ECE 738 Introduction to Signal Processing

Matlab Project I

Spring 2006

Issued: Tuesday, February 21, 2006 **Due:** Monday, March 20, 2006

The goal of this exercise is to matched filter and demodulate a linear frequency-modulated pulse. This project will allow you to explore some of the issues associated with implementing matched filters in practical applications. You will apply the Matlab functions you develop to process a reception from the recent SPICE04 long-range underwater acoustic propagation experiment.

Your report should include any analytical (*i.e.,* pencil/paper) work, Matlab plots and code, and relevant explanations for all parts of the project. Please write a report separate from the Matlab code itself. You are certainly encouraged to include comments in your Matlab code, but I will not consider the comments part of your official report. On the due date, please provide me with a hard copy of your report and an electronic copy of the Matlab functions you wrote. (I will test your algorithm by running it on a data set different from the one provided on the course website.)

Background

Linear frequency modulated (LFM) pulses are often used in sonar and radar applications. The LFM pulse, often called a *chirp* signal, is defined below:

$$s(t) = A\cos\left(\Omega_0 t + \frac{\pi W}{T}t^2\right) \qquad -\frac{T}{2} \le t \le +\frac{T}{2}. \tag{1}$$

Note that the LFM signal $s(t)$ can also be written as

$$s(t) = \mathcal{R}e\left\{s_l(t)e^{j\Omega_0 t}\right\}, \tag{2}$$

where $s_l(t)$, called the *complex envelope*, is defined below:

$$s_l(t) = Ae^{j\frac{\pi W}{T}t^2} \qquad -\frac{T}{2} \le t \le +\frac{T}{2}. \tag{3}$$

The complex envelope is a lowpass (or baseband) representation of the real LFM pulse defined by Equation 1. The phase of $s_l(t)$ varies quadratically with time, thus the derivative of the phase changes linearly with time. Since the derivative of the phase determines the instantaneous frequency, the frequency changes linearly with time (hence the name LFM), *i.e.,*

$$\text{instantaneous frequency } = \frac{d}{dt}\left\{\frac{\pi W}{T}t^2\right\} = \frac{2\pi W t}{T}. \tag{4}$$

Over the pulse duration ($-\frac{T}{2} \le t \le +\frac{T}{2}$), the frequency of $s_l(t)$ changes linearly from $-\frac{2\pi W}{2}$ to $+\frac{2\pi W}{2}$ (or from $-\frac{W}{2}$ to $+\frac{W}{2}$ Hz).

Figure 1 shows the block diagram of a system that can be used to first obtain the lowpass representation of a received signal and then to matched filter the resulting baseband signal. One of the reasons that chirp signals are often used in radar and sonar is that they have good autocorrelation properties, *i.e.*, the convolution of the chirp with its matched filter produces a signal that is concentrated in time.

Project Definition

In this project you will write the Matlab code to implement complex demodulation and matched filtering of an LFM signal.

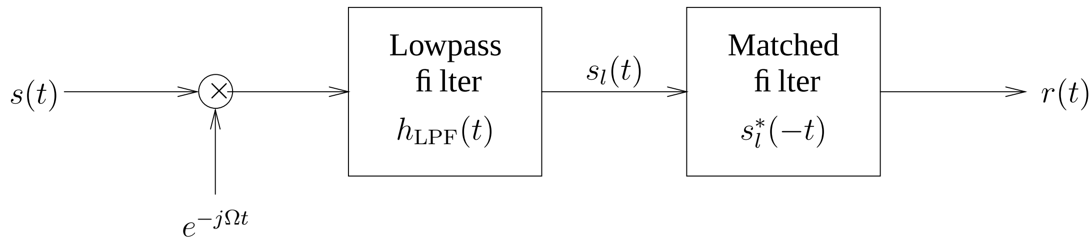

Figure 1: Block diagram of complex demodulator and matched filter.

A. Write a Matlab function called `lfmpulse` to generate samples of an LFM chirp signal over the time interval $-\frac{T}{2} \le t \le +\frac{T}{2}$. Assume that the signal has unit amplitude ($A = 1$). The inputs to the function should be the center frequency $f_0 = \frac{\Omega_0}{2\pi}$ (in Hz), the bandwidth W (in Hz), the timewidth T (in seconds), and the sampling frequency fs (in Hz). The outputs of the function should be a vector containing the samples of the complex envelope of the pulse $s_l(t)$, a vector containing samples of the real pulse $s(t)$, and a vector containing the corresponding sample times t.

Important note: don't confuse the timewidth T of the pulse with the sampling period $T_s = \frac{1}{fs}$.

B. A sampled LFM signal can be complex demodulated by first multiplying the signal by $e^{-j\omega_0 n}$ and then filtering it with a DT lowpass filter. In this part you will design the elements of a digital complex demodulator.

i. Determine an appropriate value for ω_0 in terms of the carrier frequency Ω_0 of the LFM pulse.

ii. Design a DT FIR lowpass filter for a signal sampled at a rate of 1000 Hz. This filter should pass frequencies below 100 Hz with unity gain. The filter should have an attenuation of at least 40 dB for frequencies greater than 120 Hz. Note that since the bandwidth of the chirp signal spans approximately $-\frac{W}{2}$ to $+\frac{W}{2}$, this demodulation filter will work for chirp signals with W up to 200 Hz.

You may use any of the filter design methods you learned in ECE 535 to design the filter and you may use any of the Matlab filter design functions to implement the design calculations. Provide a short justification of why you chose a particular design method and how you selected the necessary parameters for the design. Include plots of the impulse response, and the frequency response magnitude and phase of your lowpass filter.

C. Write a short Matlab function to compute the matched filter for a complex demodulated LFM pulse given a vector of samples of the pulse.

D. The complex demodulation and matched filtering process involves implementing two filters, a lowpass filter and the matched filter. These FIR filters can be implemented several different ways:

- In the time domain using convolution (via Matlab's `conv` command) or a standard DT filter structure (via Matlab's `filter` command).
- In the frequency domain by taking DFT's of the input and the filter, multiplying them together, and inverse-transforming.

Analyze the computational complexity of these two approaches and decide how you will implement these filters. You can use the numbers of multiplies and adds as a measure of the computational complexity.

E. Write a function called `lfm_demod_mf` to implement the complex demodulation and matched filtering of an LFM signal. The function should take as inputs: the received LFM signal, the sampling frequency, and the LFM pulse parameters (center frequency f_0, timewidth T, and bandwidth W). The output should be a the complex matched filtered signal and its associated time vector (for plotting purposes).

F. Test your function using a set of simulated signals that you generate with the function `lfmpulse` you wrote in part A.

First demonstrate that your code works for a noise-free LFM pulse. Note that the output of the matched filter, which is the deterministic autocorrelation of the LFM pulse, has a relatively narrow mainlobe and low sidelobes. These properties are one of the reasons LFM pulses are used in radar and sonar for range estimation.

Second, add zero-mean Gaussian white noise to the pulse and verify that the pulse compression provides an SNR gain. Provide plots of the magnitude of the output signal for all your test cases. Be sure to indicate what timewidth and bandwidth parameters you chose for your test cases.

G. As discussed in class, LFM signals were used in a recent long-range propagation called SPICE04. In this experiment, the source transmitted LFM pulses lasting for 135 seconds. The source swept from 225 Hz to 325 Hz, thus the center frequency was 275 Hz and the one-sided bandwidth was 50 Hz. The total bandwidth was 100 Hz. The signals were received on an array 500 km away from the source. The receiver sampled the signals at a rate of 1000 Hz. The data file `data_331164526.mat` contains the sampled signals for 20 hydrophones on the vertical receiving array. The signals are stored in a 155496 by 20 array (one column for each hydrophone). Use your function `lfm_demod_mf` to complex demodulate and matched filter these signals. Provide a plot of the matched filtered output for each of these 20 hydrophones.

Note that propagation in the underwater waveguide is somewhat complicated as the signal can take more than one path between source and receiver. Thus, you might expect to see multiple arrivals of the chirp signal at each hydrophone. The matched filter should help to resolve these arrivals in time.