GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL of ELECTRICAL and COMPUTER ENGINEERING

ECE 2026 Spring 2018
Lab #11: Interference Removal from Electro-Cardiogram (ECG) Signals

Date: 9–12 Apr. 2018

Pre-Lab: You should read the Pre-Lab section of the lab and do all the exercises in the Pre-Lab section ***before your assigned lab time.***

Verification: The Exercise section of each lab must be completed **during your assigned Lab time** and the steps marked *Instructor Verification* must also be signed off **during the lab time**. One of the laboratory instructors must verify the appropriate steps by signing on the **Instructor Verification** line. When you have completed a step that requires verification, simply raise your hand and demonstrate the step to the TA or instructor. Turn in the completed verification sheet to your TA when you leave the lab.

Lab Report Questions: The Lab Report Sheet has a few lab related questions that can be answered at your own pace. The completed Lab-Report sheet is due at the beginning of the next lab.

Forgeries and plagiarism are a violation of the honor code and will be referred to the Dean of Students for disciplinary action. You are allowed to discuss lab exercises with other students, but you cannot give or receive any written material or electronic files. In addition, you are not allowed to use or copy material from old lab reports from previous semesters. Your submitted work must be your own original work.

1 Introduction

IIR filters can be used for many tasks; bandpass filtering is the most common, but nulling or notch filters are also very useful for removing interference, especially narrowband interference caused by power lines. The MATLAB GUI **PeZ** is an ideal tool for studying the poles and zeros of IIR filters.

2 Pre-Lab

Please read through the information below prior to attending your lab.

2.1 Objective

The goal of this lab is to explore the connection between the time domain (n), the frequency domain ($\hat{\omega}$), and the z-transform domain, using the GUI **PeZ** in MATLAB.

1. *Placing Zeros:* When placed on the unit circle, zeros of $A(z)$ will force the frequency response to be zero which can then be used to "null out" sinusoids at one frequency.

2. *Placing Poles:* When placed near the unit circle (but inside), roots of $B(z)$ will create peaks in the frequency response which can be used to form LPFs, HPFs, and BPFs.

3. *Designing IIR Notch Filters* requires conjugate zeros on the unit circle (UC) with poles at the same angles, just inside the UC. The frequency response of the notch is much sharper than a nulling filter, which is an FIR filter formed only from the conjugate zeros on the UC.

2.2 PeZ GUI: An Introduction

In order to build an intuitive understanding of the relationship between the location of poles and zeros in the z-domain, the impulse response $h[n]$ in the n-domain, and the frequency response $H(e^{j\hat{\omega}})$ in the $\hat{\omega}$-domain, a graphical user interface (GUI) called **PeZ** was written in MATLAB for doing interactive explorations of the three domains simultaneously.[1] **PeZ** is based on the system function, represented as a ratio of polynomials in z^{-1}, which can be expressed in either factored or expanded form as:

$$H(z) = \frac{B(z)}{A(z)} = G\,\frac{\prod\limits_{k=1}^{M}(1 - z_k z^{-1})}{\prod\limits_{\ell=1}^{N}(1 - p_\ell z^{-1})} = \frac{\sum\limits_{k=0}^{M} b_k z^{-k}}{1 - \sum\limits_{\ell=1}^{N} a_\ell z^{-\ell}} \qquad (1)$$

where M is the number of zeros and N the number of poles. The filter coefficients are the polynomial coefficient $\{b_k\}$ in the numerator, and $\{1, -a_\ell\}$ for $\ell = 1, 2, \ldots, N$ in the denominator. The gain (G) is equal to b_0. If the poles (p_ℓ) and zeros (z_k) occur in pairs that satisfy the complex-conjugate property, then the following property holds:

A polynomial with real coefficients has roots that are real, or occur in complex-conjugate pairs.

The **PeZ** GUI is contained in the *SP-First* toolbox. To run **PeZ**, type `pezdemo` at the command prompt and you will see the GUI shown in Fig. 1.[2]

2.3 Controls for PeZ using `pezdemo`

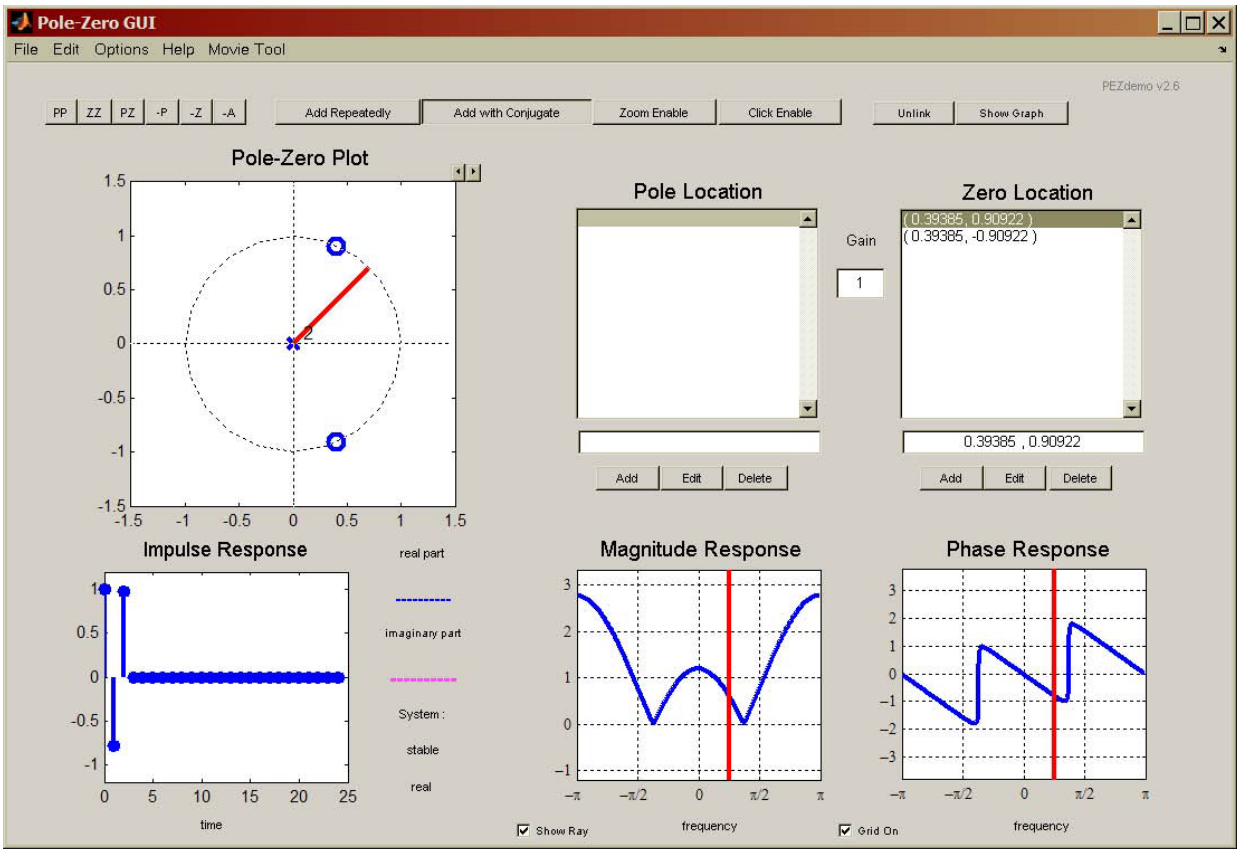
The **PeZ** GUI is controlled by the `Pole-Zero Plot` panel in the upper left where the user can add (or delete) poles and zeros, as well as move them around with the pointing device. For example, Fig. 1 shows a case where two (complex-conjugate) zeros have been added, and **PeZ** has automatically included two poles at $z = 0$, so that the system is causal. The button named `ZZ` was used to add these zeros; the button `PP` would be used to add poles. By default, the `Add with Conjugate` property is turned on, so poles and zeros are typically added in pairs to satisfy the complex-conjugate property, giving real coefficients.

To learn about the other controls in `pezdemo`, access the menu item called “Help” for extensive information about all the **PeZ** controls and menus. Here are a few things to try. In the `Pole-Zero Plot` panel you can selectively place poles and zeros in the z-plane, and then observe (in the other plotting panels) how their placement affects the impulse and frequency responses. The red ray in the z-domain window is tied to the red vertical lines on the frequency responses, and they move together. This helps identify frequency domain features that are caused by pole locations or zero locations, because the angle around the unit circle corresponds to frequency $\hat{\omega}$. In **PeZ** an individual pole/zero pair can be moved around and the corresponding $H(e^{j\hat{\omega}})$ and $h[n]$ plots will be updated as you drag the pole (or zero).

Since exact placement of poles and zeros with the mouse is difficult, there are two ways to get a specific system into the GUI First, the menu `File->Import` can be used to load the numerator and denominator coefficients. Second, an `Edit` button is provided for numerical entry of the real and imaginary parts. Before you can edit a pole or zero, however, you must first select it in the list of `Pole Locations` or `Zero Locations`. Next, you will have to make the necessary change in the edit field (box above the `Edit` button) and then press the `Edit` button for the change to take effect. Removal of individual poles or zeros can also be performed by using the `-P` or `-Z` buttons, or with the `Delete` button. Note that all poles and/or zeros can be easily cleared by clicking on the `-A` button.

[1] The original **PeZ** was written by C. Ulmer, with a later version by K. Kong.

[2] The command `pez` might invoke an obsolete version of **PeZ** which uses a black background in all the plot regions.



Gain 1

Zero Location:
(0.39385, 0.90922)
(0.39385, -0.90922)

0.39385 , 0.90922

Figure 1: GUI interface for pezdemo running in MATLAB version 6. A length-3 FIR filter is shown. The edit box for Zero Locations gives the zeros in rectangular coordinates. The two poles at the origin are not movable, so they are not listed in the Pole Location edit box.

2.4 Relationships between z, n, and $\hat{\omega}$ domains

When you work through the lab exercises, you should keep track of your observations by filling in the worksheet at the end of this assignment. In general, you want to make note of the following quantities:

- Is the length of $h[n]$ finite or infinite? For the FIR case, what is the total length of the filter?
- How does $h[n]$ change with respect to its rate of decay for IIR filters? For example, when the impulse response is of the form $h[n] = a^n u[n]$, the impulse response will fall off more rapidly when a is smaller; and if $|a| > 1$ the impulse response will grow without bound.
- A *stable system* is one that obeys the *bounded-input gives a bounded-output* property. When the impulse response grows without bound, the output is not bounded even though the input is $\delta[n]$ with is bounded by one.
- If $h[n]$ exhibits an oscillating component, what is the period of oscillation? Also, estimate the decay rate of the "envelope" that overlays the oscillation.
- How does $H(e^{j\hat{\omega}})$ change with respect to null location, peak location and/or peak width?

Note: review the "Three-Domains - FIR" under the Demos link for Chapter 7 and "Three-Domains - IIR" under the Demos link for Chapter 8 for movies and examples of these relationships.

Try various operations with **PeZ** to gain some familiarity with the interface. Below are some suggested filters that you can create.

2.5 Zeros

2.5.1 Create a Second-Order FIR Filter with PeZ

Implement the following FIR system: $H(z) = 1 - z^{-1} + z^{-2}$ by factoring the polynomial and placing the two zeros correctly. Observe the following:

- The impulse response $h[n]$ values are equal to the polynomial coefficients of $H(z)$.
- The frequency response has nulls because the zeros of $H(z)$ lie exactly on the unit circle. Compare the frequencies of the nulls to the angles of the zeros by using the red ray.
- Move the zero-pair around the unit circle and observe that the location of the null in the frequency response also moves.

2.5.2 Create an FIR Filters with Many Zeros

The 6-point running-sum FIR digital filter has the following system function:

$$H(z) = \sum_{n=0}^{5} z^{-n} = \frac{1 - z^{-6}}{1 - z^{-1}}$$

(a) Use the `roots` function in MATLAB to determine the zeros of $H(z)$ for this 5-th order FIR filter. *Note:* The formula for $H(z)$ involves a numerator-denominator cancellation that removes the zero at $z = 1$. This is called a *pole-zero cancellation.*

(b) In **PeZ** place the five zeros of the FIR filter at the correct locations. One way is to use the `edit` feature to enter the exact value of the zeros; another way is to do `File->Import` to load the numerator and denominator coefficients. Observe that the impulse response will be all ones when you have the correct zero locations. Describe the frequency response of the filter: low pass versus high pass.

(c) List all the frequencies, $\hat{\omega} \in (-\pi, \pi]$, that are nulled by the length-6 FIR filter.

Make a different FIR filter that has the system function $H_c(z) = 1 - z^{-6}$ which is the numerator above.

(a) Use **PeZ** to place the six zeros of this filter, or add one zero to the previous configuration. Write a formula for the impulse response, $h_c[n]$ using weighted, shifted impulses.

(b) The frequency response of the FIR filter in the previous part is called a *comb filter.* Explain why is it should be called a "comb", and determine where its nulls are located for $\hat{\omega} \in (-\pi, \pi]$.

(c) One of the zeros of $H_c(z)$ will be at $z = -1$. Take that zero and move it from $z = -1$ to $z = +1$ so that you have a double zero at $z = +1$. Write down the system function $\tilde{H}_c(z)$ for the new filter. *Hint:* The GUI contains the answer if you know where to look.

(d) Describe the frequency response of the filter created in part (c), i.e., LPF, HPF, or BPF.

2.6 Poles for IIR Filters

IIR filters have poles at locations away from $z = 0$, unlike causal FIR filters which have poles only at the origin. Moving these poles will change the frequency response and impulse response of an IIR filter.

2.6.1 Create a First-Order IIR Filter with PeZ

Implement the following first-order IIR system: $H(z) = \dfrac{1 - z^{-1}}{1 + 0.9z^{-1}}$ by placing its pole and zero at the correct locations in the z-plane. First, try placing the pole and zero with the mouse, and then use the `Edit` feature under `Pole Location` or `Zero Location` to specify exact locations. Since **PeZ** wants to add complex-conjugate pairs, you might have to delete one of the poles/zeros that were added; or you can turn off the `Add with Conjugate` feature. Look at the frequency response and determine what kind of filter you have, i.e., HPF, LPF, or BPF.

Now, use the mouse to "grab" the pole and move it from $z = -0.9$ to $z = 0$ and then to $z = +0.9$. To move along the real axis, you can use `Options -> Move on Real Line` from the GUI menu. Observe how the frequency response changes. Describe the type of filter for $z = 0.9$, i.e., HPF, LPF, or BPF.

2.6.2 Create a Second-Order IIR BPF with PeZ

Use the **PeZ** interface to implement the following second-order system (with two poles):

$$H(z) = \frac{1 - z^{-2}}{1 + 0.8z^{-1} + 0.64z^{-2}}$$

Factor the numerator and denominator to get the two poles and the two zeros. Then place the poles and zeros at the correct locations in the z-plane with **PeZ**. If it is hard to place the poles and zeros precisely with the mouse, then use the `Edit` feature to enter the exact locations. Since **PeZ** wants to add complex-conjugate pairs, you should only have to add one of the poles; for the zeros, the `Add with Conjugate` feature should be turned off because you will be adding two real-valued zeros.
⇒ Look at the frequency response and verify that you have created a BPF.

2.7 Using Poles and Zeros to Create Filters that are Not Bandpass Filters

It is tempting to think that with two poles the frequency response always has a peak, but there are two interesting cases where that doesn't happen: (1) all-pass filters where $|H(e^{j\hat{\omega}})| =$ constant, and (2) IIR notch filters that null out one frequency, but have a relatively flat nonzero magnitude responses elsewhere.

• Implement the following second-order system:

$$H(z) = \frac{64 + 80z^{-1} + 100z^{-2}}{1 + 0.8z^{-1} + 0.64z^{-2}}$$

by factoring to get the two poles and the two zeros, and then placing the poles and zeros in **PeZ** at the correct locations in the z-plane. Adjust the gain (G) to get the correct numerator coefficients.
⇒ Look at the frequency response and determine what kind of filter you have.[3]

• Now, use the mouse to "grab" the zero-pair and move the zeros to be *exactly on the unit-circle at the same angle as the poles.* Observe how the frequency response changes. In addition, determine the $H(z)$ for this filter, which should have the following form:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + 0.8z^{-1} + 0.64z^{-2}}$$

⇒ Describe the type of filter that you have now created.

[3] The relationship between the poles and zeros of an all-pass filter is zero = 1/(pole)*; this situation where two poles and two zeros are linked together via conjugate-inverse symmetry can be done with the `PZ` option in **PeZ**.

3 In-Lab Exercise

The lab verification requires that you write down your observations on the verification sheet when using the PeZ GUI. These written observations will be graded.

3.1 IIR Allpass Filter

An IIR allpass filter will pass all frequency components with a magnitude response that is perfectly flat across the entire frequency band, but its phase response in nonlinear versus frequency. It has one complex-conjugate zero pair and one complex-conjugate pole pair.

$$\text{Zeros at } (1/r)e^{\pm j\theta} \qquad \text{Poles at } re^{\pm j\theta}$$

where r is a number slightly less than one, so that the poles are inside the UC and the system is *stable.* Thus the system function is:

$$H(z) = \frac{B(z)}{A(z)} = G\,\frac{(1 - r^{-1}e^{j\theta}z^{-1})(1 - r^{-1}e^{-j\theta}z^{-1})}{(1 - re^{j\theta}z^{-1})(1 - re^{-j\theta}z^{-1})} \tag{2}$$

You can use **PeZ** to exhibit a typical frequency response for an alllpass filter.

(a) To create an example in **PeZ**, use $r = 0.9$ and $\theta = \pi/3$ with the `PZ` option which automatically places the zeros at the conjugate-inverse locations. Alternatively, you can calculate the zeros, and then use the `Zero Location` edit window to enter the exact values.

(b) Adjust the gain (G) so that the frequency response magnitude is always equal to 10. Show the final **PeZ** result to your Lab instructor or TA.

(c) Write out the expression for the numerator and denominator of $H(z)$ created in part (a). You should create $H(z)$ by multiplying out first-order terms with the poles and zeros in (2).
Hint: use MATLAB's `poly` function to do the calculation. Also, the GUI provides $H(z)$, so you can check your work.

Instructor Verification (separate page)

3.2 IIR Notch Filters

A common heart test is an electrocardiogram (ECG or EKG) which records electrical activity that changes during the cardiac cycle. Metal electrodes placed at several locations on the body "pick up" these cardiac electrical signals. Unfortunately, the electrodes also pick up signals from other electrical sources, most notably harmonics of the 60-Hz power signal (or 50-Hz in some other countries). The objective of this short lab is to show that you can remove a sinusoidal interference from a corrupted ECG signal, and produce a cleaned-up signal. This will tap into your DSP skills as an accomplished "filter designer."

An IIR notch filter will null out one frequency, while having a frequency response that is relatively flat across the rest of the frequency band. It has one complex-conjugate zeros and poles:

$$\text{Zeros at } e^{\pm j\theta} \qquad \text{Poles at } re^{\pm j\theta}$$

where r is a number slightly less than one. Thus the system function is:

$$H(z) = \frac{B(z)}{A(z)} = G\,\frac{(1 - e^{j\theta}z^{-1})(1 - e^{-j\theta}z^{-1})}{(1 - re^{j\theta}z^{-1})(1 - re^{-j\theta}z^{-1})} \tag{3}$$

You can use **PeZ** to exhibit a typical frequency response for a notch filter. To see an example, select $r = 0.95$ and $\theta = \pi/4$. It is also possible to multiply out the numerator and denominator of (3) to get the filter coefficients. In MATLAB the function `poly` will create a (z-transform) polynomial from its roots.

3.3 IIR Filter Implementation

In MATLAB the function that does IIR filtering is called `filter`. It requires the numerator (num) and denominator (den) coefficients,

```
yy = filter( num, den, xx ).
```

For example, `hn = filter([1,2],[1,-0.9],[1,zeros(1,50)]);` will generate the impulse response of a first-order IIR filter. Use a stem plot to view `hn` versus the time index n.

3.4 Notch Filter Removes Sinusoidal Interference

The ECG signal (plus interference) will be generated by the MATLAB function `ECGmakeSig15.p`. The function `ECGmakeSig15.p` reads data from a file `ECGdataSig15.mat`, changes the sampling rate, and then adds a sinusoidal interference. These ECG signals were recorded in Spring-2006 by GT students Josh Hammel and Chris Clarke, see Fig. 2. The output of this function will depend on your GT login name—here is the calling template.

```
function [ecgsig,fs,fint] = ECGmakeSig15( gtstring, dur )
%
%   ecgsig = vector of signal samples at fs samples/sec
%     fint = frequency of the interfering sinusoid (near 50 or 60 Hz)
% gtstring = your gt login as a string, e.g., 'gburdell7' or 'gtg555q'
%      dur = duration of the signal (optional); default = 15 secs.
```

(a) Run `ECGmakeSig15` and note the values for the sampling rate (f_s) and interference frequency (f_{int}).

Instructor Verification (separate page)

(b) Design the IIR notch filter by specifying its poles and zeros. Explain how the angle (θ) of the poles and zeros is determined by the interference frequency (f_{int}) and the sampling rate (f_s). Draw the pole-zero diagram on the verification sheet.

(c) Compute the frequency response of the notch filter, and plot the frequency response magnitude versus ***analog frequency*** f in Hz. Make a sketch of what you see on the screen (no print out is needed). Verify that the notch filter will null out a sinusoid at the given interference frequency.

Instructor Verification (separate page)

(d) Convert the poles and zeros of the IIR notch filter to filter coefficients, so that you can write a difference equation for the filter.

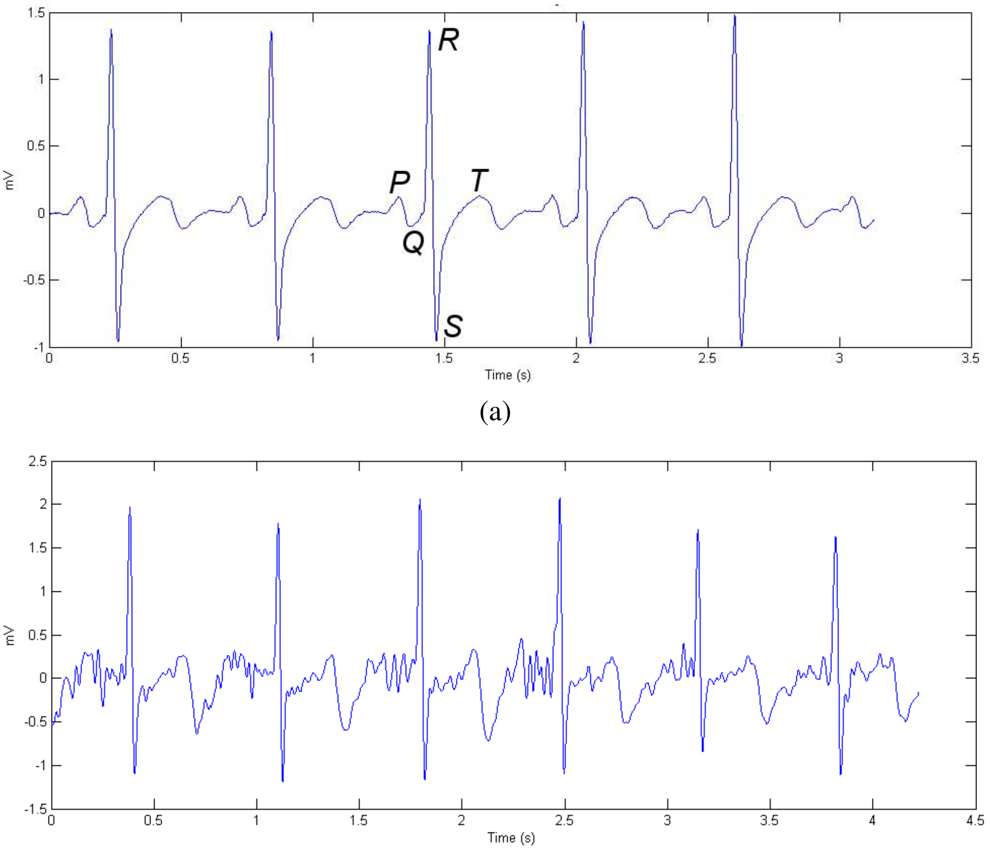
Instructor Verification (separate page)

(e) Apply the notch filter to the ECG signal and show that you can remove all the interference. Make a two-panel subplot comparing the time-domain signals *before* and *after* filtering. Show three or four periods of the ECG at the same time in both signals.

Instructor Verification (separate page)

(f) Show spectrograms of the signal before and after the notch filter. Point out where you see the interference in the input signal's spectrogram, and also where it has been removed in the output spectrogram.

Instructor Verification (separate page)



(a)

(b)

Figure 2: (a) The first ECG is very regular and follows the classic pattern. The P-wave, QRS complex, and T-wave are labeled. The P-wave results from depolarization of the sinoatrial (SA) node in the heart, leading to depolarization and contraction of the atrial cardiac muscle tissues. The QRS complex arises from depolarization of the atrioventricular (AV) node and resulting depolarization and contraction of the ventricular muscle tissue. The Q and S components of the complex are not always exhibited in a normal ECG. The T-wave results from repolarization of the ventricular tissue. (b) The second ECG is nonstandard. In place of the P and Q waves there is an irregular waveform. It appears that, in this case, the heart is in a constant state of atrial fibrillation. This normally is not a problem, as the atria contribute very little to the overall pumping ability of the heart, however it can present problems during strenuous exercise. (Recorded and described by Josh Hammel and Chris Clarke, April 2006.)

Lab #11
ECE-2026 Spring-2018
INSTRUCTOR VERIFICATION PAGE

Name: Date of Lab:

Complete Georgia Tech Course Evaluation surveys.

Verified: Date/Time:

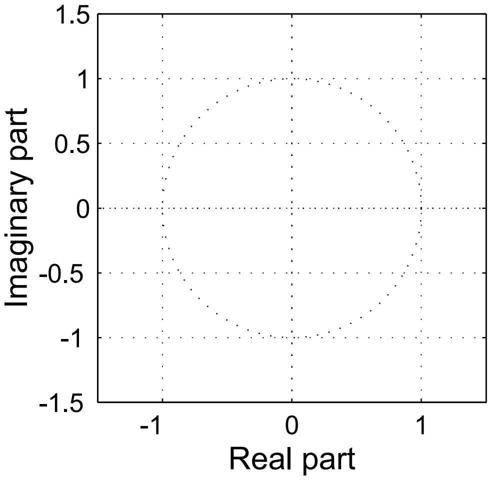
Part 3.1(a, b, c): Exhibit the allpass frequency response in **PeZ** and write out the numerator and denominator of the system function so that $|H(e^{j\hat{\omega}})| = 10$:

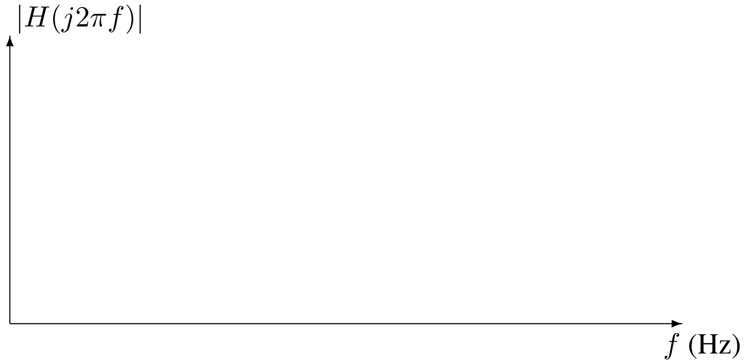
$$H(z) =$$

Verified: Date/Time:

Part 3.4(a): Record the values of the sampling rate and interference frequency.

$f_s = \boxed{}$ $f_{\text{int}} = \boxed{}$





Parts 3.4(b, c): Plot the poles and zeros of your notch filter on the z-plane grid above. In addition, sketch the frequency response in the space to the right of the pole-zero plot. Use a frequency range of 0 to $f_s/2$.

Verified: Date/Time:

Part 3.4(d): List the filter coefficients of the IIR filter by filling in the IIR difference equation below.

$$y[n] = \boxed{}\,y[n-1] + \boxed{}\,y[n-2] + \boxed{}\,x[n] + \boxed{}\,x[n-1] + \boxed{}\,x[n-2]$$

Part 3.4(e): Apply the notch filter to the ECG signal and show that you can remove all the interference. Show the two panel subplot that compares the time-domain signals *before* and *after* filtering. ***Zoom in to verify that all interference has been removed completely,*** except for a short duration at the beginning.

Verified: Date/Time:

Part 3.4(f): Show spectrograms of the signal before and after nulling and explain how the filter affects the frequency-domain characteristics of the signal.

Verified: Date/Time: