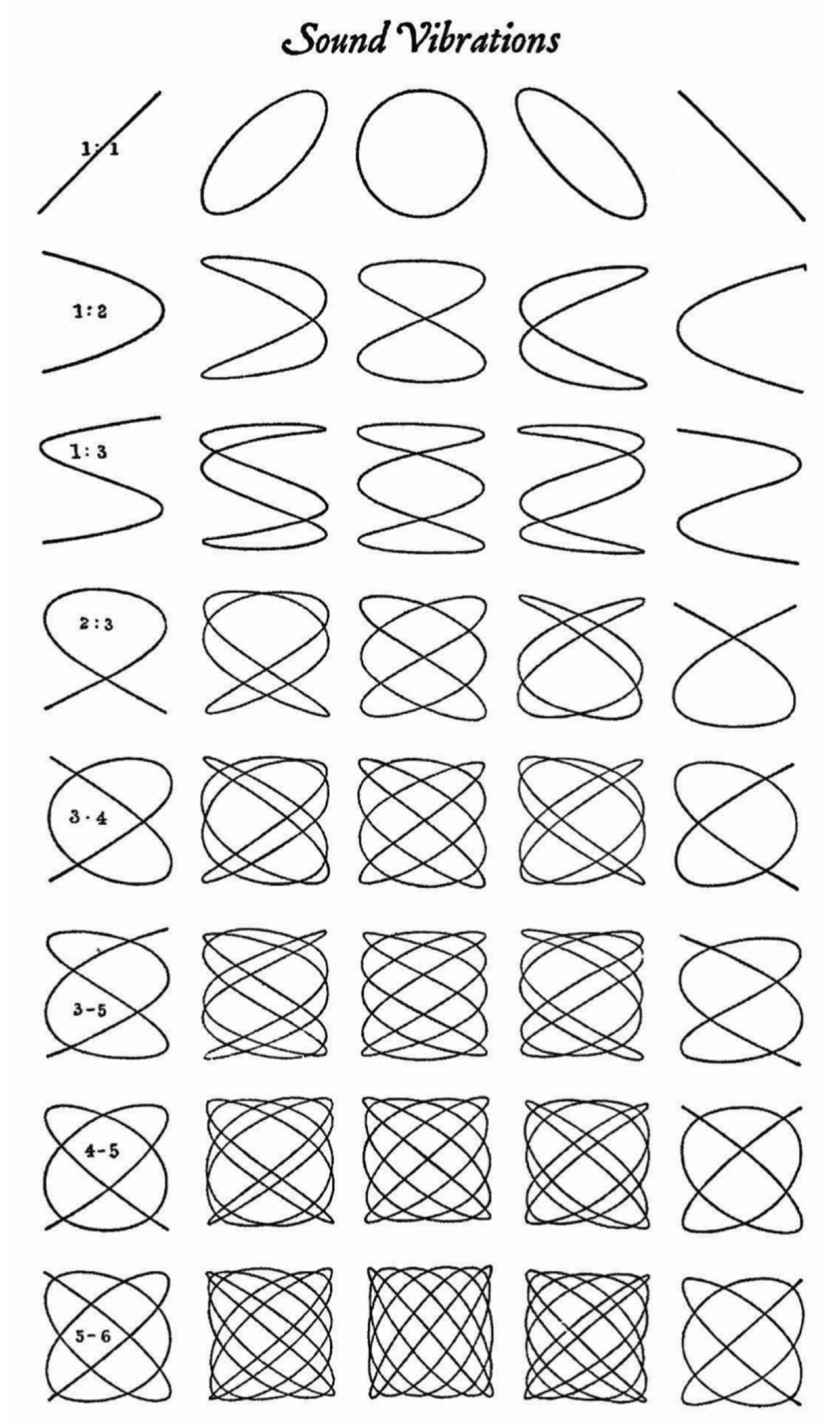
→ Changing (relative) amplitudes affects summation

$f_1=1, f_2=1.1$
$A_1=2, A_2=1$
$\phi_1=0, \phi_2=0$

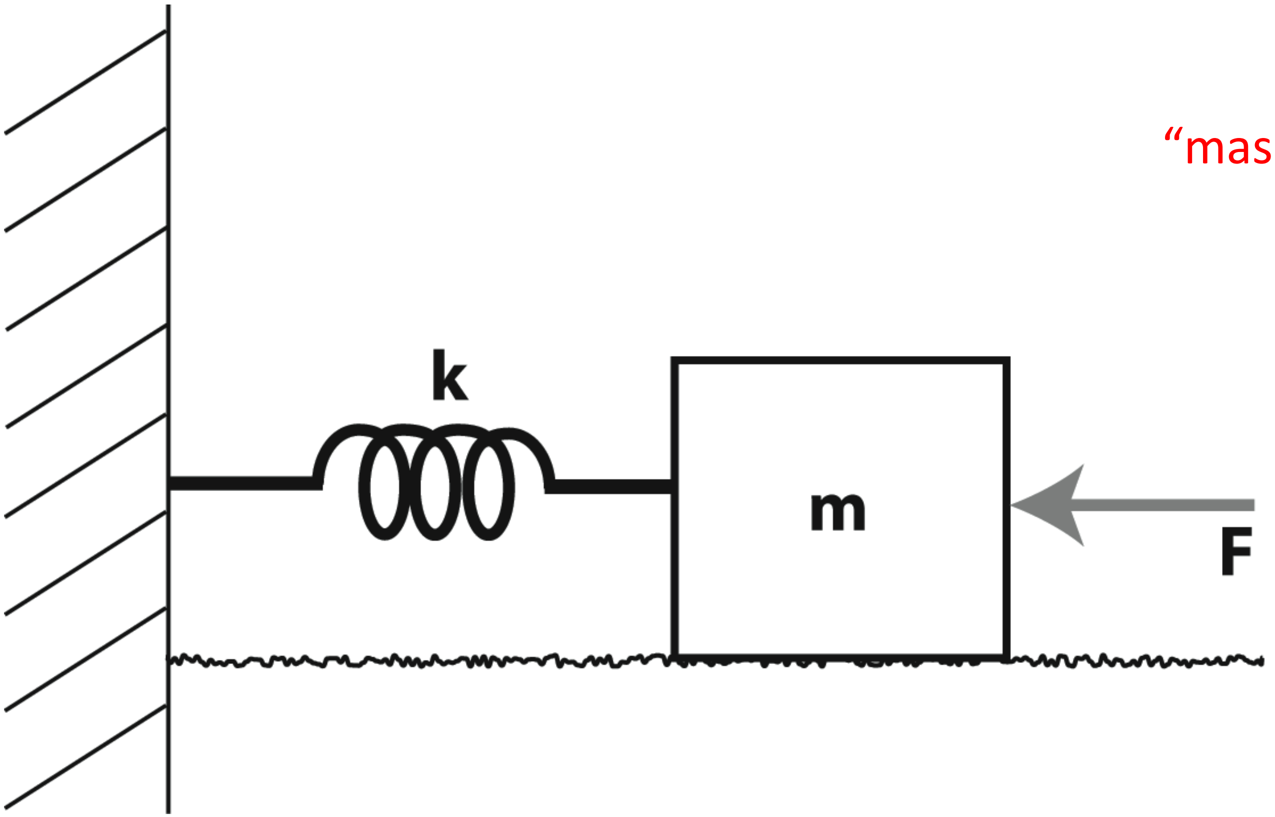


<u>Recall</u>: Lissajou figures

Jules Antoine Lissajous (1822-1880)



Tying this all back together....



"mass-on-a-spring"

- One of the more fundamental/canonical problems in all areas of physics...

Fourier & the DDHO

Consider the driven case

$$m\ddot{x} + c\dot{x} + kx = F_{ext}$$

where the driving force is a superposition of individual forces

$$F_{ext} = \sum_n F_n(t)$$

Linearity allows us consider each separately...

$$m\ddot{x}_n + c\dot{x}_n + kx_n = F_n(t)$$

... and simply add them all back up!

$$x(t) = \sum_n x_n(t)$$

$$m\ddot{x} + c\dot{x} + kx = \sum_n (m\ddot{x}_n + c\dot{x}_n + kx_n) = \sum_n F_n(t) = F_{ext}$$

For linear systems, the "door" of superposition swings both ways!

Fowles & Cassidy

Fourier & the DDHO

Now, can we make a smart choice for what all those "individual" forces are?

$$F_{ext} = \sum_n F_n(t)$$

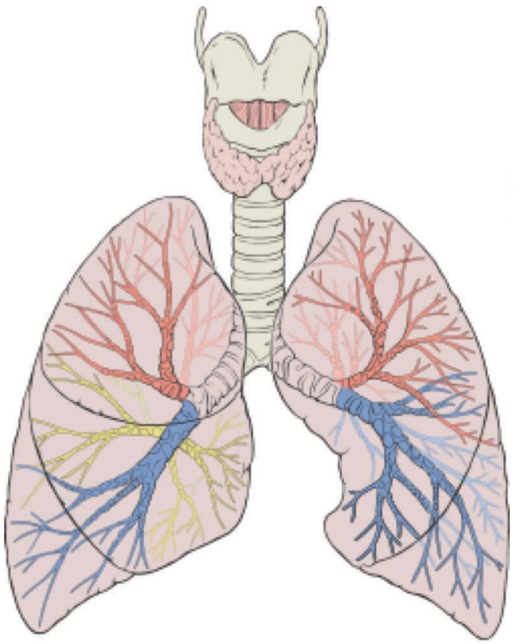
In many instances, the driving force is something that eventually repeats itself*

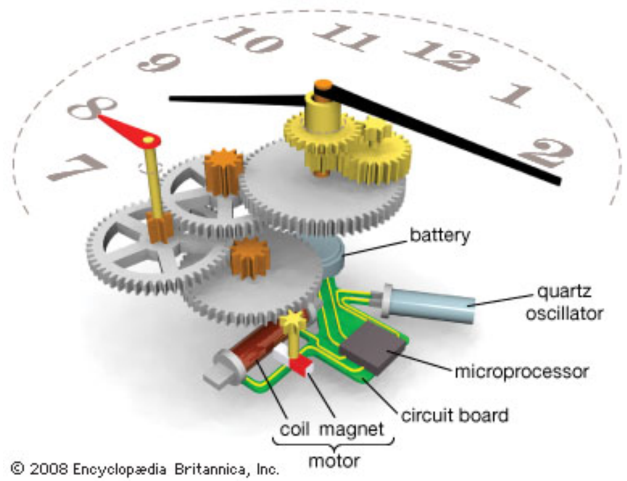
$$F_{ext}(t) = F_{ext}(t + T)$$

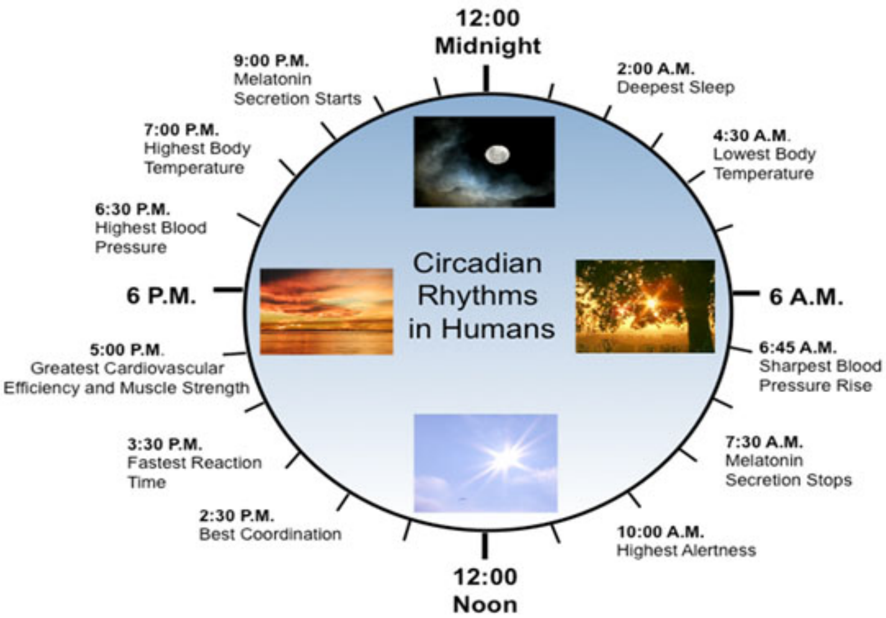
* - Note that much of what we discuss here also usefully applies even when a given "signal" is not strictly periodic (or even remotely close to such!)

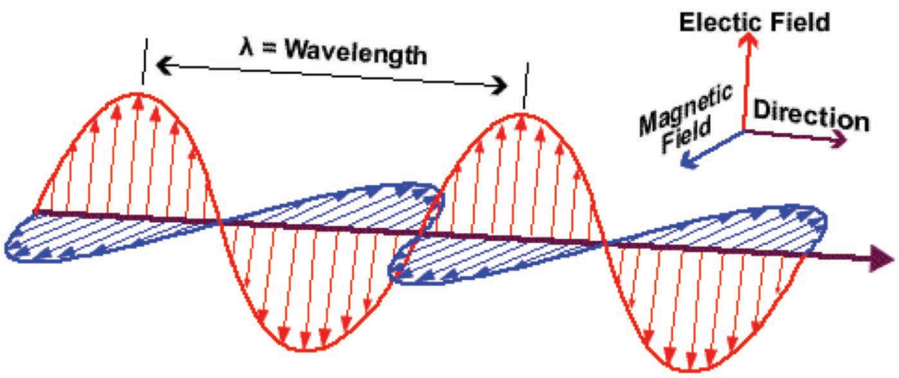
Fowles & Cassidy

Periodic forcing....









Fourier & the DDHO

Now, can we make a smart choice for what all those "individual" forces are?

$$F_{ext} = \sum_n F_n(t)$$

In many instances, the driving force is something that eventually repeats itself

$$F_{ext}(t) = F_{ext}(t+T)$$

Then it is fairly natural to represent such a function via an infinite series of sinusoids (i.e., a **Fourier series**)

$$f(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty}[a_n \cos(n\omega t) + b_n \sin(n\omega t)]$$

Fowles & Cassidy

Fourier & the DDHO

$$f(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty}[a_n \cos(n\omega t) + b_n \sin(n\omega t)]$$

Just like a Taylor series expansion, there are a unique set of coefficients (a_n and b_n) that determine the appropriate amplitude and phase of a given frequency term (i.e., $n\omega$)

There is a reasonably clear "recipe" for these coefficients...

$$a_n = \frac{2}{T}\int_{-T/2}^{T/2} f(t)\cos(n\omega t)\,dt \qquad n = 0, 1, 2, \ldots$$

$$b_n = \frac{2}{T}\int_{-T/2}^{T/2} f(t)\sin(n\omega t)\,dt \qquad n = 1, 2, \ldots$$

We can also use complex numbers to manage "bookkeeping" more efficiently....

$$f(t) = \sum_{n} c_n e^{in\omega t} \qquad n = 0, \pm 1, \pm 2, \ldots$$

$$c_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t)e^{-in\omega t}\,dt$$

Fowles & Cassidy

Fourier analysis

- Deep history throughout mathematics, physics, engineering, biology, ...

- Backbone of modern signal processing and linear systems theory

- Lays at foundation of many modern methodologies in medical imaging (e.g., MRI, CT scans)

- Builds off the basic idea of a *Taylor series* (which posits we can describe a function as an infinite series of polynomials)

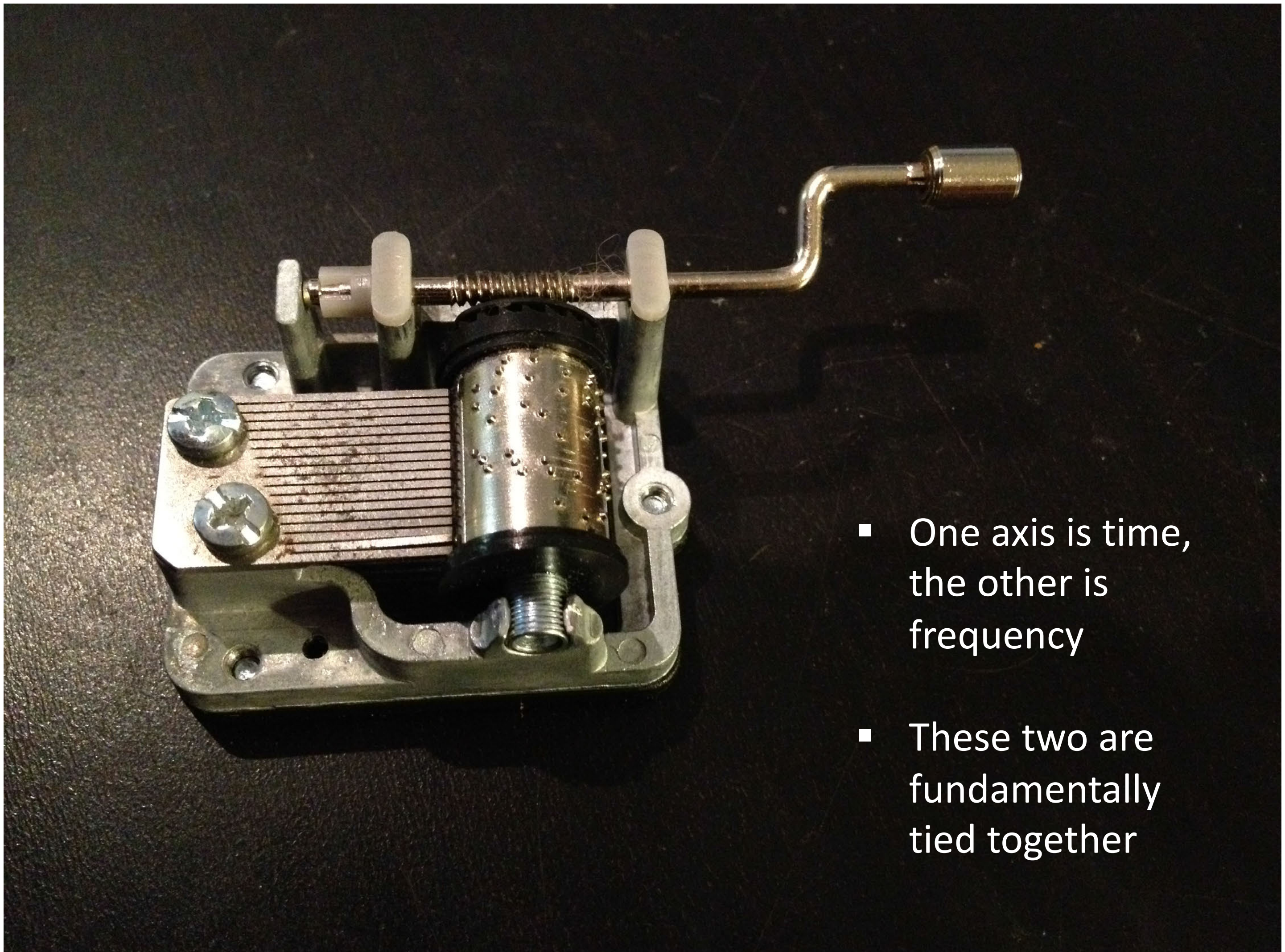
Joseph Fourier (1768-1830)

Basic idea: Represent 'signal' as a sum of sinusoids

Note: We focus on 1-D here for clarity, but these ideas generalize to higher dimensions (e.g., 2-D for images)

<u>Key idea:</u> Fourier transform

- Allows one to go from a time domain description (e.g., recorded signal) to a spectral description (i.e., what frequency components make up that signal)



Fourier series

Intuitive connection back to Taylor series:

$$y(x_1 + \Delta x) \approx y(x_1) + \sum_{n=1}^{N} \frac{1}{n!} \left. \frac{d^n y}{dx^n} \right|_{x_1} (\Delta x)^n. \qquad \text{(D.2)}$$

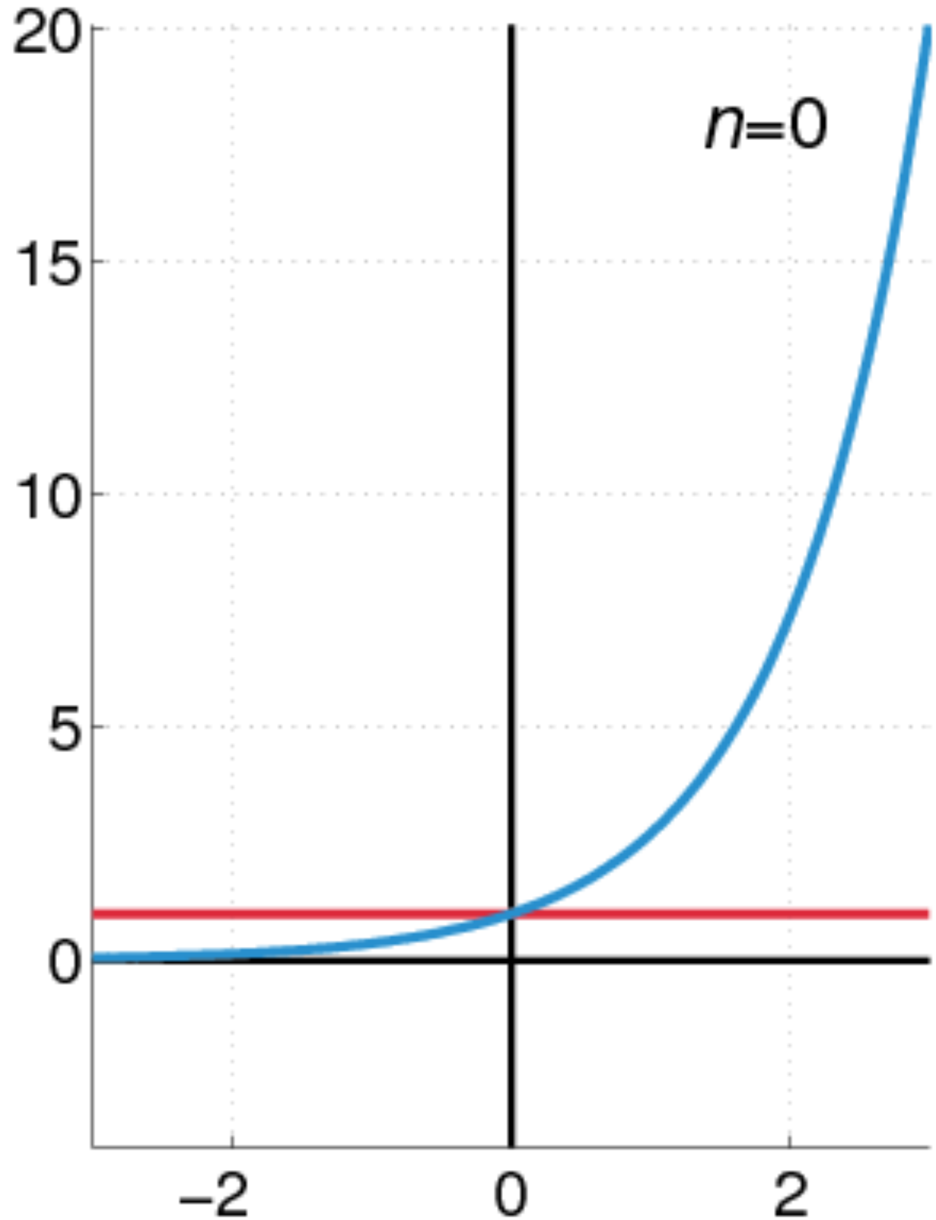
$$f(x) = f(x_o) + f'(x_o)(x - x_o) + \frac{f''(x_o)}{2!}(x - x_o)^2 + \cdots + \frac{f^{(n)}(x_o)}{n!}(x - x_o)^n + \cdots$$

$$= \sum_{n=0}^{\infty} \frac{f^{(n)}(x_o)}{n!}(x - x_o)^n$$

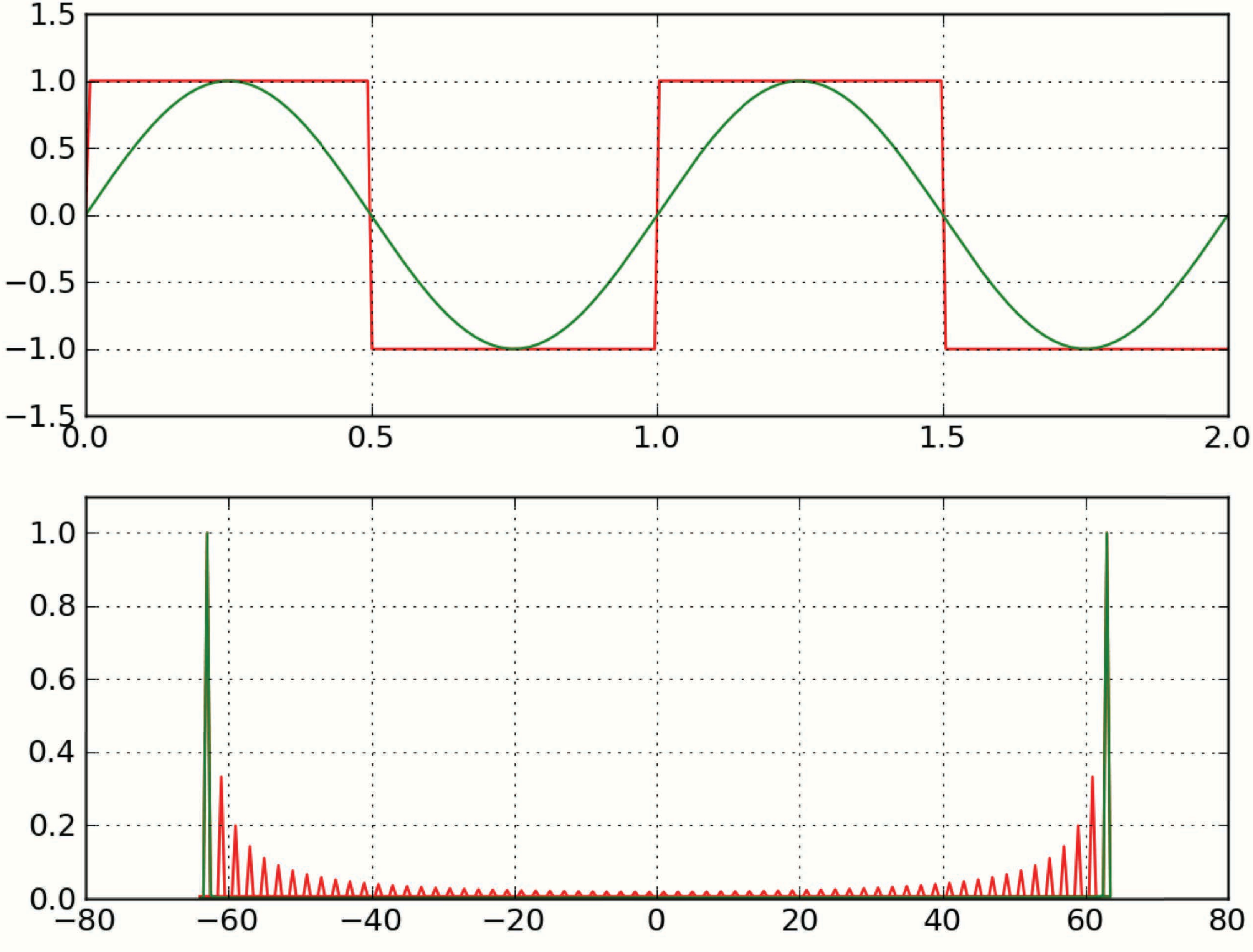
Taylor series → Expand as a (infinite) sum of polynomials

Different Idea: Fourier series → Expand as a (infinite) sum of sinusoids

"The exponential function e^x (in blue), and the sum of the first n+1 terms of its Taylor series at 0 (in red)."



wikipedia (Taylor series)



wikipedia (square wave)

Fourier series

$$f(t) = a_o + a_1 \sin(\omega t) + b_1 \cos(\omega t) + \\ + a_2 \sin(2\omega t) + b_2 \cos(2\omega t) + \\ + a_3 \sin(3\omega t) + b_3 \cos(3\omega t) + \cdots$$

$$= A_0 + A_1 \sin(\omega t + \phi_1) \\ + A_2 \sin(2\omega t + \phi_2) \\ + A_3 \sin(3\omega t + \phi_3) + \cdots$$

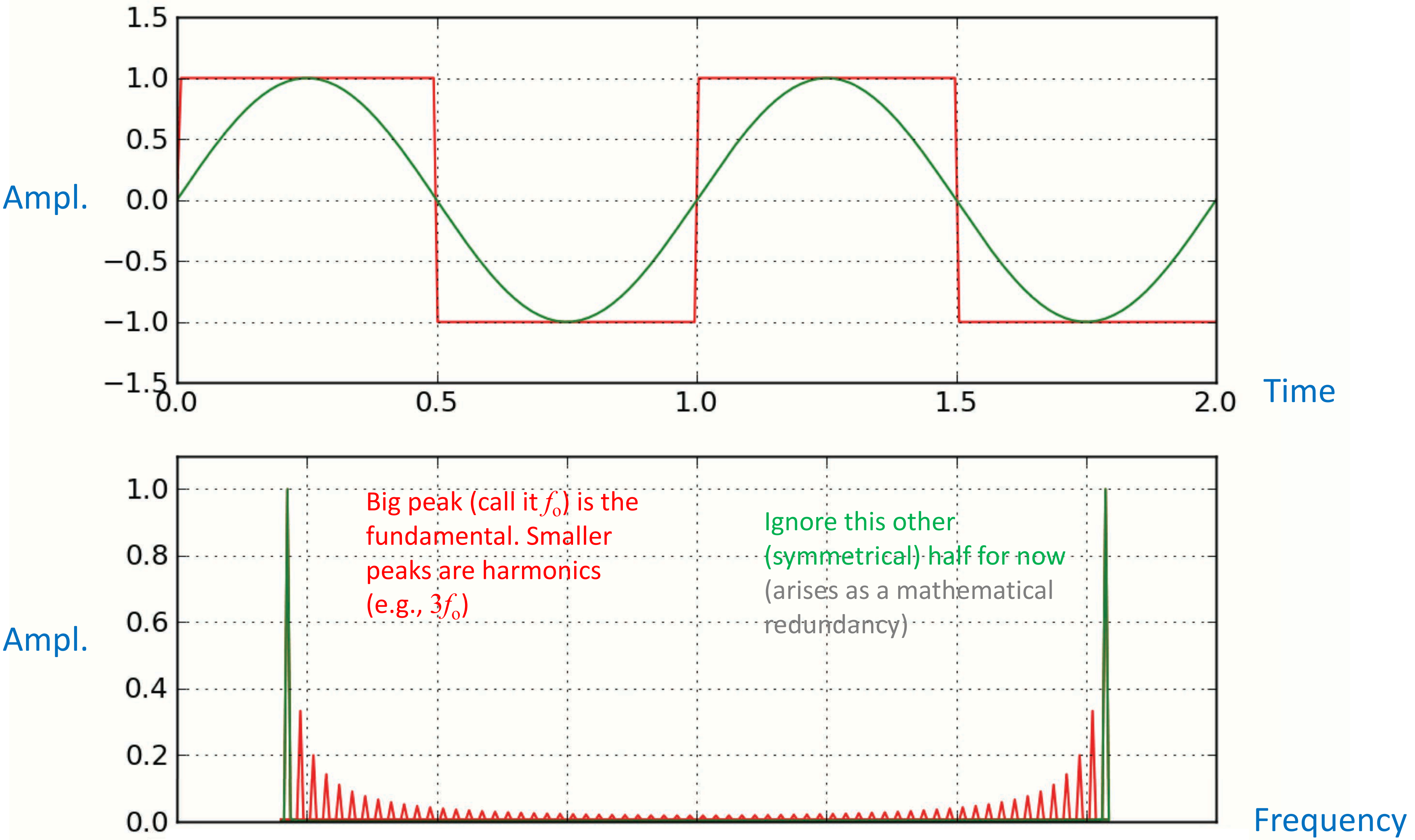
$$= \sum_{n=0}^{\infty} A_n \sin(n\omega t + \phi_n)$$

$$a_n = \frac{2}{T}\int_{-T/2}^{T/2} f(t)\cos(n\omega t)\,dt \qquad n = 0, 1, 2, \ldots$$

$$b_n = \frac{2}{T}\int_{-T/2}^{T/2} f(t)\sin(n\omega t)\,dt \qquad n = 1, 2, \ldots$$

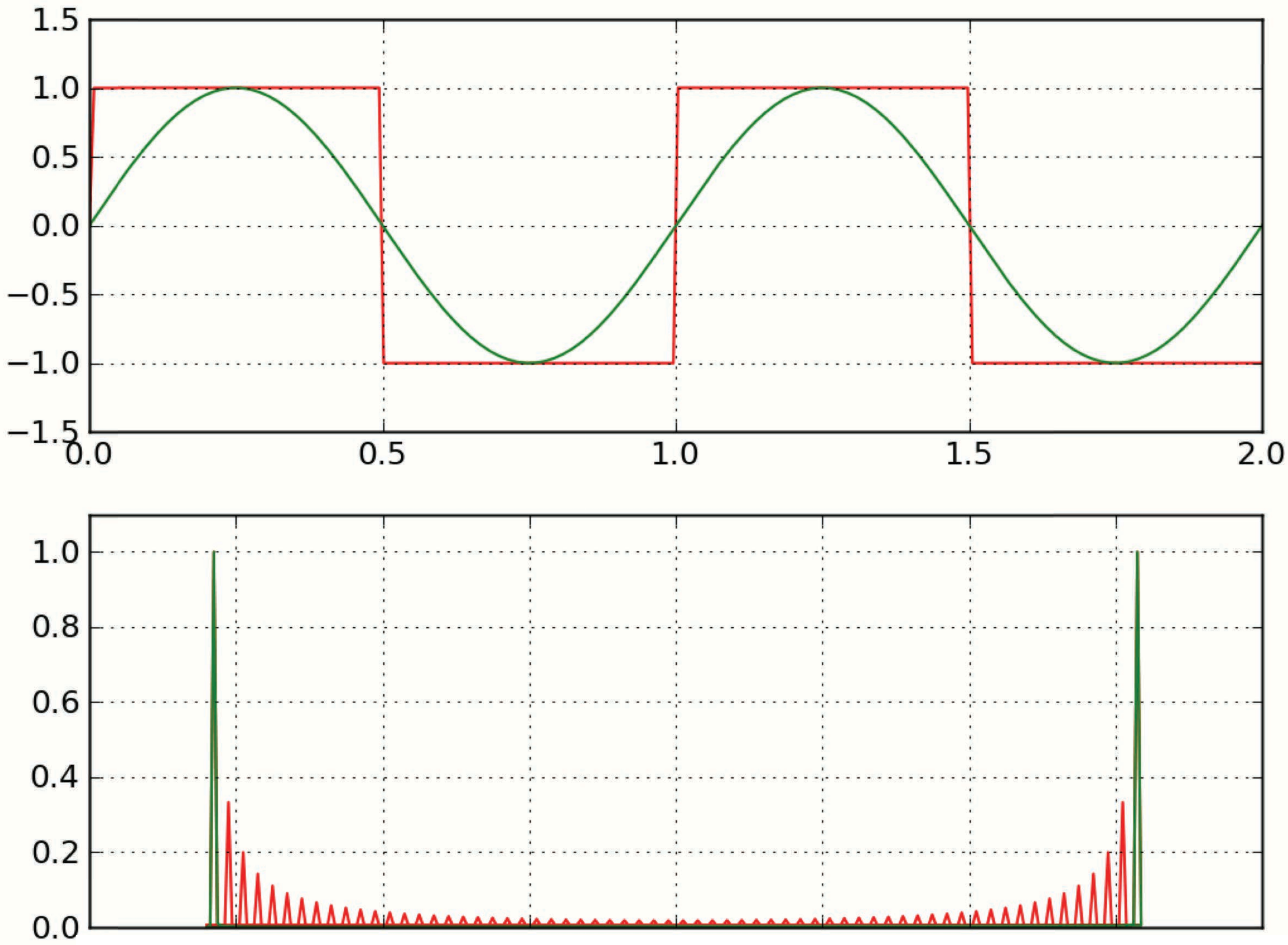
$$= \sum_{n=0}^{\infty} B_n e^{in\omega t} \qquad \text{where } B_n \in \mathbb{C},\ i = \sqrt{-1}$$

Complex #s are much more compact and easier to deal with e.g., Square "Waves"



wikipedia (square wave)

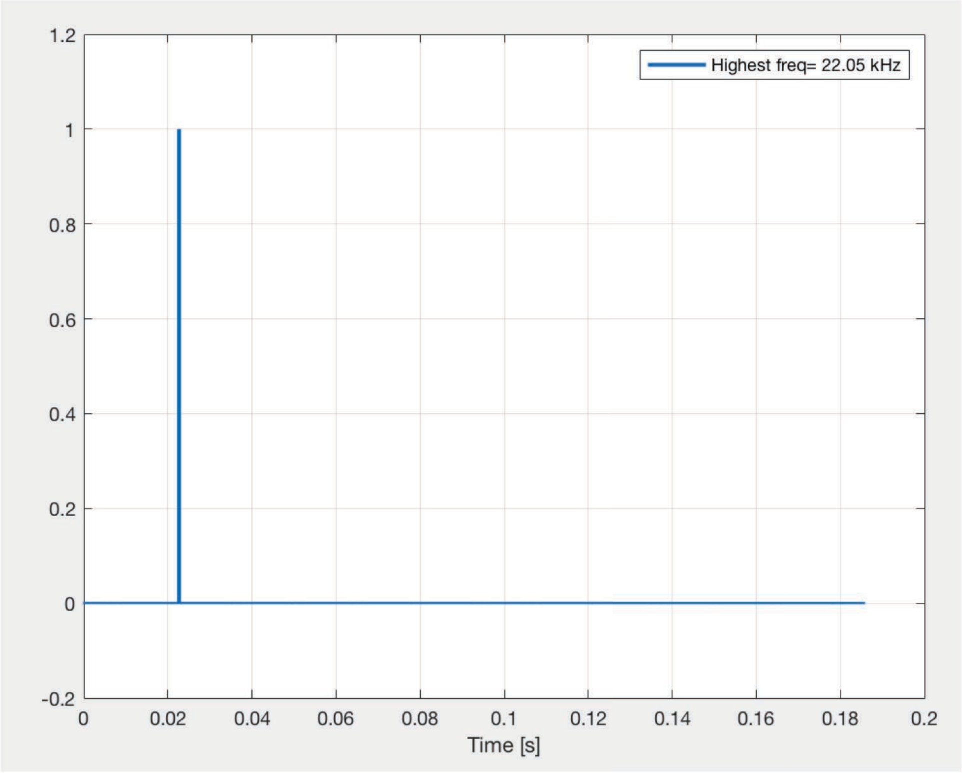
```
% ### EXsquareW.m ###    2017.02.19 C.Bergevin

% Visually demonstrate the build-up of a square wave
% by adding successive (user-specified) terms of the Fourier series
% expansion; also quantifies the Gibbs phenomonon

clear
% ===========================================================
P.order= [1 2 4 8 15 25 100 500];    % array of # of terms to compute {[1 2 4 8 15 25 100 500]}
P.tau= 1;    % period {1}
P.A= 1;      % peak-to-peak amplitude {1}
P.M= 10000;    % total # of points per interval (must be even?) {1000}
P.pause= 0.5;    % time to pause between displaying new iterates [s] {0.5}
% ===========================================================
t= linspace(-1.5*P.tau,1.5*P.tau,3*P.M);  % time array
squareT= repmat(P.A*([zeros(P.M/2,1);ones(P.M/2,1)]-0.5),3,1)';   % create sawtooth baseline
% --- use a loop to add in the terms
for mm=1:numel(P.order)
    tempN= P.order(mm);
    squareF= 0;    % dummy initial indexer
    for nn= 1:tempN
        nextTerm= (2*P.A/pi)*(1/(2*nn-1))*sin((2*nn-1)*(2*pi/P.tau)*t);  % create next term in 
series
        squareF= squareF+ nextTerm;
    end
    % --- estimate "overshoot"
    [M,indx]= max(squareF);
    disp(['Overshoot ratio ~',num2str(M/max(squareT))]);
    % --- visualize
    figure(1); clf;
    h1= plot(t,squareT,'b-','LineWidth',2); hold on; grid on; xlabel('x'); ylabel('y'); 
ylim(0.65*P.A*[-1 1]);
    h2= plot(t,squareF,'r--','LineWidth',2); legend([h1 h2],'square','Fourier 
series','Location','NorthWest');
    title(['(truncated) Fourier reconstruction of square wave w/ ',num2str(tempN),' terms']);
    pause(P.pause);
end
```


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.5
1.0
1.5
2.0
1.0
0.8
0.6
0.4
0.2
0.0

Recall: Impulses

$$\delta(x) = \begin{cases} +\infty, & x = 0 \\ 0, & x \neq 0 \end{cases}$$

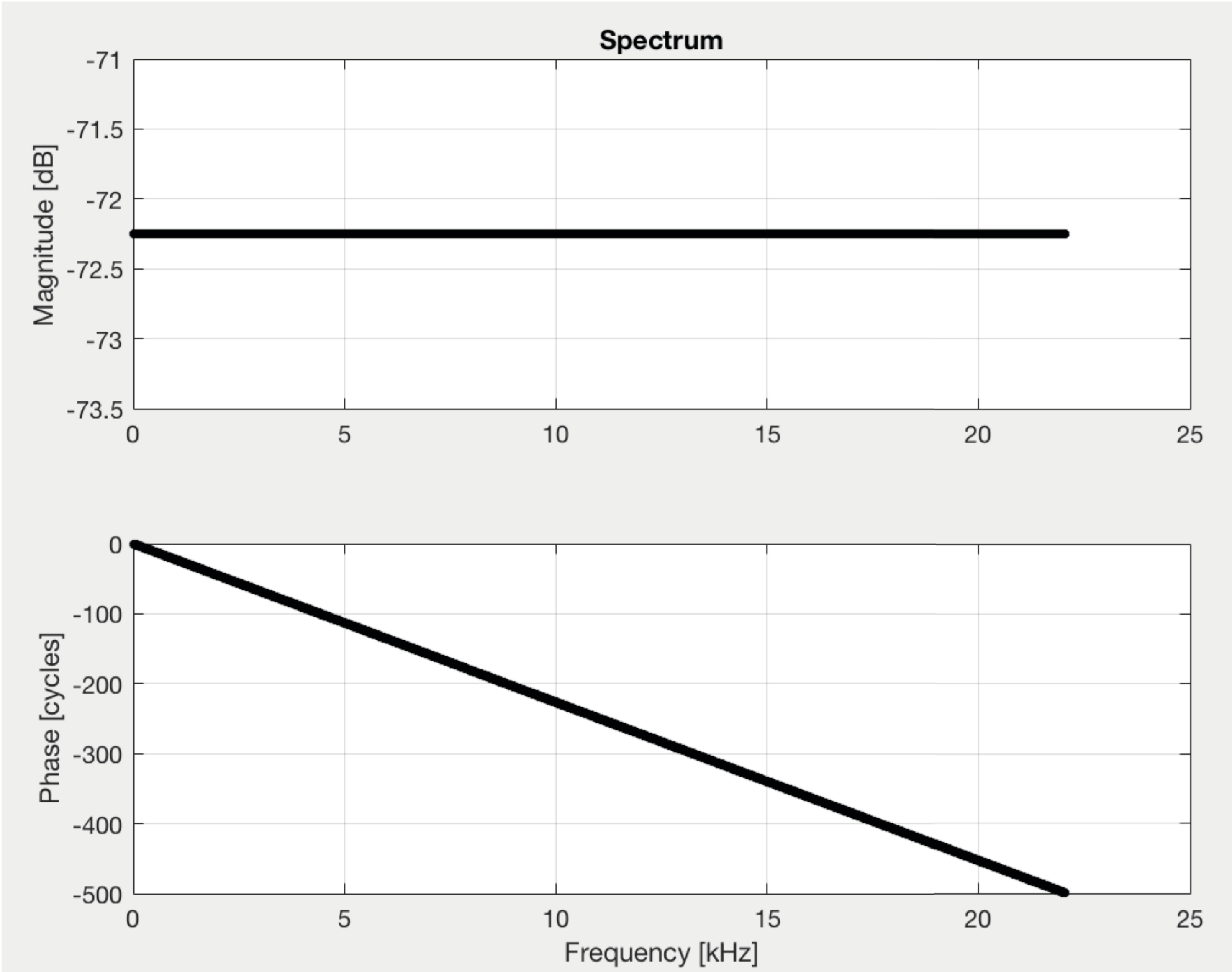
$$\int_{-\infty}^{\infty} \delta(x)\, dx = 1.$$



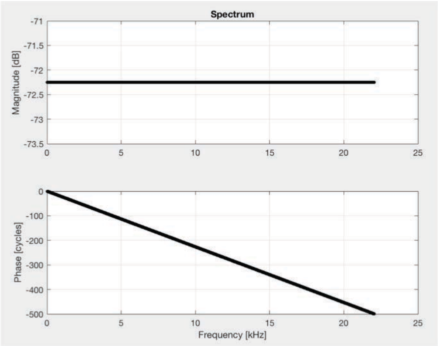
Delta functions are much more ubiquitous in your lives (and science) than you might think/realize......

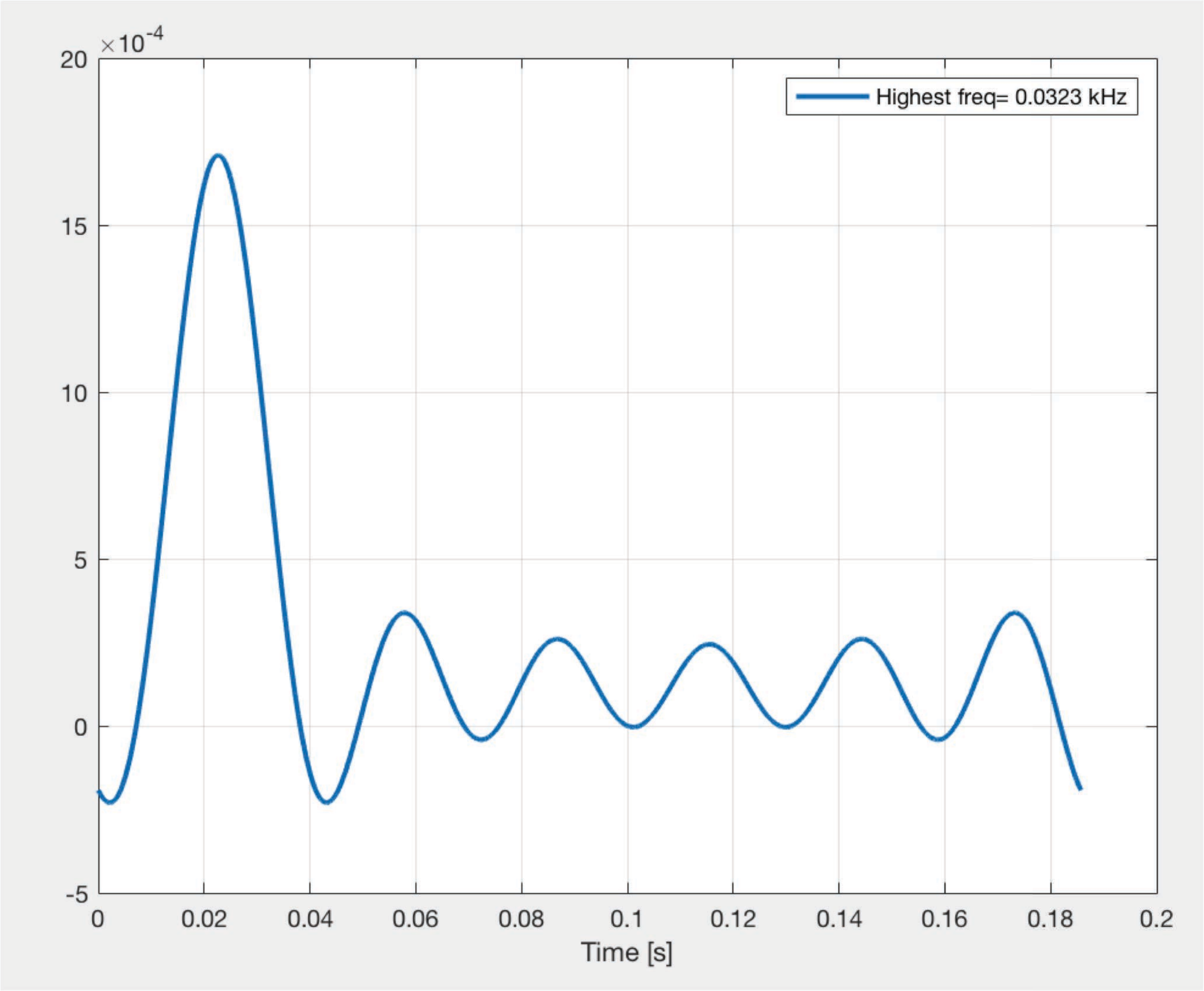
```
% ### EXbuildImpulse2.m ###       11.03.14
clear; clf;
% --------------------------------
SR= 44100;          % sample rate [Hz]
Npoints= 8192;      % length of fft window (# of points) [should ideally be 2^N]
                    % [time window will be the same length]
INDXon= 1000;      % index at which click turns 'on' (i.e., go from 0 to 1)
INDXoff= 1001;    % index at which click turns 'off' (i.e., go from 1 to 0)
% --------------------------------
dt= 1/SR;  % spacing of time steps
freq= [0:Npoints/2];     % create a freq. array (for FFT bin labeling)
freq= SR*freq./Npoints;
t=[0:1/SR:(Npoints-1)/SR]; % create an appropriate array of time points
% build signal
clktemp1= zeros(1,Npoints); clktemp2= ones(1,INDXoff-INDXon);
signal= [clktemp1(1:INDXon-1) clktemp2 clktemp1(INDXoff:end)];
% ------------------------------
% *******
% plot "final" time waveform of signal
if 1==1
    figure(3); clf; plot(t*1000,signal,'ko-','MarkerSize',5)
    grid on; hold on; xlabel('Time [ms]'); ylabel('Signal'); title('Time Waveform')
end
% *******
% now compute/plot FFT of the signal
sigSPEC= rfft(signal);
% MAGNITUDE
figure(1); clf;
subplot(211); plot(freq/1000,db(sigSPEC),'ko-','MarkerSize',3)
hold on; grid on; ylabel('Magnitude [dB]'); title('Spectrum (or "Look Up Table")')
% PHASE
subplot(212); plot(freq/1000,cycs(sigSPEC),'ko-','MarkerSize',3)
xlabel('Frequency [kHz]'); ylabel('Phase [cycles]'); grid on;
% *******
% now make animation of click getting built up, using the info from the FFT
sum= zeros(1,numel(t)); % (initial) array for reconstructed waveform
inclV=[1:30,floor(linspace(31,floor(0.9*numel(freq)),100)),...
    floor(linspace(0.9*numel(freq),numel(freq),20))];
figure(2); clf; grid on;
for nn=1:numel(freq)
    sum= sum+ abs(sigSPEC(nn))*cos(2*pi*freq(nn)*t + angle(sigSPEC(nn)));
    if ismember(nn,inclV),  plot(t,sum,'LineWidth',2); grid on; xlabel('Time [s]');
    legend(['Highest freq= ',num2str(freq(nn)/1000),' kHz'])
    pause(3/(nn));  end
end
```

EXbuildImpulse2.m

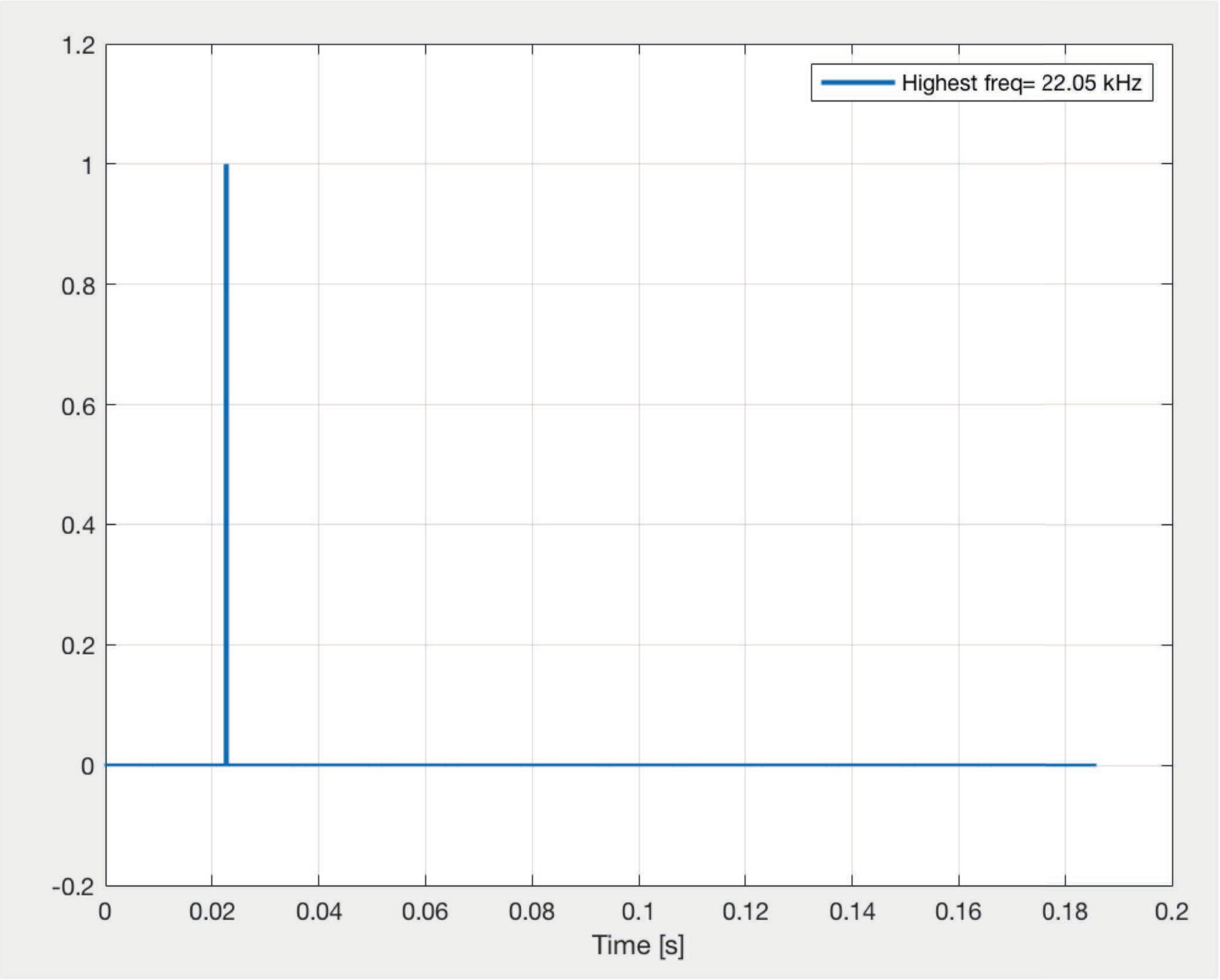


EXbuildImpulse2.m



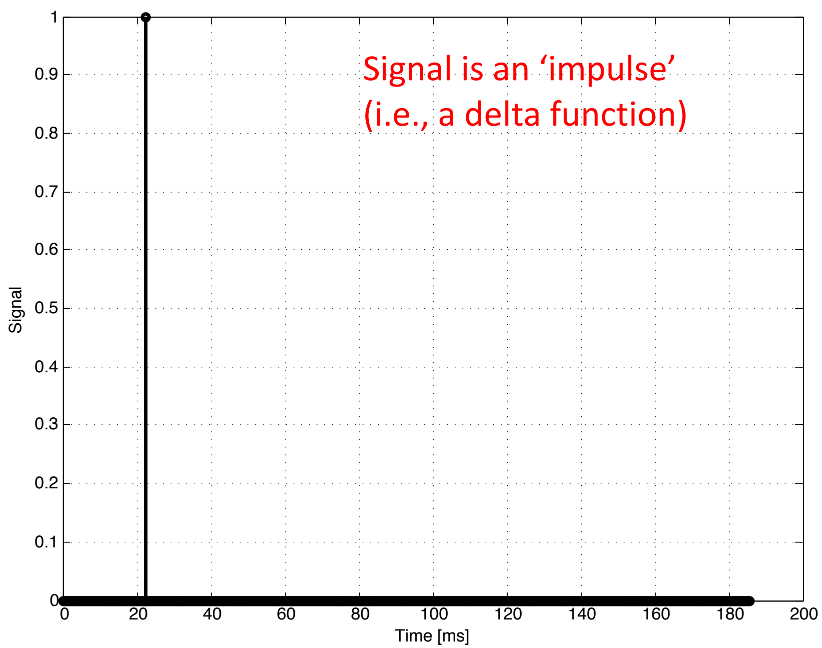


EXbuildImpulse2.m

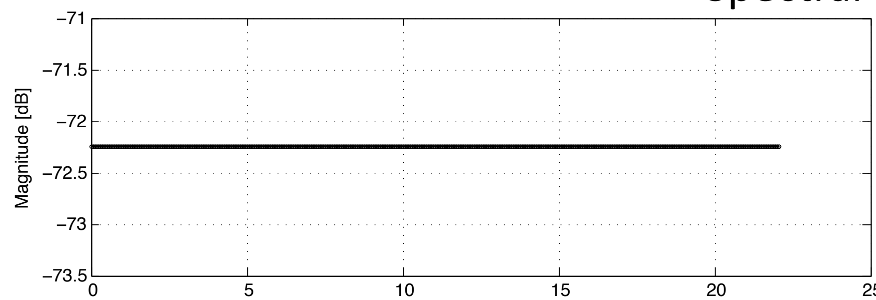


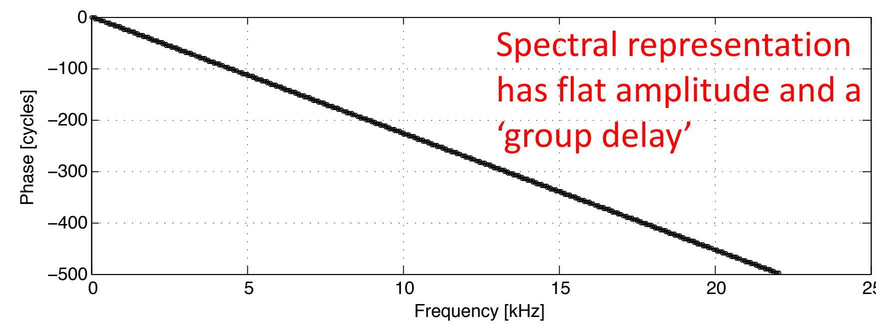
EXbuildImpulse.m

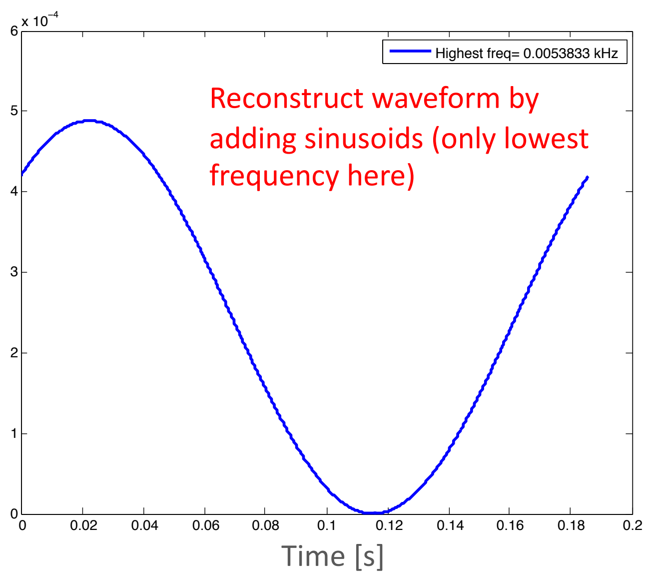
Temporal

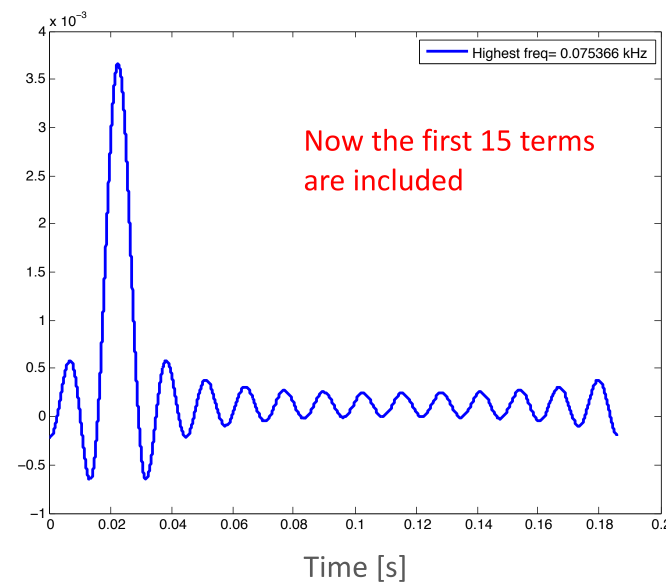


Spectral





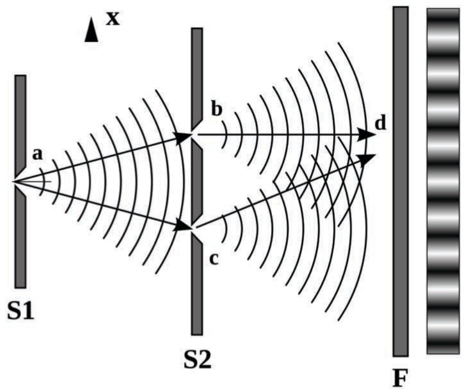


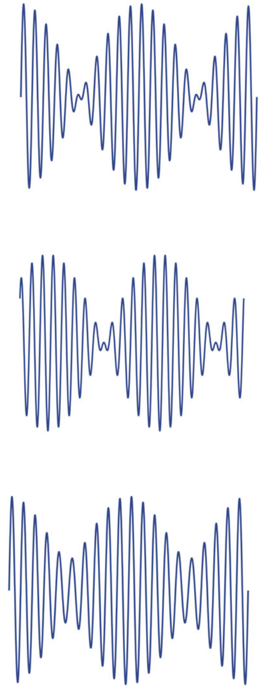


→ Eventually all the sinusoids add up such that things cancel out everywhere except at the point of the impulse!

Summary

Superposition





DDHO is a linear system

$$m\ddot{x}+c\dot{x}+kx=\sum_n (m\ddot{x}_n+c\dot{x}_n+kx_n)=\sum_n F_n(t)=F_{ext}$$

Periodic forcing & superposition

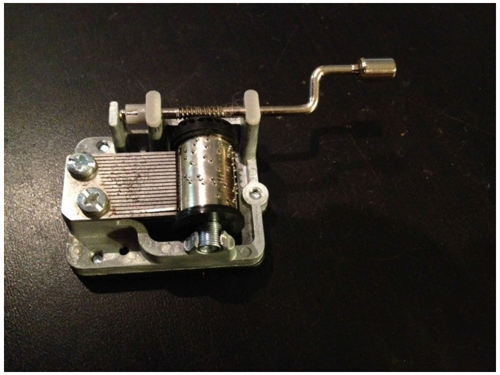
$$F_{ext}=\sum_n F_n(t)$$

Fourier analysis

$$f(t)=\tfrac{1}{2}a_0+\sum_{n=1}^{\infty}[a_n\ \cos(n\omega t)+b_n\ \sin(n\omega t)]$$

$$a_n=\frac{2}{T}\int_{-T/2}^{T/2} f(t)\ \cos(n\omega t)\,dt \qquad n=0,1,2,\ldots$$

$$b_n=\frac{2}{T}\int_{-T/2}^{T/2} f(t)\ \sin(n\omega t)\,dt \qquad n=1,2,\ldots$$



EXspecREP3.m

```
% ### EXspecREP3.m ###       10.29.14
% Example code to just fiddle with basics of discrete FFTs and connections
% back to common real-valued time waveforms
% --> Demonstrates several useful concepts such as 'quantizing' the frequency
% Requires: rfft.m, irfft.m, cycs.m, db.m, cyc.m
% ------
% Stimulus Type Legend
% stimT= 0 - non-quantized sinusoid
% stimT= 1 - quantized sinusoid
% stimT= 2 - one quantized sinusoid, one un-quantized sinusoid
% stimT= 3 - two quantized sinusoids
% stimT= 4 - click I.e., an impulse)
% stimT= 5 - noise (uniform in time)
% stimT= 6 - chirp (flat mag.)
% stimT= 7 - noise (Gaussian; flat spectrum, random phase)
% stimT= 8 - exponentially decaying sinusoid (i.e., HO impulse response)

clear; clf;
% --------------------------------
SR= 44100;          % sample rate [Hz]
Npoints= 8192;      % length of fft window (# of points) [should ideally be 2^N]
                    % [time window will be the same length]
stimT= 8;   % Stimulus Type (see legend above)
f= 2580.0;          % Frequency (for waveforms w/ tones) [Hz]
ratio= 1.22;    % specify f2/f2 ratio (for waveforms w/ two tones)
% Note: Other stimulus parameters can be changed below
% --------------------------------
dt= 1/SR;  % spacing of time steps
freq= [0:Npoints/2];    % create a freq. array (for FFT bin labeling)
freq= SR*freq./Npoints;
% quantize the freq. (so to have an integral # of cycles in time window)
df = SR/Npoints;
fQ= ceil(f/df)*df;   % quantized natural freq.
t=[0:1/SR:(Npoints-1)/SR];  % create an array of time points, Npoints long
% ----
% compute stimulus
if stimT==0 % non-quantized sinusoid
    signal= cos(2*pi*f*t);
    disp(sprintf(' \n *Stimulus* - (non-quantized) sinusoid, f = %g Hz \n', f));
    disp(sprintf('specified freq. = %g Hz', f));
elseif stimT==1     % quantized sinusoid
    signal= cos(2*pi*fQ*t);
    disp(sprintf(' \n *Stimulus* - quantized sinusoid, f = %g Hz \n', fQ));
    disp(sprintf('specified freq. = %g Hz', f));
    disp(sprintf('quantized freq. = %g Hz', fQ));
elseif stimT==2     % one quantized sinusoid, one un-quantized sinusoid
    signal= cos(2*pi*fQ*t) + cos(2*pi*ratio*fQ*t);
    disp(sprintf(' \n *Stimulus* - two sinusoids (one quantized, one not) \n'));
elseif stimT==3     % two quantized sinusoids
    fQ2= ceil(ratio*f/df)*df;
    signal= cos(2*pi*fQ*t) + cos(2*pi*fQ2*t);
    disp(sprintf(' \n *Stimulus* -  two sinusoids (both quantized) \n'));
elseif stimT==4     % click
    CLKon= 1000;     % index at which click turns 'on' (starts at 1)
    CLKoff= 1001;   % index at which click turns 'off'
    clktemp1= zeros(1,Npoints);
    clktemp2= ones(1,CLKoff-CLKon);
    signal= [clktemp1(1:CLKon-1) clktemp2 clktemp1(CLKoff:end)];
    disp(sprintf(' \n *Stimulus* - Click \n'));
elseif stimT==5     % noise (flat)
    signal= rand(1,Npoints);
    disp(sprintf(' \n *Stimulus* - Noise1 \n'));
elseif stimT==6     % chirp (flat)
    f1S= 2000.0;     % if a chirp (stimT=2) starting freq. [Hz] [freq. swept linearly w/ time]
    f1E= 4000.0;   % ending freq.  (energy usually extends twice this far out)
    f1SQ= ceil(f1S/df)*df;      %quantize the start/end freqs. (necessary?)
    f1EQ= ceil(f1E/df)*df;
    % LINEAR sweep rate
    fSWP= f1SQ + (f1EQ-f1SQ)*(SR/Npoints)*t;
    signal = sin(2*pi*fSWP.*t)';
    disp(sprintf(' \n *Stimulus* - Chirp \n'));
elseif stimT==7     % noise (Gaussian)
    Asize=Npoints/2 +1;
    % create array of complex numbers w/ random phase and unit magnitude
    for n=1:Asize
        theta= rand*2*pi;
        N2(n)= exp(i*theta);
    end
    N2=N2';
    % now take the inverse FFT of that using Chris' irfft.m code
    tNoise=irfft(N2);
    % scale it down so #s are between -1 and 1 (i.e. normalize)
    if (abs(min(tNoise)) > max(tNoise))
        tNoise= tNoise/abs(min(tNoise));
    else
        tNoise= tNoise/max(tNoise);
    end
    signal= tNoise;
    disp(sprintf(' \n *Noise* - Gaussian, flat-spectrum \n'));
elseif stimT==8 % exponentially decaying cos
    alpha= 500;
    signal= exp(-alpha*t).*sin(2*pi*fQ*t);
    disp(sprintf(' \n *Exponentially decaying (quantized) sinusoid*  \n'));
end

% ------------------------------
% *******
figure(1); clf      % plot time waveform of signal
plot(t*1000,signal,'k.-','MarkerSize',5); grid on; hold on;
xlabel('Time [ms]'); ylabel('Signal'); title('Time Waveform')
% *******
% now plot rfft of the signal
% NOTE: rfft just takes 1/2 of fft.m output and nomalizes
sigSPEC= rfft(signal);
figure(2); clf; % MAGNITUDE
subplot(211)
plot(freq/1000,db(sigSPEC),'ko-','MarkerSize',3)
hold on; grid on;
ylabel('Magnitude [dB]')
title('Spectrum')
subplot(212)    % PHASE
plot(freq/1000,cycs(sigSPEC),'ko-','MarkerSize',3)
xlabel('Frequency [kHz]'); ylabel('Phase [cycles]'); grid on;
% -------
% play the stimuli as an output sound?
if (1==1),  sound(signal,SR);   end
% -------
% compute inverse Fourier transform and plot?
if 1==1
    figure(1);
    signalINV= irfft(sigSPEC);
    plot(t*1000,signalINV,'rx','MarkerSize',4)
    legend('Original waveform','Inverse transformed')
end
```

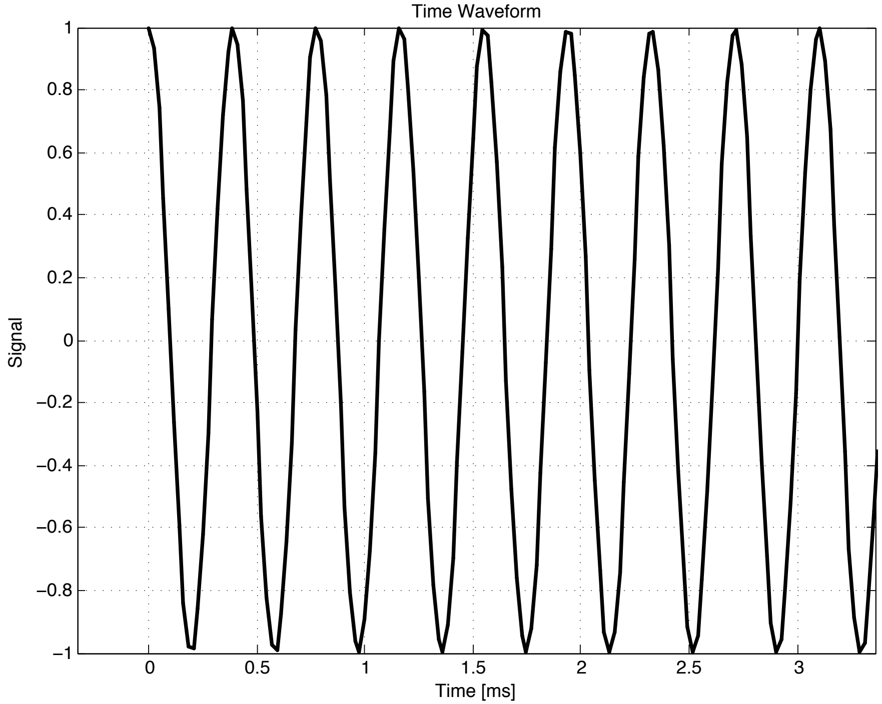
Fourier transforms of basic (1-D) waveforms

EXspecREP3.m

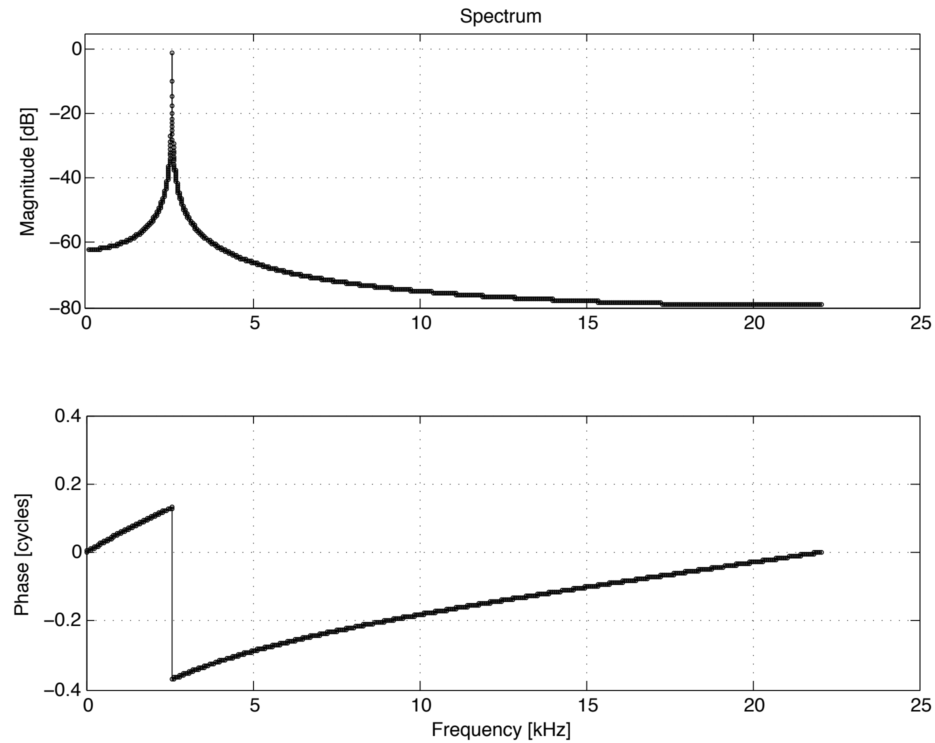
```
stimT= 0 - non-quantized sinusoid
```

```
SR= 44100;  % sample rate [Hz]
Npoints= 8192; % length of fft window
```

Time domain



Spectral domain



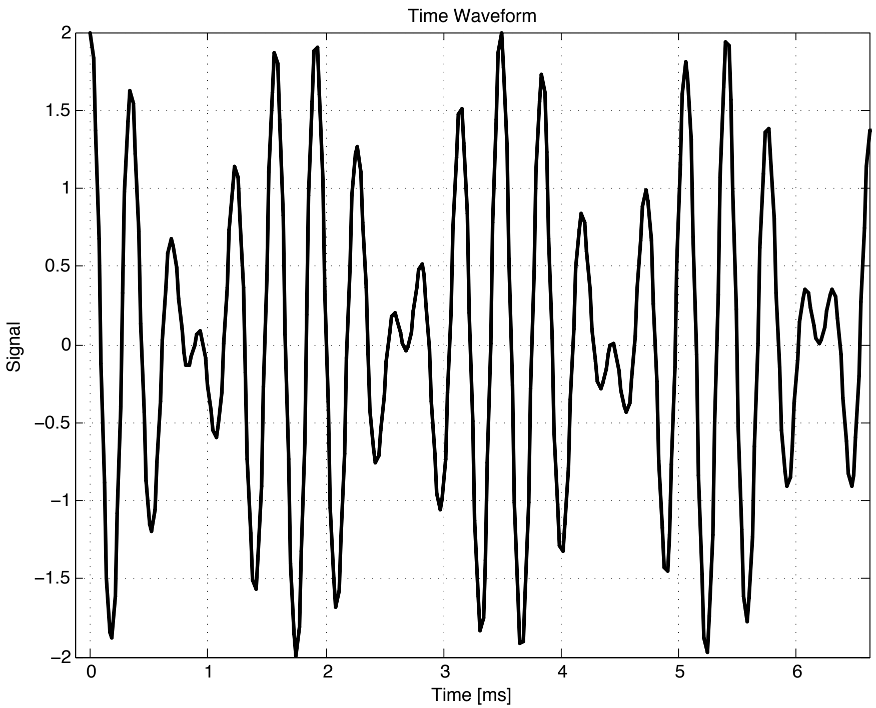
Note: The phase is 'unwrapped' in all the spectral plots

- Magnitude shows a peak at the sinusoid's frequency

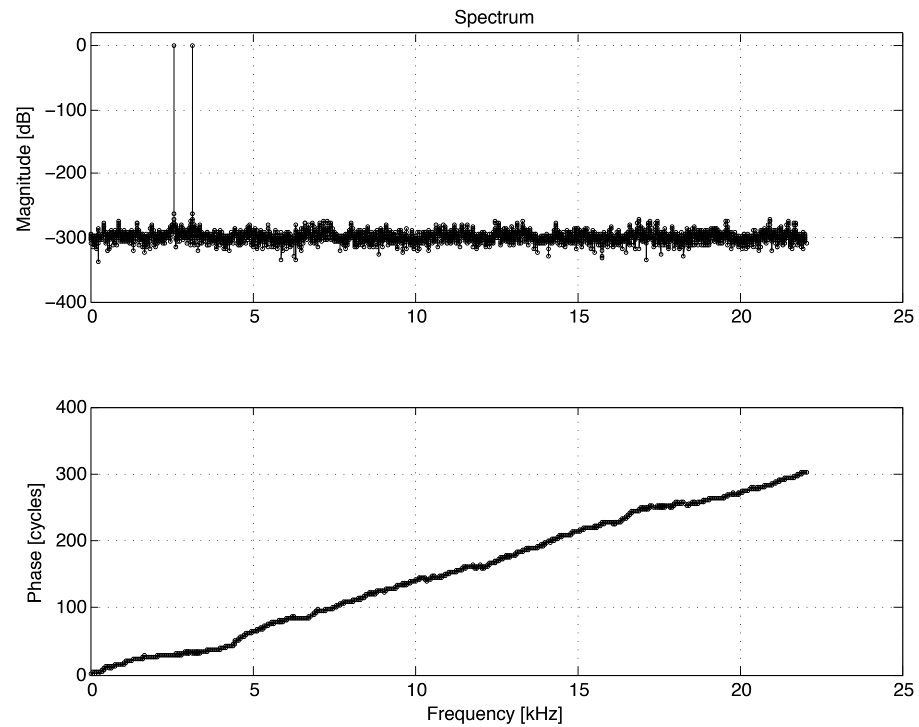
Fourier transforms of basic (1-D) waveforms

stimT= 3 – two quantized sinusoids

Time domain



Spectral domain

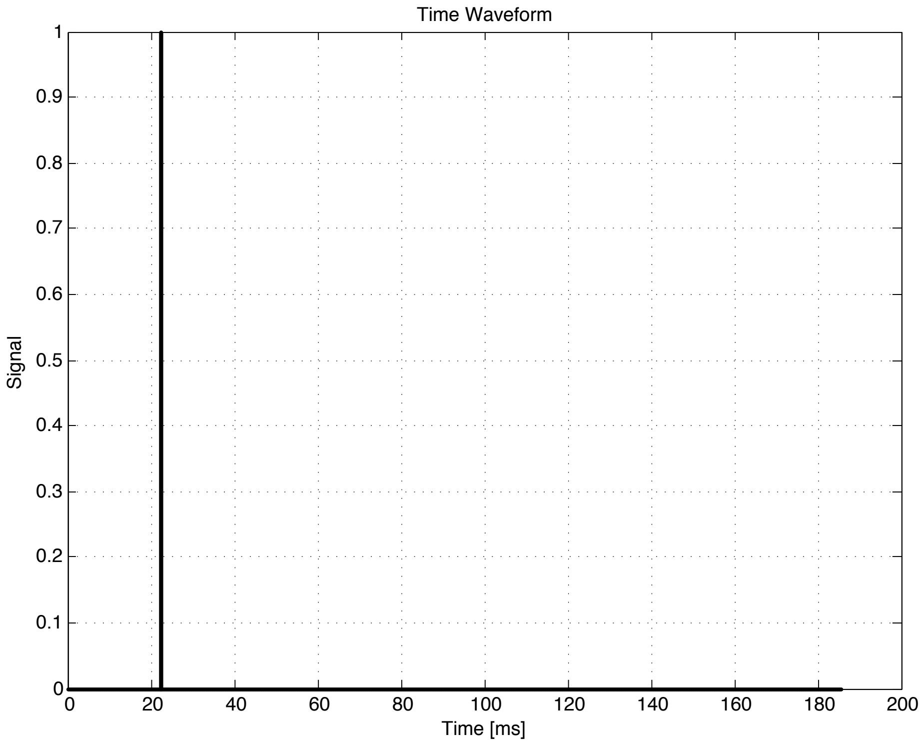


- Magnitude shows two peaks (note the 'beating' in the time domain)

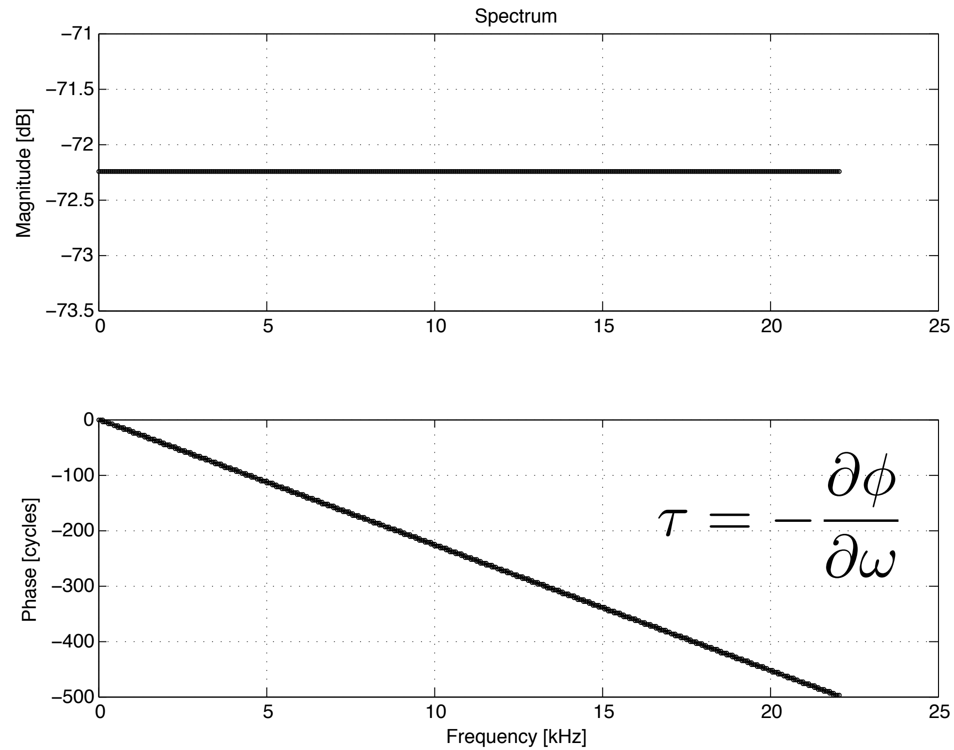
Fourier transforms of basic (1-D) waveforms

```
stimT= 4 - click (i.e., an impulse)
```

Time domain



Spectral domain



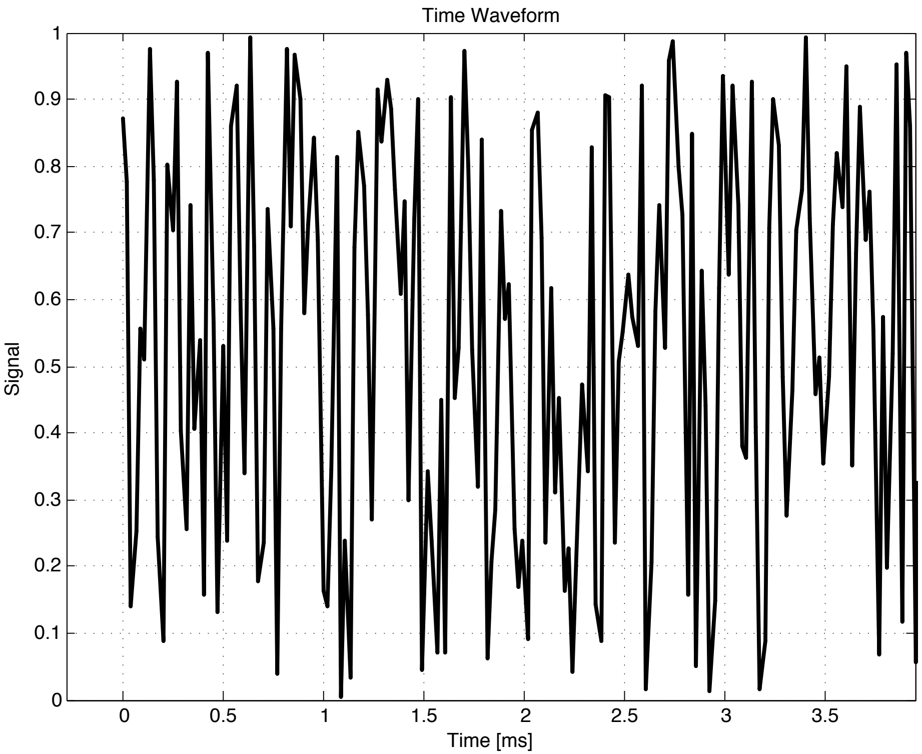
$$\tau = -\frac{\partial \phi}{\partial \omega}$$

- Click has a flat magnitude (This is also a good place to mention the concept of a '*group delay*')

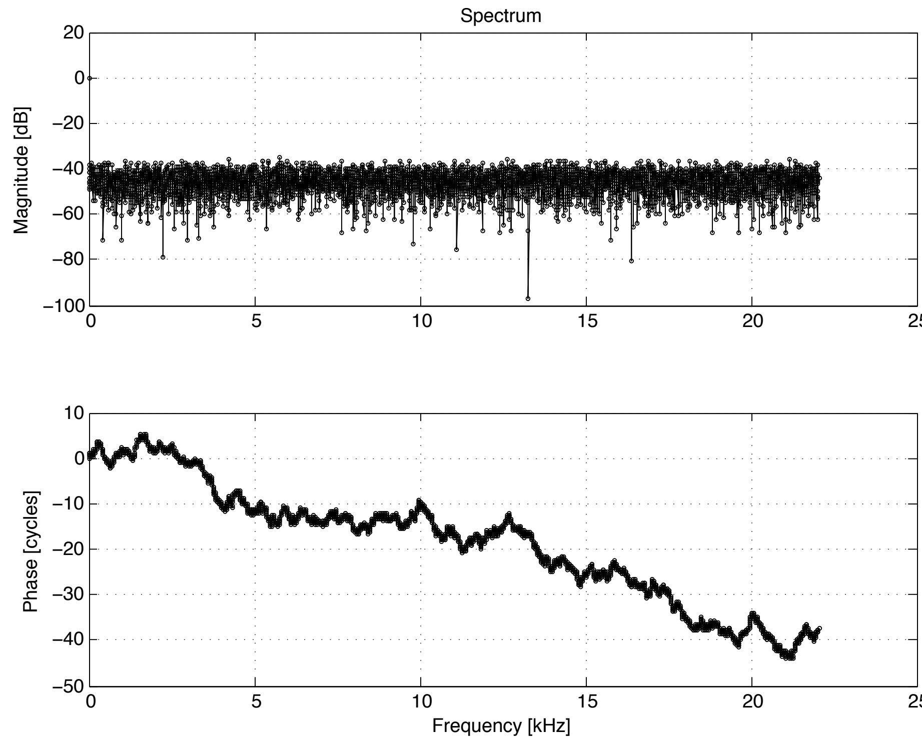
Fourier transforms of basic (1-D) waveforms

```
stimT= 5 - noise (uniform distribution)
```

Time domain



Spectral domain



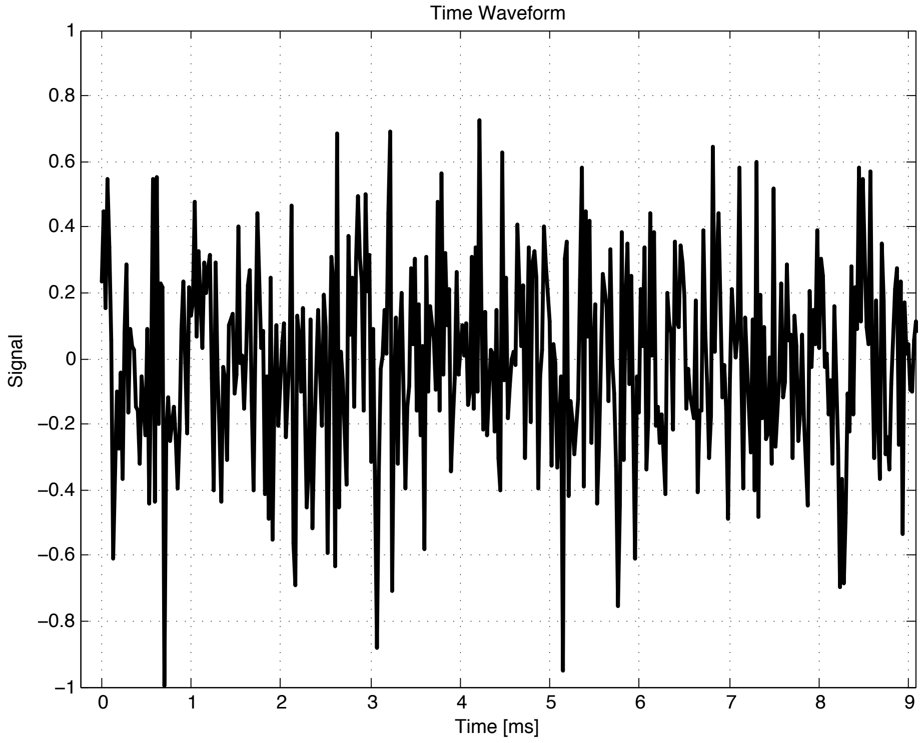
➢ Magnitude is flat-ish (on log scale), but actually noisy. Phase is noisy too.

Fourier transforms of basic (1-D) waveforms

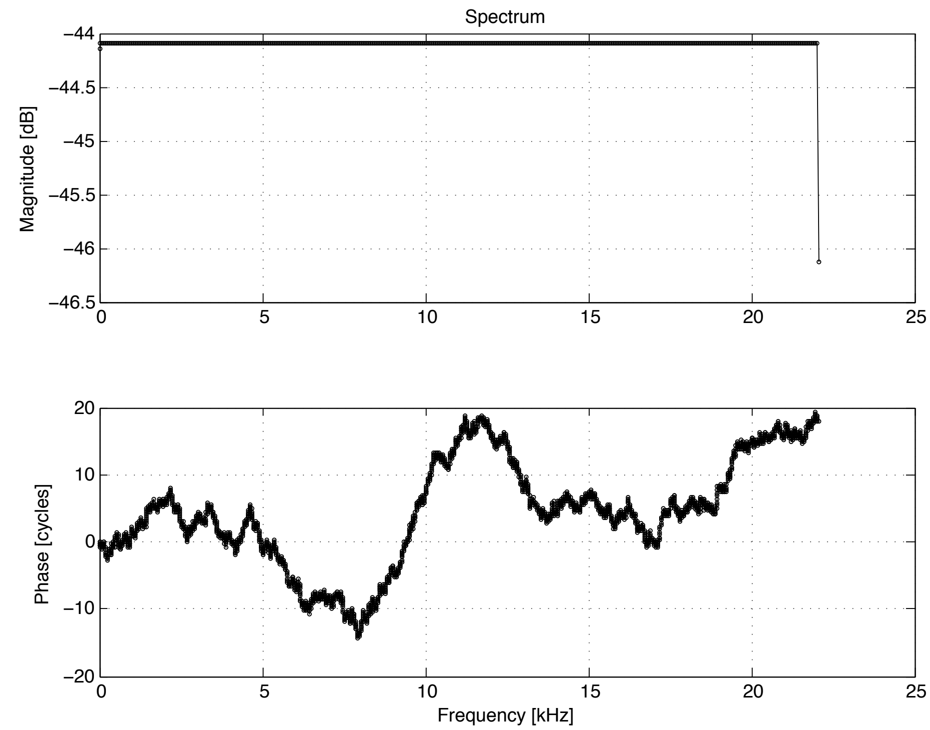
EXspecREP3.m

```
stimT= 7 - noise (Gaussian distribution)
```

Time domain



Spectral domain



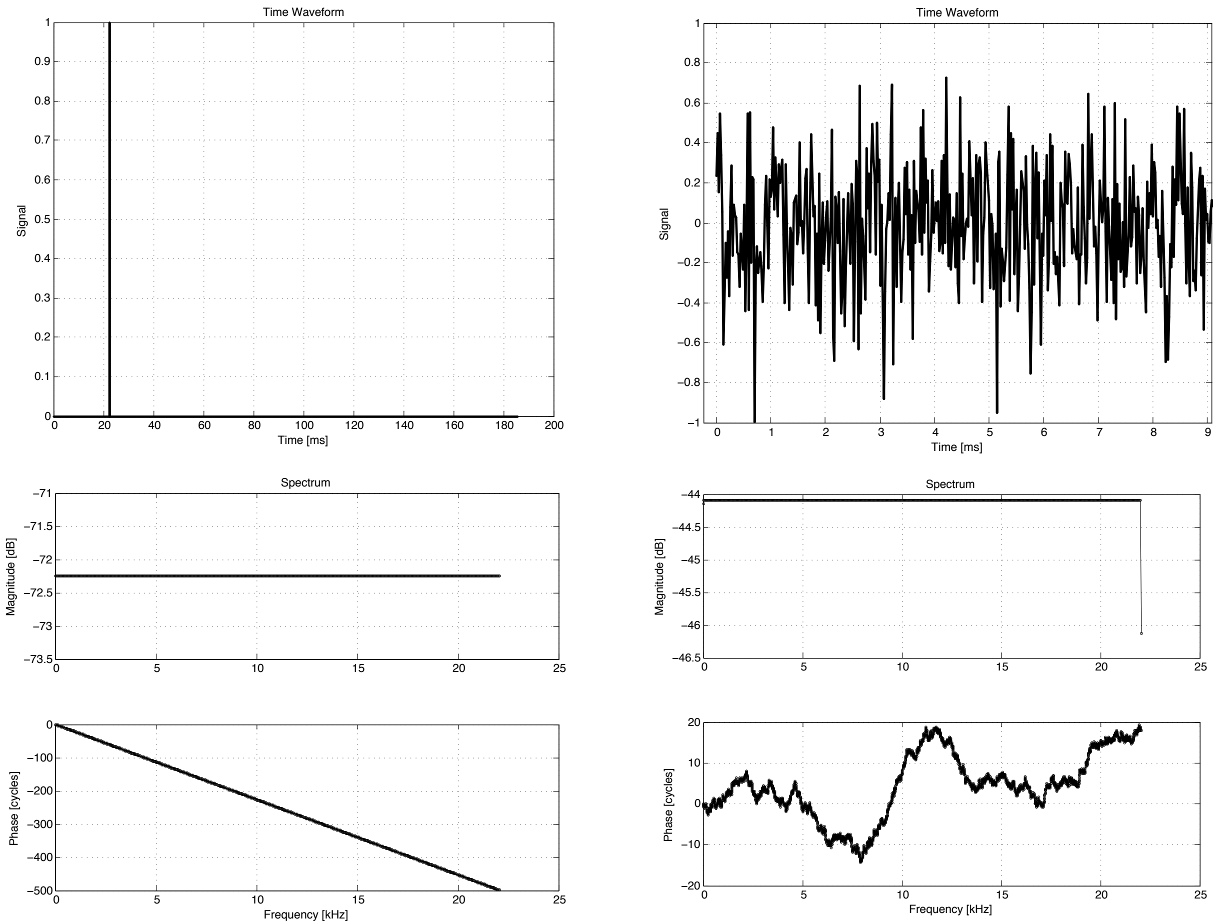
- Magnitude is flat just like an impulse (i.e., flat), but the phase is random

Fourier transforms of basic (1-D) waveforms

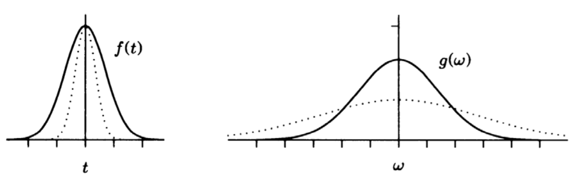
Impulse | Noise

Time domain

Spectral domain



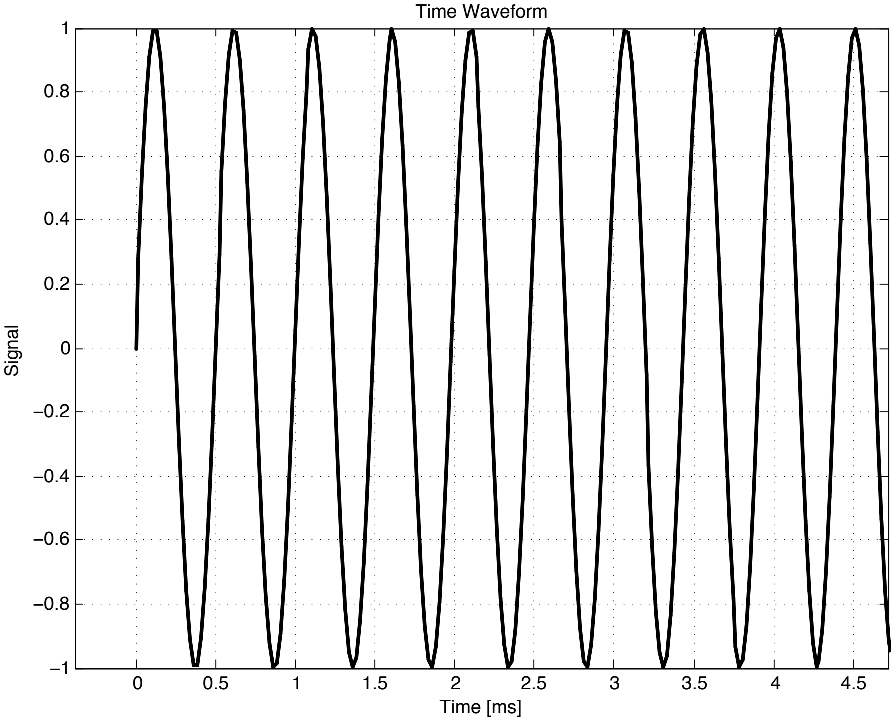
→ Remarkable that the magnitudes are identical (more or less) between two signals with such different properties. The key difference here is the phase: *Timing is a critical piece of the puzzle!*



Fourier transforms of basic (1-D) waveforms

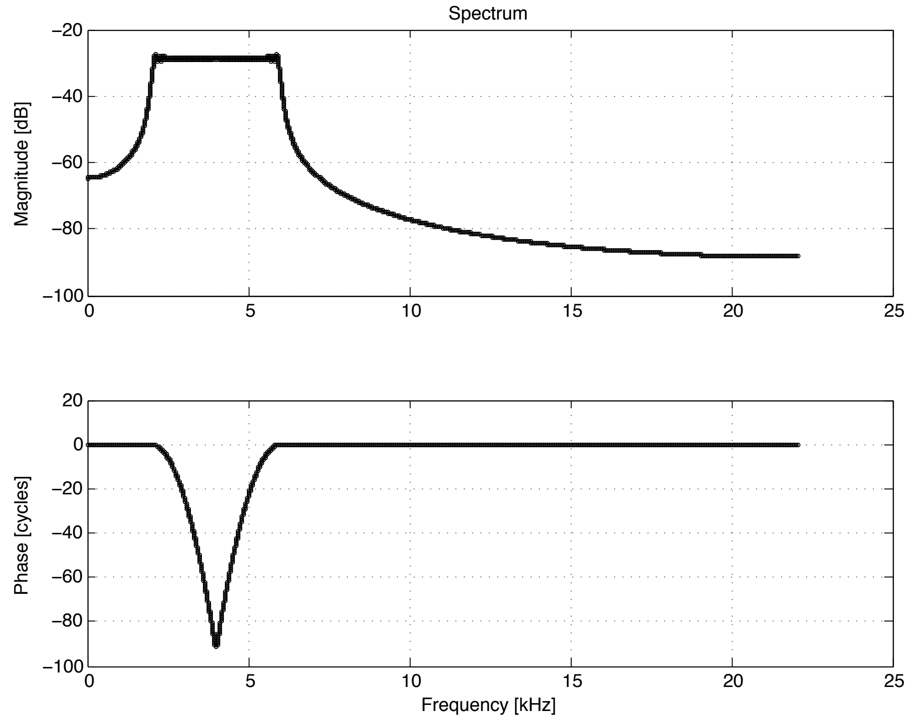
```
stimT= 6 - chirp (flat mag.)
```

Time domain



Hard to see on this timescale, but frequency is changing (increasing) with time

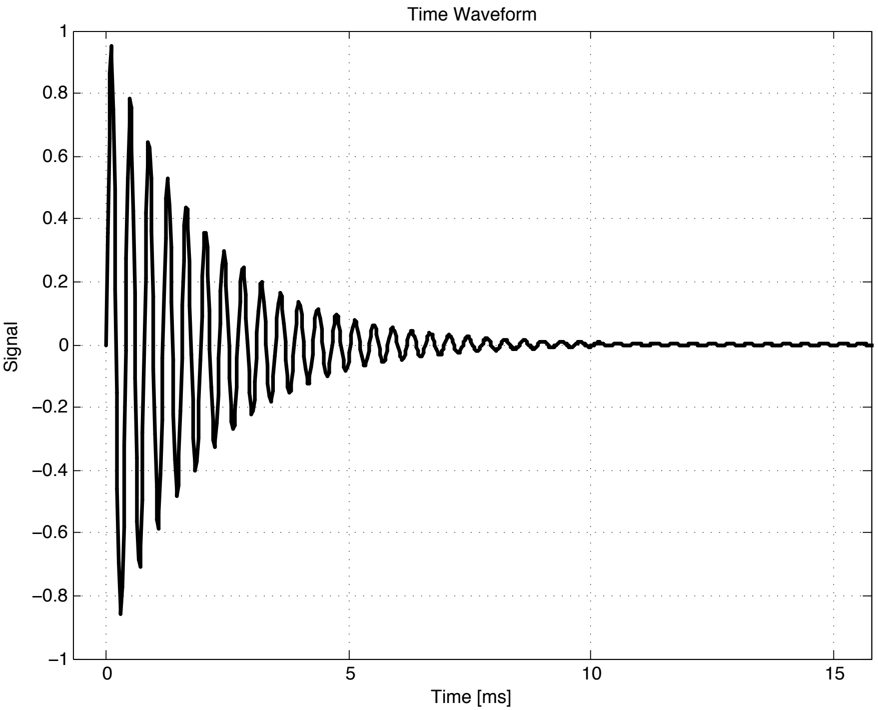
Spectral domain



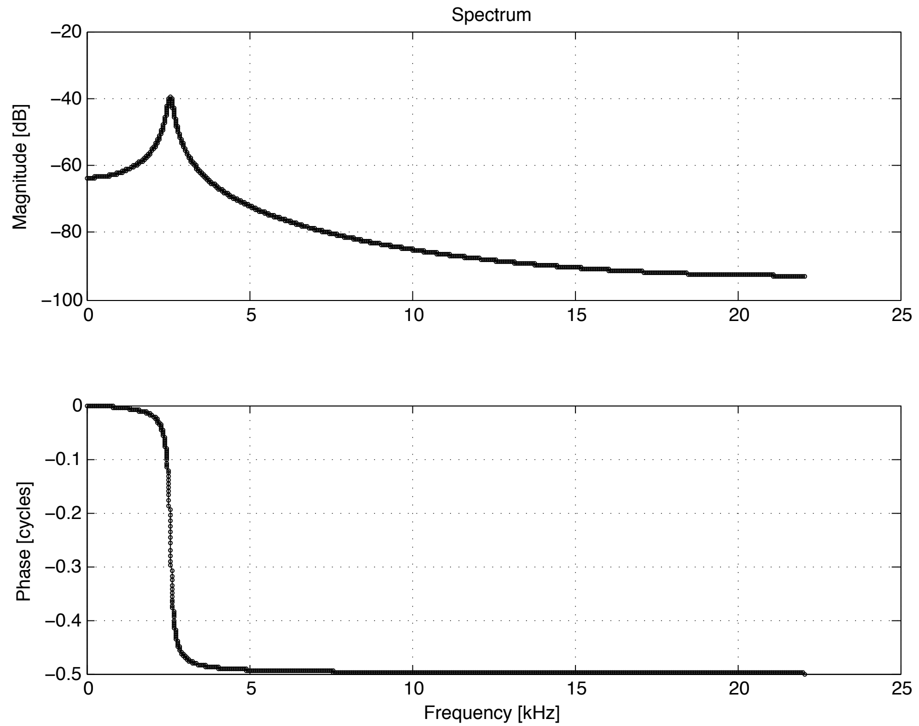
Fourier transforms of basic (1-D) waveforms

`stimT= 8 - exponentially decaying sinusoid`

Time domain



Spectral domain



- This seems to look familiar....