% In-Lecture Assignment #1 on Feb. 5, 2025. Based on homework problem 1.2**.**

***% Key takeaways: (1) Chirp signals are useful in localization, testing and training***

***% because they linearly sweep a range of frequencies, and (2) Spectrograms analyze***

***% a signal in the time and frequency domains simultaneously so that frequencies can***

***% be localized in time. Spectrogram trades off frequency resolution for time resolution.***

% **Chirp Signals**: Please see slides 1-14 to 1-16 of [CommonSignalsInMatlab.pptx](http://users.ece.utexas.edu/~bevans/courses/realtime/handouts/CommonSignalsInMatlab.pptx).

% **Spectrograms**: Please see slides 1-17 to 1-20 of [CommonSignalsInMatlab.pptx](http://users.ece.utexas.edu/~bevans/courses/realtime/handouts/CommonSignalsInMatlab.pptx).

% **Introduction**: A chirp signal is a sinusoid whose principal frequency

% increases (or decreases) over time.  A chirp signal has the form

% c(t) = cos( q(t) )  where q(t) = 2 p ( f0 + 0.5 fstep t ) t = 2 p f0 t + p fstep t2

% The principal frequency in Hz is f0 when t = 0 and then changes over time at a

% rate of fstep in units of Hz/s. The principal frequency of a sinusoid at a given

% point in time is called the instantaneous frequency, and it is defined as

% dq (t) / dt in units of rad/s.  dq (t) / dt = 2 p f0 + 2 p fstep t = 2 p (f0 + fstep t).

% We divide dq (t) / dt by 2p to obtain instantaneous frequency in Hz of f0 + fstep t.

% **(a)** **Generate a chirp** signal that lasts 10s with f0 = 20 Hz and fstep = 420 Hz/s.

% Use sampling rate fsof 44100 Hz.  The chirp will sweep through the principal

% frequencies of the keys on an 88-key piano. Here’s Matlab code to get started.

%%% Generate a chirp signal with frequency increasing

%%% from f0 to (f0 + fstep time) over time seconds

time = 10;

f0 = 20;

fstep = 420;

fs = 44100;

Ts = 1 / fs;

t = 0 : Ts : time;

%%% Add code here to define the chirp signal y = cos( angle(t) )

angle = 2\*pi\*f0\*t + pi\*fstep\*t.^2;

y = cos(angle);

% **(b) Play the chirp signal** as an audio signal. Describe what you hear.

% *I hear a rising pitch over time. Sounds like a slide whistle or a tsunami warning siren*

*%* ([rb.gy/18exl](https://rb.gy/18exl)). *Note: Some laptop playback systems cannot play frequencies below 200 Hz.*

sound(y, fs);

pause(time+1);

**% (c) Plot the spectrogram** of the chirp signal and describe the visual representation.

*% Spectrogram shows a yellow line that represents the principal frequency in the chirp*

*% signal. The line goes from 20 Hz at time 0s to 4220 Hz at time 10s. The spectrogram*

*% plot is on the next page. See Appendix A for explanation of spectrogram arguments.*

figure;

blockSize = 256; overlap = 128;

spectrogram(y, hamming(blockSize), overlap, blockSize, fs, 'yaxis');  
title('(c) Spectrogram with block size 256 and overlap 128');

% **(d) Give the code** for the spectrogram that would improve the

% frequency resolution by a factor of two vs. part (c).

*% The frequency resolution is what is possible from observing a signal for a block of*

*% N samples which lasts for N Ts seconds.  From homework problem 0.1, the frequency*

*% frequency resolution in Hz is the inverse of the observation time or 1 / (N Ts) = fs / N.*

% *Increase N to decrease (improve) frequency resolution.*

*% The yellow line in the spectrogram with N doubled is half the width vs. part (c).*

*% Please see the derivation of frequency resolution in Appendix B.*

figure;

blockSize = 2\*256; overlap = 128;

spectrogram(y, hamming(blockSize), overlap, blockSize, fs, 'yaxis');  
title('(d) Spectrogram with block size 512 and overlap 256');

% **(e) Give the code** for the spectrogram that would improve the time resolution,

% *i.e. localizing frequency components in time,* by a factor of two *vs. part (c).*

*% The time resolution means the ability to identify when a frequency component occurs*

*% in time.  In a block of N samples, we do not know when frequency components occur, and*

*% hence, our time resolution in seconds is N Ts. We improve time resolution by reducing N.*

figure;

blockSize = 256/2; overlap = blockSize/2;

spectrogram(y, hamming(blockSize), overlap, blockSize, fs, 'yaxis');  
title('(e) Spectrogram with block size 128 and overlap 64');

A graph of a number of colored lines

Description automatically generated with medium confidenceA graph of a line graph

Description automatically generated with medium confidenceA graph of a number of numbers

Description automatically generated with medium confidence

(e)

(d)

In all three spectrogram plots, the extent of the horizontal time axis is the same (from 0 to 10s) and the extent of the vertical frequency axis is the same (from 0 to ½ *fs* where *fs* = 44100 Hz). We have chosen *fs* to satisfy the sampling theorem *fs* > 2 *fmax* where *fmax* is the maximum frequency of interest (4220 Hz) and to be a standard audio sampling rate.

(c)

**Appendix A: Arguments to the MATLAB spectrogram function by Dan Jacobellis**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Appendix B: Derivation of Frequency Resolution**

Frequency resolution of Hz means two frequency components spaced Hz apart can each be clearly identified by an algorithm, e.g. well separated in a plot of the frequency domain.

We’ll illustrate the concept of frequency resolution by revisiting homework problem 0.1.

Homework 0.1 concerned a sine signal *c*(*t*) lasting from 0s to 1s. The mathematical expression is a two-sided sine signal multiplied by a rectangular pulse that lasts from 0s to 1s:

*c*(*t*) = sin(2 π *fc* *t*) rect(*t –* ½)

The continuous-time Fourier transform of *r*(*t*) = rect(*t –* ½) is a sinc function times a phase shift

where and

due to the modulation property.

Below are the plots of on the left and  for Hz on the right:

Chart, line chart, histogram

Description automatically generatedChart, line chart, histogram

Description automatically generated

For a signal lasting 0s to 1s and containing sinusoids at frequencies 3 Hz and 4 Hz,

*c2*(*t*) = sin(2 π *f0* *t*) rect(*t –* ½) + sin(2 π *f1* *t*) rect(*t –* ½)

let’s see if we can resolve the two frequencies. We’re looking for two peaks in the frequency domain plot that are well separated at 3 Hz and 4 Hz. Between the peaks, the magnitude response should not be higher than the “sidelobes” at frequencies higher than 1 Hz in .

Chart, histogram

Description automatically generatedChart, line chart

Description automatically generated

Clean separation of 3 Hz and  
4 Hz frequency components

Difficulty separating 3.2 Hz and 4 Hz frequency components

More generally, for a rectangular pulse of duration *T* seconds, the frequency resolution is 1/*T*. The value of 1/*T* is also the null bandwidth.

In the course of computing the spectrogram, we apply a rectangular pulse to the discrete-time signal to extract a block of samples to compute their Fourier series coefficients using the fast Fourier transform. Consider a discrete-time signal that is a two-sided sine signal and the first *N* samples are kept:

*c*[*n*] = sin(*wc* *n*) rect((*n* – *N*/2)/*N*)

Here, *r*[*n*] = rect((*n* – *N*/2)/*N*) which has amplitude 1 for and 0 elsewhere. We can also write *r*[*n*] = *u*[*n*] – *u*[*n*-*N*] where *u*[*n*] is the unit step function. The discrete-time Fourier transform of *r*[*n*] is a periodic sinc function times a phase shift :

where w is in units of rad/sample. The periodic sinc function is periodic in w with period 2p. Here’s one period of for *N* = 8 :

Diagram

Description automatically generated

w = -pi : 0.001 : pi;

N = 8;

Rw = sin(N\*w/2) ./ sin(w/2);

figure;

plot(w, abs(Rw), 'k','LineWidth',2 );

title( '|R(w)|' );

xlabel( 'w' );

ylim( [0, 9] );

This is the magnitude response of an averaging filter with 8 coefficients. Please see the [Designing Averaging Filters handout](http://users.ece.utexas.edu/~bevans/courses/realtime/lectures/01_Sinusoids/DesigningAveragingFilters.pdf).

The first zero for the magnitude response in positive frequencies occurs at 2p / *N*. This is the null bandwidth and also the frequency resolution .

Let’s connect the frequency resolution in the discrete-time frequency domain to the continuous-time frequency domain:

means that