ECE 301

TERM PROJECT
LINEAR SYSTEMS

Spring 2021

The class project is intended to be a hands-on, creative learning experience. Each student is expected to demonstrate his/her grasp of the material drawn from the lectures, text, or other sources, by applying signal processing techniques to some problem of interest. To enhance creativity, there is considerable flexibility in both the topic as well as the method of execution. Most students will likely choose to actually implement some interesting signal processing algorithm, or perhaps, a series of algorithms in a simulated signal synthesis or analysis application. Those students choosing to implement algorithms on a computer may use any system they like, including their home PC.

Throughout the class, as well as ECE 220, and other courses, you have been making use of MATLAB. For the final project, though, you have the option of using any software package you prefer including MATLAB, Python, R, LabVIEW, C++, etc. Of course, projects that simply implement an already-available signal processing subroutine are discouraged. Rather, it is more interesting to apply a combination of standard algorithms (possibly library routines) to a problem of interest. Please note that a copy of the software will be required to be handed in.

You may perform this project individually or in a group of 2 or 3 with approval.

Each project will be graded on the basis of (*i*) creative effort; (*ii*) code quality; (*iii*) writeup or documentation. You will need to submit a writeup of 4 pages in IEEE double column single spaced format https://www.ieee.org/conferences/publishing/templates.html (See example 1, example 2, example 3) and your code separately (no page limit).

Alternatively, and recommended, you may submit a MATLAB live document, or Python Jupyter notebook, or equivalent, in lieu of the plain source code. Note that the 4-page report must not contain blocks of raw source code. However, concise pseudocode is allowed. See example 4.

Important Dates

- *March 19* One-page project proposal that describes your project idea and what exactly you plan to do.
- *April 17* Two-page progress report that describes what you have accomplished so far including sample outputs and the remaining to-do items.
- *April 30* Final *submission due.*

Open-ended projects:

The following list is quite incomplete - your ideas may be quite satisfactory. The most important factor to consider is complexity - can I actually finish this project? Be sure to carefully prepare your project proposal so that we can comment on it. Remember, this course is about signals and systems, with some introductory materials on machine learning. Think about the course topic: digital signals & systems. You can design systems to better process signals. You can create signals. You can analyze the outputs of systems and try to infer the input. You can use real data to do classification or regression.

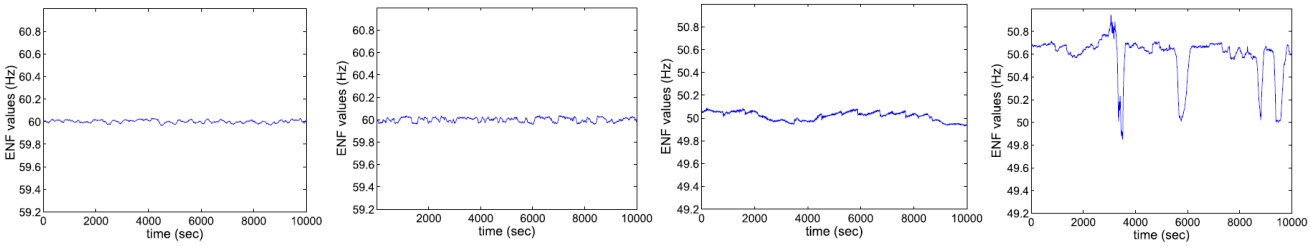
1. Participate a machine learning competition on Kaggle.com or complete a machine learning task using a dataset from Kaggle.com.
2. Implement a word recognition routine.
3. Speaker identification or recognition.
4. Vocoders that synthesize funny audio sounds.
5. Text-to-speech converter.
6. Denoising record recording or old music.
7. Digital watermarking of audio.
8. Extracting a particular musical instrument from an orchestra.
9. Adjustable audio equalizer.
10. Automatic loudness / volume control.
11. "Distortion enhancer" for violin, guitar, banjo, etc.
12. Adaptive audio filter.
13. Implementation of wavelet-based audio coding techniques.
14. Audio compression comparison: DFT vs. DCT vs. wavelet techniques.
15. Signal classification, e.g., noise, music, speech, etc.
16. Remove commercials from recorded radio music.
17. Equalize an analog (FM or AM) communication signal.
18. Equalize a digital communication signal. Tradeoffs here.
19. Image deblurring.
20. Stereo audio processing.
21. Financial prediction. Smooth stock market data. Predict stock prices.
22. Remove lines from scanned notebook paper with writing.

Well-guided projects:

Fourier analysis can be intuitively applied to estimate the frequency of a periodic signal. Below, we provide two real-world applications motivated frequency estimation projects, one on the forensic analysis of multimedia recording using inherently embedded power signatures and the other on contact-free heart rate sensing using facial videos. Project 23 focuses more on simulation and analysis, and Project 24 focuses more on data acquisition and implementation.

23. Frequency Estimator Design [simulation + analysis]

The electric network frequency (ENF) in the US has a nominal value of $f_0 = 60$ Hz and its instantaneous value fluctuates round the nominal value as a function of time [1]. The following plots show some sample ENF signals from various power grids [2].



US East Texas Ireland Lebanon

The ENF signals can be captured by audio or video recordings made in areas where there is electrical activity. This makes the ENF a good criterion for the forensic analysis of a multimedia recording. With a proper reference database, one can tell when and where a recording was made, and whether the recording has been tampered.

Let us formally define the ENF signal as $f(t) = f_0 + u(t)$, where f_0 is the nominal value and $u(t)$ is the random, fluctuating component. Research found that the random signal $u(t)$ can be modeled as an autoregressive (AR) random process [1]. In this project, we assume that its sampled version (with a discretization step size of 0.2 seconds) follows the first-order AR process, namely,

$$u[n] = 0.9\, u[n-1] + e[n],$$

where the $e[n]$ is a zero-mean white Gaussian noise process with standard deviation $\sigma_e = 0.05$ Hz. Note that the discretization leads to the relationship of $n = t / 0.2$, and the discretized ENF signal can still be decomposed into $f[n] = f_0 + u[n]$. A realization of a white Gaussian process of length n with standard deviation of q can be obtained by using the `q*randn(1, n)` command.

(a) [Simulate ENF Signals] Generate 5 different discrete ENF signals of 10 minutes long each and plot them in the same figure. Limit the dynamic range of the vertical axis to around 60 Hz to ensure that the fluctuation can be easily observed. Label the horizontal axis in the unit of minute. Use different color for different ENF signals.

(b) [Simulate Voltage Signal of Power Supply] For each ENF signal $f[n]$, synthetically generate a 10-minute long sinusoidal-like signal $x[n]$ whose frequency changes every 0.2 second at a sampling frequency of $f_s = 150$ Hz using the following equation for frequency modulation (FM):

$$x[n] = \cos(\phi_0 + 2\pi \textstyle\sum_{\ell=1}^{n} f[\ell] / f_s),$$

where ϕ_0 is a random initial phase that can be implemented using `2*pi*rand()` and $f[\ell]$ is the frequency during ℓth sampling period.

(c) [Implement Frequency Estimator] Implement a periodogram-based frequency estimator. You can obtain the estimated power spectral density (PSD) function by applying a fast Fourier transform (FFT) on an input signal and then square magnitude of the FFT result. Note that the PSD is a real, nonnegative function of frequency. Use a large `nfft` parameter for FFT to ensure the PSD function is smooth enough. If the input is a sinusoid signal, a peak can be seen in the PSD function. Use Matlab's built-in peak detection algorithm to find the location of the peak and then map it the frequency value in Hz.

Note that your frequency estimator should output an estimated frequency given an input time-domain signal $x[n]$ and its sampling frequency. To ensure the correctness of the frequency estimator you implemented, you may want to conduct "unit testing" for your function: Use synthetically generated signals with known ground-truth frequency to validate. Report your choice of `nfft` parameter and the resolution of frequency in Hz.

Note that if you want to test the frequency estimator on a 10-minute long recording, each time you should only consider a small window (e.g., 2-second long) of signal as $x[n]$. To track the frequency of the whole recording, you can cut it into nonoverlapping segments, or you can use a sliding window with 50% to 95% overlap.

If you want to compare estimated ENF signal to the ground-truth ENF signal the but their sampling rate do not match, you may resample the estimated ENF signal using Matlab's built-in function.

(d) [Frequency Interpolation] No matter how large the `nfft` parameter is, the estimated frequency can only be an integer multiple of the frequency resolution. You will design an interpolation process that can refine the coarse frequency estimate obtained in (c). Consider the frequency index i that has the largest PSD value, $S(i)$. Also consider the value of the peak's left neighbor ($i-1$, $S(i-1)$) and right neighbor ($i+1$, $S(i+1)$). Derive a deterministic, mathematical function that takes as input i, $S(i-1)$, $S(i)$, and $S(i+1)$ and output an adjusted index of peak $i^* \in (i-1, i+1)$ that is the peak position of a quadratic curve passing through $S(i-1)$, $S(i)$, and $S(i+1)$.

(e) [Performance of Estimator Under Noisy Scenarios] Define the signal-to-noise ratio (SNR) = $10 \log_{10}(A^2/\sigma^2)$ of a noise corrupted ENF signal, $y[n] = x[n] + v[n]$, where A is the amplitude of $x[n]$, and σ^2 is the variance of the additive white Gaussian noise, $v[n]$. (Do not confuse σ with σ_e.) Generate 9 groups of 8-second signals $y[n]$ with SNR = −40, −30, …, 0, …, 30, 40 dB, respectively, with a sampling frequency of 150 Hz. Each group of signals consists of 1,000 realizations of sinusoid-like signals with different initial phases ϕ_0 and ENF signals $f[n]$. What is the effect of noise levels to the accuracy of the frequency estimates in terms of the mean squared error (MSE)?

(f) Examine how the change of parameter values such as the length of window, the overlapping percentage of the windows affect quality of frequency estimation. Discuss your findings.

24. Heart Rate Estimation via Facial Videos [acquisition + implementation]

Below is a concise introduction to heart rate estimation using face videos appeared in [3]: "Contact-free monitoring of the heart rate using videos of human faces is a user-friendly approach compared to conventional contact based ones such as electrodes, chest belts, and finger clips. Such monitoring system extracts from a face video a 1-D sinusoid-like face color signal that has the same frequency as the heartbeat. The ability to measure heart rate without touch-based sensors is attractive and gives it potentials in such applications as smart health and sports medicine."

You are given a preprocessed face color signal sampled at 30 Hz in which the heart rate fluctuates between 78 and 85 beats per minute (bpm). Note that 60 bpm = 1 Hz. When necessary, increase the `nfft` parameter of `FFT()` to improve the resolution of displayed frequency.

(a) [Implement Frequency Estimator] Implement the periodogram-based frequency estimator described in part (c) of the "Frequency Estimator Design" project.

(b) [Track Heart Rate Using Preprocessed Data] Use the periodogram with 98% window overlap to estimate the heart rate signal. Make sure the time series of estimated heart rate have no values outside the interval of 78 to 85 bpm. Determine an optimal window length between 1 to 10 seconds with a step size of 1 second.

(c) [Take a Facial Video] Take a 2-minute facial video while your face appears completely still within the video recording. Please do some light exercise such as walking upstairs before capturing the video to ensure that your heart rate is high at the beginning and low in the end. For safety, do not exercise excessively such as running before taking the facial video. Make sure the video capturing condition is well lit. Try to avoid direct sun light on your face or bright outdoor lighting in the background of the video. Indoor light usually works better. Try to avoid light directly above your head or from your back.

(d) [Track Heart Rate Using Self-Captured Video] Load your video in Matlab using the computer vision toolbox. For each video frame, crop out a rectangular region from your right cheek, and do a spatial averaging for the color values within the region. You will obtain 3 average values from each frame and need to concatenate the values along the time to obtain time series for R, G, and B channels, respectively. Linearly combine the time series from three channels using weights −1, 2, and −1 to obtain a so-called a remote photoplethysmography (rPPG) 1-D signal. Use the frequency estimator to track your heart rate from your right cheek and plot it as a function of time. Repeat the same procedure for your left cheek and the forehead. If your implementation is correct and facial video was captured in a way that the heartbeat signal survives, you should be able to confirm that the three estimated heart rates signals are consistent.

[1] R. Garg, A. L. Varna, A. Hajj-Ahmad and M. Wu, "'Seeing' ENF: Power-signature-based timestamp for digital multimedia via optical sensing and signal processing," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 9, pp. 1417-1432, Sept. 2013.
[2] A. Hajj-Ahmad, R. Garg and M. Wu, "ENF-based region-of-recording identification for media signals," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1125-1136, Jun. 2015.
[3] Q. Zhu, C.-W. Wong, C.-H. Fu, and M. Wu, "Fitness heart rate measurement using face videos," *IEEE International Conference on Image Processing (ICIP'17)*, Beijing, China, 17–20 Sep. 2017.

Resources for Finding Ideas

There are various places to find ideas for projects. You might check for books on application areas of interest: speech recognition, wavelets, array processing, acoustics, etc. Other interesting references:

Signal Processing Magazine
https://signalprocessingsociety.org/publications-resources/ieee-signal-processing-magazine/past-issues

Proceedings of the IEEE
https://proceedingsoftheieee.ieee.org/view-recent-issues/browse-past-issues/

You can access the articles on IEEExplore via NC State library:
https://www.lib.ncsu.edu/databases/more_info.php?database=28715

ECE 301 **TERM PROJECT**
LINEAR SYSTEMS Spring 2021

GROUP PROJECTS

Joint group projects (involving no more than 3 students) are acceptable, and are even encouraged. The following guidelines, however, apply:

- Group projects must be cleared with me first. This may be done as part of the Project Proposal.

- In the Project Proposal, the creative effort, as well as other work, applied by each participant should be laid out as much as possible. Thus, each participant should be helping to solve the problem.

- Same holds for the Project Progress Report.

- There should be a single Final Project Report, complete with a cover page indicating all participants' names. For individual projects, the write-up is not to exceed 8 pages of text; for two-person projects, not to exceed 16 pages; for three-person projects, not to exceed 24 pages (figures not included in these). In other words, no more than eight pages per participant.

- The Final Project Report should be divided into sections, where each participant outlines his/her contributions, what he/she learned, what he/she did, etc. While it is understood that there will be a synergism in doing a combined project, and that all participants may contribute ideas to various aspects of the project, an attempt should be made to clarify the work efforts.

- The most important aspect of the project is “what you learned.” This is what I will be most interested in. Therefore, I suggest that the project be selected based on your interest(s). While ambitious projects (within reason) are not discouraged, ambitiousness is not the basis for grading. Conversely, grading will not be based on the successfulness of a difficult project, but rather, on the effort applied and the amount learned.