Beat Notes and Time-Varying Signals

1 Introduction

The objective of this lab assignment is to introduce more complicated signals that are related to the basic sinusoid. These signals such as frequency modulation (FM) and amplitude modulation (AM) are widely used in communication systems such as radio and television, but they also can be used to create interesting sounds that mimic musical instruments.

Pre-Lab:

Read this assignment sheet and sections 3.2 and 3.8 in the textbook. Do problem 3.4 on beat notes from the text before lab (this problem is also part of HW3, so it needs to be one way or another!). Your lab TA will begin by reviewing your work.

2. Background

More complicated signals can be obtained by summing sinusoids together (such as Fourier Series), changing the amplitude of the signal with time (Amplitude Modulation) or changing the frequency of the sinusoid with time (Frequency modulation). This lab investigates the sum of two closely spaced sinusoids and frequency modulation.

2.1 Sinusoidal Multiplication or Amplitude Modulation

When two sinusoids with different frequencies are multiplied together, an audio effect called a beat note occurs. Another use for this phenomenon is Amplitude Modulation (AM) which is used for radio broadcasting where the audio signal is multiplied by a high frequency carrier wave.

$$x(t) = \cos(2\pi f_1 t)\cos(2\pi f_2 t)$$

If the Inverse Euler Formula is applied, then this equation can be simplified as follows:

$$x(t) = \left(\frac{e^{j2\pi f_1 t} + e^{-j2\pi f_1 t}}{2}\right)\left(\frac{e^{j2\pi f_2 t} + e^{-j2\pi f_2 t}}{2}\right) = \frac{1}{4}\left(e^{j2\pi(f_1+f_2)t} + e^{-j2\pi(f_1+f_2)t} + e^{j2\pi(f_1-f_2)t} + e^{-j2\pi(f_1-f_2)t}\right)$$

$$= \frac{1}{2}\cos\left(2\pi\left(f_1+f_2\right)t\right) + \frac{1}{2}\cos\left(2\pi\left(f_1-f_2\right)t\right)$$

Note that neither of the two frequencies is in the spectrum of the signal. The choice of *frequencies* determines the nature of the signal. For amplitude modulation or beat signals, we pick two frequencies close together to get a pronounced warble.

$$x(t) = A\cos\left(2\pi\left(f_c - \Delta\right)t\right) + B\cos\left(2\pi\left(f_c + \Delta\right)t\right) .$$

2.2 Chirp or Linearly Swept Signal

Many interesting signals can be produced by changing the argument of a generalized sinusoid as different functions of time. This is called *fm synthesis* or Frequency Modulation and it is used to create instrument simulations and interesting sound effects. A *chirp* signal is a sinusoid whose frequency changes linearly from a starting frequency to an ending one. In other words,

$$x(t) = A\cos\left(\psi(t)\right)$$

A “standard” sinusoid has
$\psi(t) = \omega t + \phi$

However, much more interesting signals can be created with different functions such as a quadratic, exponential , or sinusoid as shown below.

$$\psi(t) = 2\pi\mu t^2 + 2\pi f_0 t + \phi$$
$$\psi(t) = e^{at} + 2\pi f_0 t + \phi$$
$$\psi(t) = \cos(2\pi f_1 t) + 2\pi f_0 t + \phi$$

The frequency spectra of the signals is hard to analyze, however a decent approximation of signal behavior can be used by looking at the first derivative of the argument function. This is known as the *instantaneous frequency* of the signal.

The instantaneous frequency of the quadratic phase signal is:

$$d\psi(t)/dt = f_i = 2\pi(2\mu t + f_0)$$
$$f_i = ae^{at} + 2\pi f_0$$
$$f_i = -2\pi f_1 \sin(2\pi f_1 t) + 2\pi f_0$$

Note that for the quadratic signal, the instantaneous signal changes linearly versus time, with a slope of 2 μ and an initial frequency of f_0 if t starts at t=0. This signal produces a sound similar to a siren or a chirp.

Note: Problem 3.17 gives more detail on the linear chirp signal.

2.3 Spectrograms

In addition to looking at signals in the time domain, it is often useful to look at the spectrum of a signal. A signal’s spectrum shows which frequencies are present in the signal. As shown in class, a constant frequency sinusoid spectrum consists of two impulse functions at $2\pi f_0$ and -$2\pi f_0$. For more complicated signals, there may be many spikes. For more complicated signals such as music or frequency modulation, the spectrum changes with time. In these cases, the spectrogram of a signal is used instead of a spectrum. A spectrogram is found by estimating the spectrum over multiple short windows of time. Section 3.7 describes this process in more depth. The magnitude of the spectrum over these time “windows” is plotted as intensity or color on a two dimensional plot with time on one axis and frequency on the other.

In Matlab, the function *spectrogram* will be used to compute the spectrogram. The spectrogram function divides a signal into segments. Longer segments provide better frequency resolution; shorter segments provide better time resolution. For more information, see http://www.mathworks.com/help/signal/examples/practical-introduction-to-time-frequency-analysis.html. There are theoretical limits on how well short pieces of the signal can represent the frequency content in a signal. Generally, longer time windows give better frequency resolution. We will investigate this more later in the class. The *spectrogram* function can be called as follows: spectrogram(xx, 1024, 512, [], Fs). The first argument, *xx*, is the time signal, 1024 is the number of samples in the time window, 512 is the number of samples of overlap between the time windows, and Fs is the sampling frequency. **NOTE: the spectrogram function requires the Matlab signal processing toolbox.**

In order to see what the spectrogram function does, run the following code.

```
N = 1024;
n = (0:N−1);
```

```
w0 = 2*pi/5;
x = sin(w0*n)+10*sin(2*w0*n);
spectrogram(x,128,64,[],100);
```

Lab Assignment:

Part 1: Beat Notes

Beat notes provide an interesting way to investigate the time-frequency characteristics of spectrograms. Although some of the mathematical details are beyond the reach of this course, it is not difficult to appreciate the following issue: there is a fundamental trade-off between knowing which frequencies are present in a signal (or its spectrum) and knowing how those frequencies vary with time. As mentioned previously in Section 3.3, a spectrogram estimates the frequency content over short sections of the signal. If we make the section length very short we can track rapid changes in the frequency. However, shorter sections lack the ability to do accurate frequency measurement because the amount of input data is limited. On the other hand, long sections can give excellent frequency measurements, but fail to track frequency changes well. For example, if a signal is the sum of two sinusoids whose frequencies are nearly the same, a long section length is needed to "resolve" the two sinusoidal components. This trade-off between the section length (in time) and frequency resolution is equivalent to Heisenburg's Uncertainty Principle in physics. More discussion of

A beat note may be viewed as a single frequency sinusoid whose amplitude varies with time or as two signals with different constant frequencies. The equation for the beat signal is:

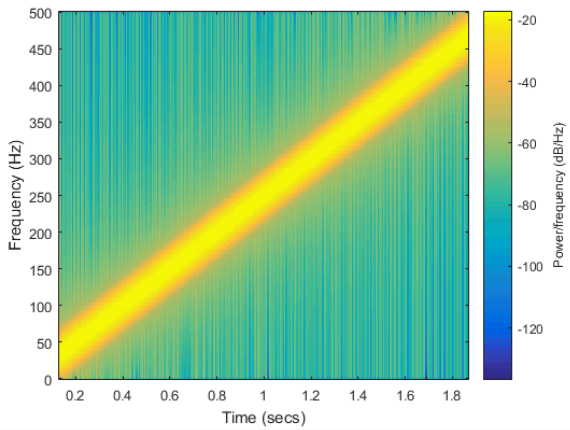
$$x(t) = A\cos\left(2\pi\left(f_c - \Delta\right)t\right) + A\cos\left(2\pi\left(f_c + \Delta\right)t\right)$$
$$= 2A\cos\left(2\pi\Delta t\right)\cos\left(2\pi f_c t\right)$$

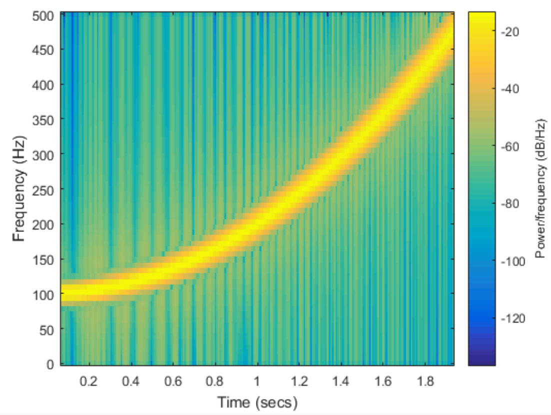
a. Create and plot a beat signal with Δ = 40 Hz; a time duration, T_{dur}= 0.5 sec, sampling frequency, Fs, of 11025 Hz, and an f_c=1200 Hz. Listen to the signal using the command *soundsc*. Comment on what you hear. Plot 0.5 seconds of the signal.

b. Find the spectrogram using a window length of 2048 using the commands:
spectrogram(x,2048,1024,[],Fs);
colormap(1-gray(256)); (makes figure grayscale for printing)
Comment on what you see. Are the correct frequencies present in the spectrogram? If necessary, use the zoom tool to locate the frequencies.

Part 2: Investigating frequency modulation (FM) signals:

A chirp signal is also known as swept frequency cosine. The frequency of the cosine changes in time. The frequency can change in many different ways as seen in the spectrograms below for linear and quadratic sweeps. We will focus on a linear sweep where the instantaneous frequency is linear as shown in the left hand plot. The plot on the right shows the case where the instantaneous frequency is a quadratic.





To generate a chirp signal, we start with a cosine signal with constant frequency:

$x(t) = A\cos(2\pi f_0 t + \phi)$

To get a linearly swept frequency signal, the cosine frequency must change linearly with time. If the cosine argument is generalized to the time function, $x(t) = A\cos(\psi(t))$, this means that the derivative of the cosine's argument must be a linear function. Defining

$\psi(t) = 2\pi\mu t^2 + 2\pi f_0 t + \phi$

will give a linear function as the time derivative:

$\frac{d\psi(t)}{dt} = 2\pi 2\mu t + 2\pi f_0$ in radians/sec or $\frac{1}{2\pi}\frac{d\psi(t)}{dt} = \frac{2\pi 2\mu t + 2\pi f_0}{2\pi} = 2\mu t + f_0$.

This gives a frequency that changes linearly with time.

The following *Matlab* code will synthesize a linear sweep chirp signal:

```
fsamp =11025; % set sampling frequency
dt =1/fsamp; % set sampling interval in seconds
dur = 1.5;%set signal duration in seconds
tt=0:dt:dur; % create vector of time samples spaced at dt seconds
psi =2*pi* (500*tt.^2+200*tt +100); % set argument for chirp function
xx=10*cos(psi); % modulate signal with cosine and amplitude cos
soundsc(xx,fsamp);% play signal
specgram(xx,1024,fsamp);% plot spectrogram
```

(b) In MATLAB signals can only be synthesized by evaluating the signal's defining formula at discrete instants of time. These are called *samples* of the signal. For the chirp we do the following:

$$x(t_n) = A\cos(2\pi\mu t_n^2 + 2\pi f_0 t_n + \phi)$$

In the MATLAB code above, what is the value for t_n? What are the values of A, μ, f_0, and ϕ?

(c) Determine the range of frequencies (in hertz) that will be synthesized by the MATLAB script above. Make a sketch by hand of the instantaneous frequency versus time. What are the minimum and maximum frequencies that will be heard?

(d) Listen to the signal to determine whether the signal's frequency content is increasing or decreasing (use `soundsc()`). Notice that `soundsc()` needs to know the rate at which the signal samples were created. For more information do `help sound` and `help sound()`.

Use the code above to synthesize a chirp, then insert your code into the following framework:

```
function   [xx,tt] = mychirp( f1, f2, dur, fsamp )
%MYCHIRP      generate a linear-FM chirp signal
%
%  usage:   xx = mychirp( f1, f2, dur, fsamp )
%
%       f1 = starting frequency
%       f2 = ending frequency
%      dur = total time duration
%    fsamp = sampling frequency  (OPTIONAL: default is 11025)
%
%       xx = (vector of) samples of the chirp signal
%       tt = vector of time instants for t=0 to t=dur
%
if( nargin < 4 )   %-- Allow optional input argument
   fsamp = 11025;
end
```

e. As a test case, generate a chirp sound whose frequency starts at 400 Hz and ends at 4000 Hz with a duration of 7 seconds. Listen to the chirp and include a listing of the mychirp.m function that you wrote. Look at Example 3.8 on page 62 of your textbook to see how to set the parameters.

f. Use the *mychirp.m* function to synthesize a "chirp" signal with the following parameters: time duration of 3 seconds with a sampling frequency of 11025 Hz; the starting frequency is 5000 Hz and the ending frequency is 500 Hz. Listen to signal, what does it sound like (e.g., is the frequency movement linear)? Does it chirp down or up? Create a spectrogram of the signal to verify that you have the correct instantaneous frequency.

Lab Write-up:
Answer all questions in lab including your plots and descriptions of what you heard.

Pre-Lab:
Explain what a beat note is both mathematically and in what you hear.

Problem 3.4 in text

Part 1:
Do parts a and b. Provide plots for both the time signals and the spectrogram.

Part 2:
Answer questions b-f for the parameters given.