ELEC4844/8844 Signal Processing for Software Defined Radio

Assignment 1, Semester 1, 2024

Assignment 1 consists of 3 parts, accounting for 10 marks in total. For questions requiring MATLAB simulation, you must include your MATLAB codes and the appropriate numerical and/or graphical results in your answers.

In Defence 1, you will present your Assignment 1 results and will be asked questions.

PART 1. FREQUENCY ESTIMATION (3 MARKS)

A sinusoidal signal is received in additive Gaussian noise. The received signal and noise is time-sampled and digitised

$$x[n] = A\cos\left[2\pi\frac{f_x}{f_s}(n-1)+\theta\right]+\eta[n]$$

$1 \le n \le N$; f_s is the sampling frequency. The frequency f_x of the received signal is unknown, but is a uniformly distributed random variable in the interval (90 MHz, 110 MHz). θ is a uniformly distributed random variable in the interval ($-\pi$, π). $\eta[n]$ is a sequence of zero-mean Gaussian random variables with variance σ^2.

1a) Find in the engineering literature or create yourself a frequency estimation algorithm based on the received sequence $x[n]$ ($1 \le n \le N$),

$$\hat{f}_x = g(x[n])$$

and write a MATLAB program to implement the algorithm. Use the following parameters: $A = 0.8$, $N = 2^{10}$, $f_s = 800$ MHz (therefore each estimate is obtained using a time record of $NT_s = N/f_s = 1.28$ µs), and $SNR_{dB} = 0$ dB to illustrate your algorithm.

1b) Compute the root-mean-square (rms) error in the frequency estimate

$$\varepsilon_{rms} = \sqrt{E\left[\left(f_x - \hat{f}_x\right)^2\right]}$$

by MATLAB simulation, as a function of SNR_{dB} between -30 dB and 15 dB. $E[\cdot]$ stands for expectation, which is found by running many trials of a simulation (e.g. 1000 times) and then taking the average value. Keep the parameters A, N, and f_s same as above.

Plot ε_{rms} (on the y-axis in log scale) as a function of SNR_{dB} (on the x-axis in linear scale).

1c) Explain why ε_{rms} has an upper bound for small SNR_{dB} and a lower bound for large SNR_{dB}. Use MATLAB simulation to illustrate your arguments.

PART 2. DIGITAL PHASE SHIFTER (3 MARKS)

A chirp (sweep signal) is a signal whose instantaneous frequency increases or decreases with time. Let

$$x(t) = \cos[\theta(t)]$$

its instantaneous frequency is

$$f(t) = \frac{1}{2\pi}\frac{d\theta(t)}{dt}$$

2a) Generate a linear chirp signal with instantaneous frequency

$$f(t) = f_0 + \mu t$$

with f_0 = 220 Hz and μ = 330 Hz/s. The sampling frequency f_s = 48 kHz, and the signal last for 2 seconds. Play the sound of the generated chirp signal and explain what you hear. Plot out the frequency spectrum of the chirp signal.

2b) Find in the engineering literature or create yourself a digital phase shifter that add delay to the input. Process the chirp signal above to generate a delayed signal, and add it to the original input to obtain the output:

$$y(t) = x(t) + x(t + \tau)$$

Try the following values for the delay τ = 0.2, 2, 10, 40, 100, and 200 ms. Play the sound of the output signal $y(t)$. Explain the effect you hear.

2c) Block (i.e. frame-based) processing is typically required for signal processing in real time for lasting input signals.

Partition the chirp signal into blocks of 4096 samples per block (frame), and use either overlap-add or overlap-save method to realise the signal processing above in 2b. Demonstrate that your algorithm is able to process the input audio signal in real time.

Note that you may need to adjust your digital phase shifter to suit the block processing process.

PART 3. RATE CONVERSION AND OVERSAMPLING (4 MARKS)

A piece of music is provided in the auxiliary file ‘audiodata.mat’, which contains the sampled audio signal $x[n]$ and the sampling frequency f_s (= 44.1 kHz).

3a) Downsample the audio signal by $M = 12$. Play the downsampled signal to identify one section of time that you can hear the aliasing effect. Calculate and plot the frequency spectrum of the particular section of the downsampled signal, and compare to the same range of frequency spectrum of the original audio signal.

3b) Design a digital decimation lowpass filter, which requires a passband frequency $f_{\mathrm{PASS}} = 1800$ Hz with passband attenuation $\delta_{\mathrm{PASS}} = 0.1$ dB, and stopband frequency $f_{\mathrm{STOP}} = 1840$ Hz with stopband attenuation $\delta_{\mathrm{STOP}} = 80$ dB. Implement the decimation filter before downsampling. Play the downsampled signal and plot the spectrum (using the same section of time you identified above in 3a) to verify that the aliasing effect is eliminated in the downsampled signal.

3c) Instead of performing the downsampling/decimation in one stage as above, now divide the entire process to two stages, with $M_1 = 4$ for the first stage and $M_2 = 3$ for the second stage. Accordingly, two lowpass filters are required before each stage of downsampling, respectively.

Design the two filters according to the following specifications:

$f_{\mathrm{PASS},1} = 3000$ Hz	$f_{\mathrm{PASS},2} = 1800$ Hz
$\delta_{\mathrm{PASS},1} = 0.05$ dB	$\delta_{\mathrm{PASS},2} = 0.1$ dB
$f_{\mathrm{STOP},1} = 9000$ Hz	$f_{\mathrm{STOP},2} = 1840$ Hz
$\delta_{\mathrm{STOP},1} = 100$ dB	$\delta_{\mathrm{STOP},2} = 80$ dB

Calculate the overall frequency response over the range from 0 to $f_s/2$ (=22.05 kHz) for the two-stage process, and compare to the frequency response of the filter designed in 3b. Process the original audio signal using the two-stage decimation, and play the result to verify whether the overall performance is equivalent to that of one-stage decimation.

3d) Considering the two-stage process above in 3c, calculate the Nyquist frequency for the first stage of decimation.

It can be seen that the value is smaller than $f_{\mathrm{STOP},1}$, meaning some aliasing interference has already occurred during the first stage. Explain whether such aliasing interference is propagated to and retained after the second stage or not, and illustrate your arguments.