# EECE 5698 Fall 2025
# PreLab 2: Aliasing and Sound

Instructor: Prof. Kevin Fu
TAs: Hui Zhuang, Nuntipat Narkthong
Last updated: 09/15/2025, 09 PM
Submission Deadline: Sept. 25, 11:45 AM.
Submit your report as an individual on Canvas.

## Part 0: Looking for teammates

Starting from this lab, you are required to work in groups of two students. Each student must submit their own Prelab individually, but the Lab report should be submitted jointly by the group. You are free to choose your partner. If you are unable to find one, please contact us promptly via Piazza. You may also change partners for different labs if you wish.

**Question 0: Input your name and your teammate's name.**

## Part 1: Signal Aliasing

### Basics

Signal injection attacks against sensors largely depend on sensors' incapability of handling non-linearities. Specifically, the injection signals produced by the adversaries generate non-linear components after passing through the sensor conditioning paths. **These non-linear components are often not taken into consideration by the sensor designers and thus cannot be eliminated by the sensors themselves.** Obviously, such non-linear behaviors are often not documented in the sensor specifications and user manuals either, creating a gap of information between the adversaries and sensor users that allow adversaries to manipulate a sensor system/application.

Besides the non-linear phenomena including the asymmetric clipping in "Walnut" [1], aliasing of signals is another phenomenon widely explored in previous research to create unexpected sensor signal transformations including the paper [1] that lab02 is based on. Aliasing often happens in the analog-to-digital converters (ADCs) of sensors. When a physical analog (continuous) signal has a frequency larger than half of the ADC sample rate, i.e., the Nyquist frequency, an alias of this signal that has another frequency will be produced in the digitized

discrete signals. One good news is that aliasing is at least predictable if the analog signal frequency and the ADC sample rate are known. The formula is:

$$f_a = |2m\, f_N - f_s|$$

where $f_N$ is the Nyquist frequency, $f_s$ is the signal frequency, and m is an integer such that $f_a \leq f_N$.

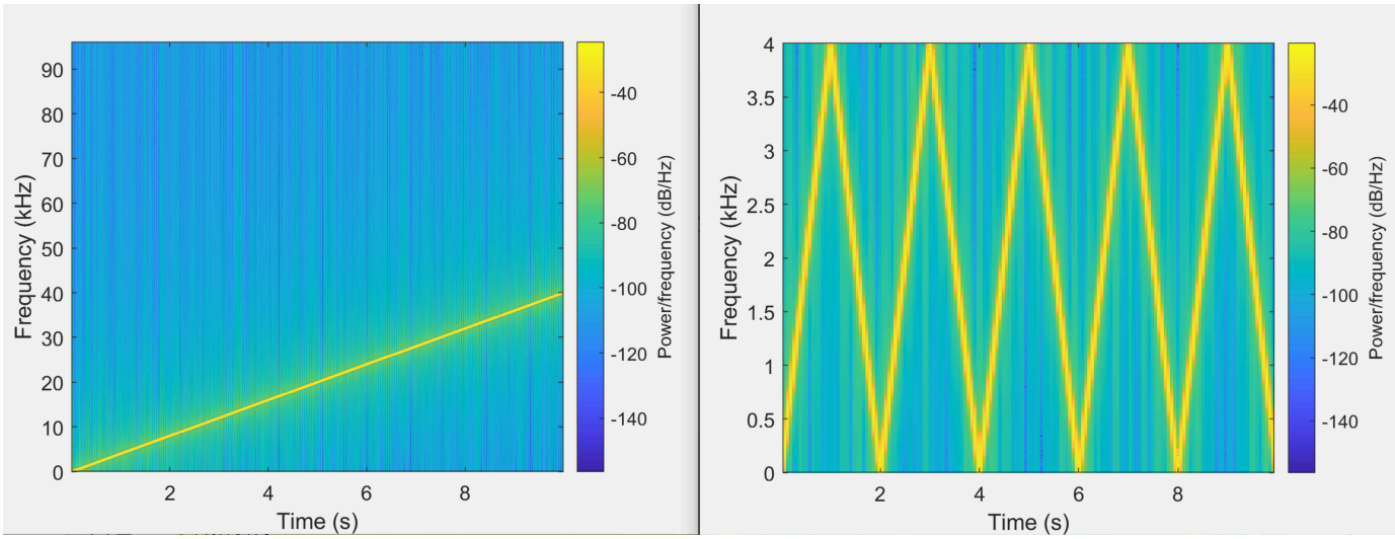


Fig. 0. (Left) A 40k Hz chirp without aliasing that is stored with a sample rate of 192k Hz. (Right) The 40k Hz chirp aliased by using a virtual ADC with a sample rate of 8k Hz.

Fig. 0 (left) shows a simulation of aliasing with a chirp signal linearly increasing from 1 to 40k Hz in 10s that gets sampled by an ADC with a sample rate of 8k Hz. As shown by the spectrogram, signals higher than 4k Hz are aliased to frequencies lower than 4k Hz. Note that there is no way to realistically represent any continuous physical signals in computers since computers are digital. The way to simulate continuous signals in computers is to use a high enough sample rate (at least twice the signal frequency) when storing the simulated continuous signals so that there is no aliasing. In this example, we stored the 40k Hz chirp signal in a 192k Hz sample rate wav file. **Using a simulated ADC to sample the simulated physical signals in computers is essentially a resampling process.** Note that all simulations in this prelab need to be done in Matlab or Python as Audacity does not allow resampling that creates aliases.

**Question 1:**

- Submit a figure of the unalised and aliased spectrograms that replicates Fig. 0. Note you need to come up with a way by yourself to simulate an ADC with a sample rate of 8k Hz. Use the wav file provided above. The two spectrograms you plotted should have 192k Hz and 8k Hz sample rates respectively.
- The virtual ADC needs to be implemented in Matlab/Python. Note that once you understand what an ADC does, the virtual ADC is just one line of code without requiring any libraries. You can plot spectrograms with either Matlab/Python or Audacity. If plotting

spectrograms with Audacity, you need to save the aliased signals in a new wav file. Both Matlab and Python allow you to export data to wav.

- You should clearly label the X and Y axes with the same frequency ranges as Fig. 0.

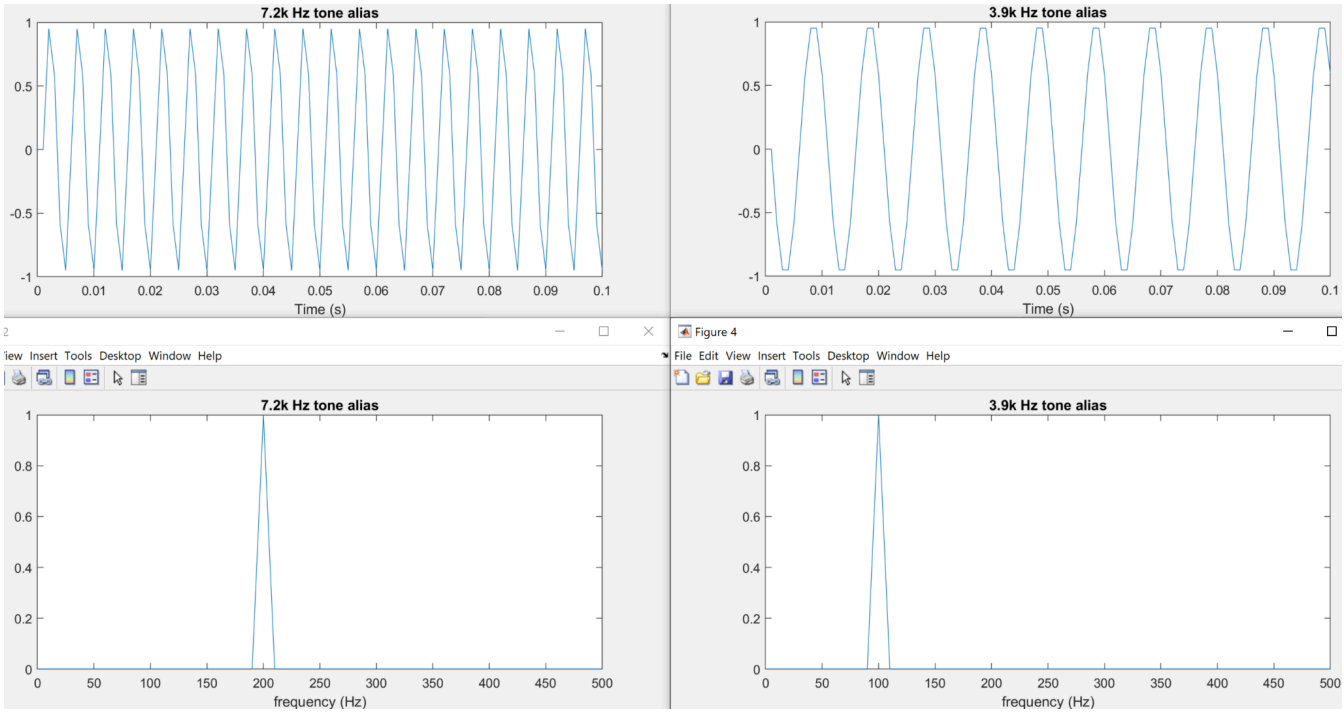


Fig. 1

**Question 2:**

- With an ADC sample rate of 1k Hz, what are the frequencies of the aliases corresponding to 7.4k, 3.5k, and 2k Hz respectively?
- Submit a figure with 4 subfigures that replicates Fig. 1. The figures show the time and frequency domain (FFT) alias signals in the first 0.1 seconds under a 1k Hz ADC sample rate that corresponds to the provided 7.2k Hz and 3.9k Hz signals. Remember to do FFT only over the first 0.1s of the aliases. The Y axes of these plots do not matter, but you need to clearly specify the X axes (time or frequency). Clearly specify the original signal frequency on your subfigures.

## DC Output Biasing

As you have already seen in Question 2, the alias of an AC signal inputted into the ADC can even become a DC signal in certain conditions as shown in Fig. 2. Specifically, the injected signal frequency needs to be multiples of the ADC sample rate. The capability of achieving DC

output biasing is the foundation of the acoustic injection attack against accelerometers explored in [1].

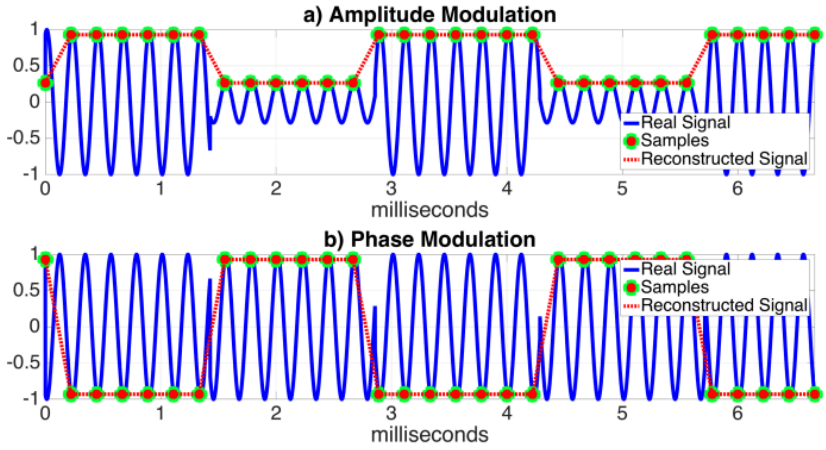


Fig. 2. DC output biasing [1]

DC output biasing allows adversaries to change the output of the sensor's ADC in a more controlled manner. Note that such fine-grained control does not come for free: the adversary needs to know the sample rate of the sensor, either through experiments or reading its specifications, and needs to accurately control the output frequency of its injection signals.

Another important thing to know about DC output biasing is its sensitivity to the phase of signals relative to the ADC sampling points, as shown in (b) of Fig. 2. Fig. 3 shows a simulation of DC output biasing with the 3.9k Hz signal (with a 192k Hz audio sample rate), an ADC sample rate of 100 Hz, and five different phase shifts (offsets) when the ADC resamples the signal in the first 0.1s. For example, an offset of 0 means the ADC resampling the 3.9k Hz signal from the very beginning of the signal while an offset of 10 signal sample points means resampling the signal starting from the 11th data point in the wav file provided. Note that the signal sample point here refers to the smallest time increment in the simulated physical signal file (192k Hz sample rate). For example, a signal sample point offset of 2 means a time shift of 2/192k seconds in the physical world and a signal phase shift of 2*2pi*3.9k/192k. As you can see, a very small sampling phase shift/offset can cause a large variation in the DC output biasing amplitude. This suggests the adversary needs very fine-grained control of the injection signal in order to produce a desired good DC bias.

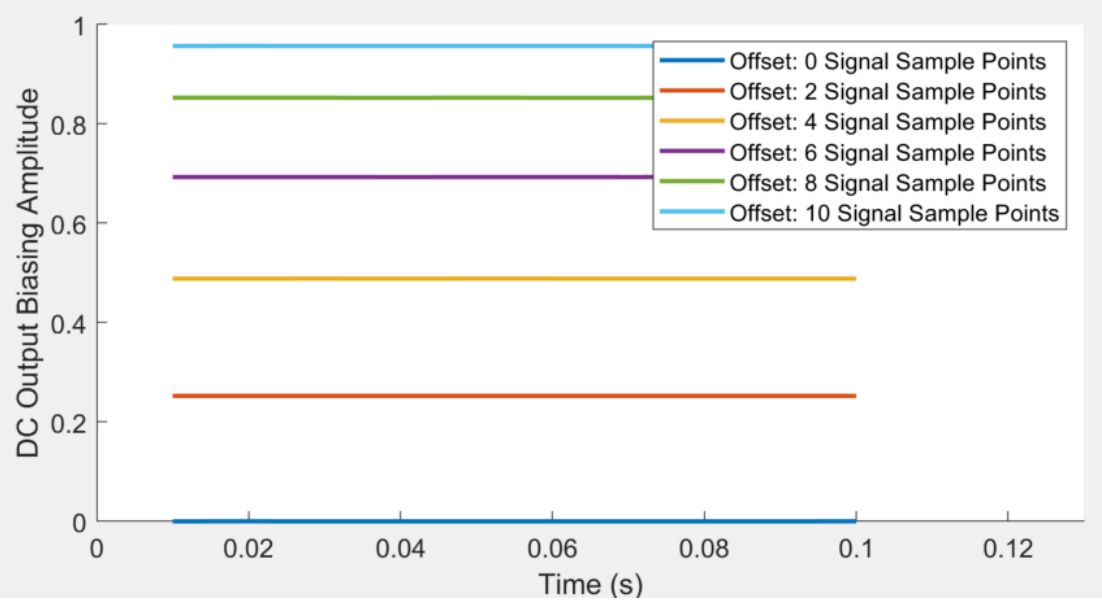


Fig. 3

**Question 3:**

- Submit a figure that replicates Fig. 3. The aliases plotted should have a sample rate of 100 Hz. Only plot the first 0.1s of the aliases. You need to give clear legends of the aliases. The aliases you get are supposed to have the same amplitudes as the ones in Fig. 3.

## Part 2: Digital & Analog Filtering

You may be wondering why not just filter out the aliases in the ADC outputs using an easy digital filter in your computer/microcontroller as a defense. The short answer is that it seriously reduces the available bandwidth of the sensor outputs and thus affects the sensor system's usability. Imagine you have an ADC with a sample rate of 1k Hz. All attacker-induced aliases are already under 500 Hz. So if you want to remove them, you need a digital filter whose cutoff frequency is much lower than 500 Hz. On the other hand, the benign signals that sensor users want to utilize are also mostly 500 Hz (that's why sensor designers use such a 1k Hz ADC). The digital filter will thus also eliminate these benign signals. On the contrary, analog filters operate before the ADC turning injected signals into aliases and can reduce just adversarial signals without affecting the benign signals too much. We will test an analog low pass filter in Lab02. You can refer to Section 4 of [2] for a more detailed explanation.

**Question 4:**

- Let there be an ADC with a sample rate of 400 Hz. We define low pass filters' cutoff frequencies as the 3dB-attenuation points. What cutoff frequencies do you need to

achieve 3dB attenuation in the amplitude of attacker injected signals using digital low pass filters when the injected signals have frequencies of 580 and 750 Hz respectively?
- What are the cutoff frequencies needed for analog low pass filters to achieve such 3dB attenuation?

# Part 3: Sound and Mechanical Resonance

## Sound

A core concept of Lab02 is the mechanical nature of sound waves. Essentially, sounds are vibrations of objects. The sounds humans can hear is often due to vibration of air molecules that further vibrates our eardrum. Of course, we also know that humans can hear sounds through bone conduction headphones which propagate sounds through bones instead of air molecules. In summary, sound embodies the transmission of kinetic energy through a system of different mediums.

One possible categorization of sound is air-borne versus structure-borne. Air-borne sounds describe the energy transmission through air molecules alone, while structure-borne sounds describe the energy transmission through non-air objects. Oftentimes, structure-borne paths allow for higher efficiency (lower energy loss) and higher capacity of energy transmission. The two types of transmissions almost always coexist.

In the attack studied in [1] and Lab02, energy generated by electronic speakers propagates to accelerometers, essentially changing the accelerometer readings by vibrating the sensing mass of the sensor. [1] mainly analyzed air-borne sounds because it represents a more realistic attack scenario where the adversaries do not need to physically get close to the sensors. However, this requires a high energy generated by the speaker because of the low efficiency of air-borne transmission. In our lab, we will explore structure-borne propagation instead to amplify the attack effectiveness.

## Mechanical Resonance

Similar to the concepts of electrical frequency response and resonance introduced in Lab01, mechanical systems also have these properties. From a physics perspective, electrical circuits and mechanical structures can be described by highly similar wave propagation functions that determine their frequency responses and resonance points. In the mechanical systems of lab02 that consist of speakers and MEMS accelerometers, adversaries may also maximize the effectiveness of their attacks by using frequencies close to the systems' mechanical resonance points. Also note that real-world systems are often too complex to be modeled by math equations. It is thus highly challenging for adversaries to directly calculate frequency response

and resonance. Given this, frequency sweep is a common approach to get an estimation of these properties.

# Part 4: Tool Preparation

## Route sensor readings from Teensy 4.0 to laptop and save data

We need to save and analyze the accelerometer sensor readings routed by the Teensy 4.0 microcontroller to our laptop through USB serial in Lab02. The Teensy board reading accelerometer outputs works with the Arduino IDE on Linux, Windows, and macOS. Besides installing Arduino IDE, you just need to install the Teensyduino add-on. The Arduino IDE looks like Fig. 4 when the add-on is installed. You have already done this step in PreLab 01.

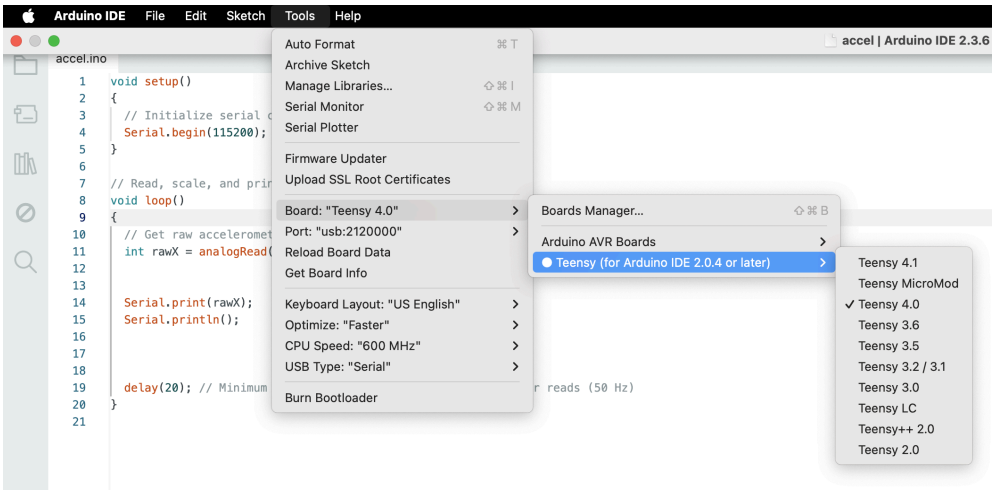


Fig. 4

We also provide a Python script (Python 3) that can save sensor data on your laptop. The usage of this script is described here. It is recommended to use Python 3.6 or 3.8, but other Python 3 versions might also work. **Before the lab, please install the dependencies of this script and make sure at least one laptop of your group is able to run this script**. If properly installed, the following message should appear when you try to run the script without connecting a Teensy board to your laptop.

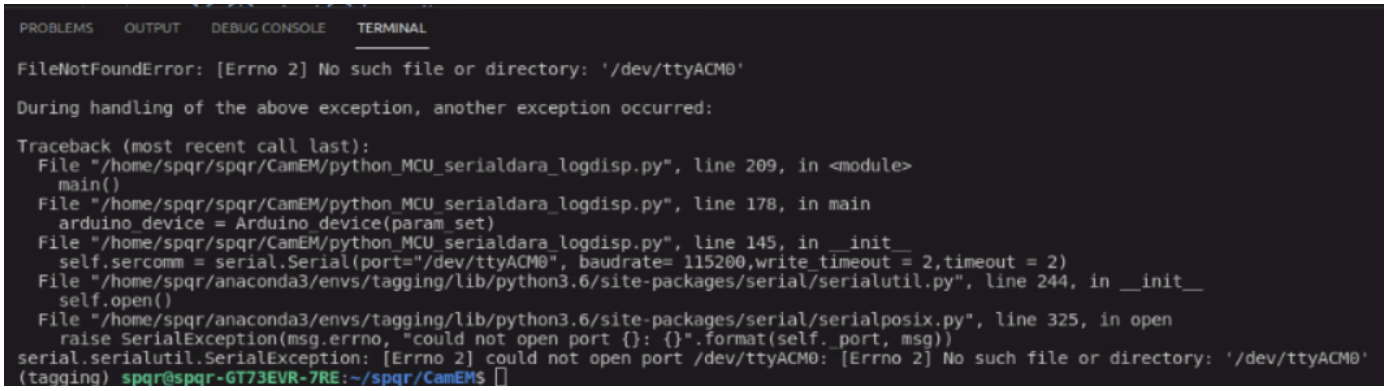


Fig. 5

The sensor readings of the accelerometer's X axis will be saved in a csv file. You also need to prepare a Matlab/Python script that is able to read these csv files and plot the signals as you will need to do this for the in-lab experiments. You can use this example csv file to prepare you read&plot script. All saved csvs in lab02 have a sample rate of 50 Hz.

When working as groups of two people for the in-lab experiments, it is recommended that one member focuses more on making the software needed to work on the laptop after the equipment such as Teensy 4.0 boards are handed to them, and the other student work on the setup and connections needed.

**Question 5:**

- Plot the data in the example csv file with the correct X axes (time).

[1] Trippel, Timothy, et al. "WALNUT: Waging doubt on the integrity of MEMS accelerometers with acoustic injection attacks." *2017 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2017.
[2] Bolton, Connor, et al. "Touchtone leakage attacks via smartphone sensors: mitigation without hardware modification." *arXiv preprint arXiv:2109.13834* (2021).