ECE2111 laboratory 4:
Electrocardiogram (ECG) Signal Filtering

Prepared by: Dr. Faezeh Marzbanrad

Aim

The aims of this lab are

1. to become familiar with the electrocardiogram (ECG) as a typical physiological signal.
2. to design an appropriate filters to remove noise and interferences.
3. to extract useful information such as the heart rate from ECG.

Introduction

This lab is about processing electrocardiograms (ECGs) to reduce contamination by noise and various other forms of interference. An ECG is a time-varying physiological signal which reflects the ionic current flow causing the cardiac (heart) fibers to contract and relax subsequently. It is obtained by recording the potential difference between two electrodes placed on the skin. ECG represents the successive atrial depolarization and repolarization as well as ventricular depolarization and repolarization occurring at every normal normal cycle of heartbeat. These events manifest as the peaks and troughs of the ECG waveform, namely P, Q, R, S, and T, shown in Figure 1. The most common application of an ECG is to estimate the heart rate. This is achieved by detecting R-peaks. Other applications include QT-interval and PR-interval measurement, and derivation of respiration rate from the ECG. The heart rate is usually estimated using R-R intervals, i.e., finding the time between successive R-peaks and inverting the interval lengths to find the instantaneous heart rate. This can also be averaged to estimate the mean heart rate. The variability of RR-intervals can also provide additional physiological information beyond the heart rate.

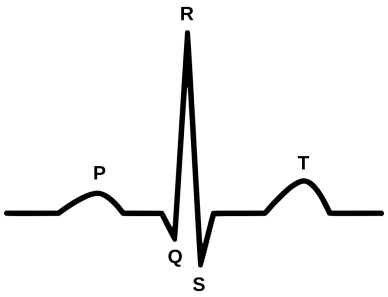

Figure 1: ECG waveform for a normal cardiac cycle.

To extract clinical information from a real (noisy) ECG requires reliable signal processing. There are various sources of noise and interference which contaminate an ECG, including power-line interference (a sinusoidal signal at 50 Hz or 60 Hz), low frequency electrode contact noise, baseline drift and motion artifacts, high frequency noise from electrical activity due to muscles around the chest wall, as well as various other types of instrumentation noise. Removal of these noise and interference signals is the first step in detecting the ECG peaks and extracting physiological information.

Scheduling

This lab runs over two weeks (weeks eight and nine) and is due at the end of the lab session in week 9.

Prelab

There are two prelab questions in these lab notes. By the start of week 8 (for the precise due date see Moodle), read through the lab document, find the prelab questions, and answer them.

Submit your answers to the prelab questions via the Moodle quiz called 'prelab 4' on the Moodle page. You are expected to do this individually.

Results document

You are required to organize your code and outputs in what we will call a "results document". This must be created following the guidelines in the accompanying file "Formatting requirements for lab results document". Submit this by the end of week 9 (for the precise due date see Moodle) via the 'results document' assignment link on Moodle. **Your submission must reflect your own work!**

End-of-lab quiz

There is a timed, end-of-lab quiz that must be completed by the end of week 9 (for the precise due date see Moodle). **Please do not start this quiz until you are ready.** This quiz tests your understanding of the lab material. **It must be completed individually.**

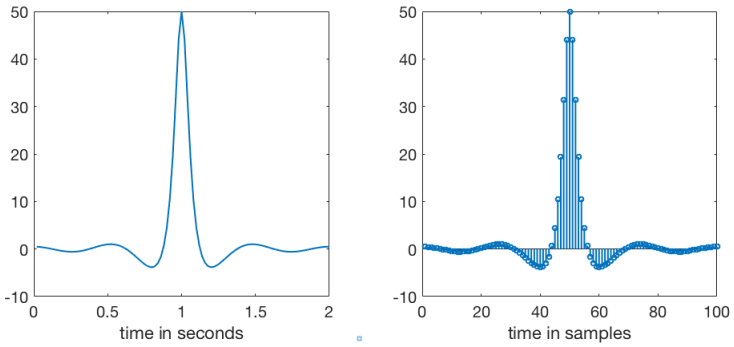

Figure 2: On the left is a plot of x vs time. On the right is a plot of x_s vs time.

Plotting signals in the frequency domain

In this preliminary section we describe different ways to make frequency domain plots of a discrete-time finite duration signal. We illustrate these with the example of the continuous-time signal

$$x(t) = \frac{\cos(4\pi t)}{1.1 + \cos(\pi t)} \quad \text{for all } t$$

with fundamental period 2 seconds. If we sample this signal with sampling rate 50 samples/second, we obtain

$$x_s[n] = \frac{\cos(\pi n/12.5)}{1.1 + \cos(\pi n/50)} \quad \text{for all } n.$$

Plots of x and x_s over one period are shown in Figure 2.

Plotting the discrete-time Fourier series coefficients In lab 3, when we plotted a frequency domain representation of a discrete-time signal, we plotted the coefficients of its discrete-time Fourier series decomposition. We used a stem plot because these are indexed by integers. We did this using the code:

```
N = length(x);
X = (1/N)*fft(x);
kX = -floor(N/2):(N-1-floor(N/2));
figure(1); stem(kX, fftshift(abs(X)));
figure(2); stem(kX, fftshift(angle(X)));
```

which plots the magnitude and phase of the Fourier series coefficients from $-\lfloor N_0/2 \rfloor$ to $N_0 - 1 - \lfloor N_0/2 \rfloor$. For our example we obtain the plot in Figure 3.

Plotting a representation of the discrete-time Fourier transform Another representation that is quite useful is to rescale the horizontal axis by the fundamental frequency $\omega_0 = 2\pi/N_0$. This makes sense becuase the kth Fourier series coefficient corresponds to a complex exponential at frequency $k\omega_0$. In this case the horizontal axis has units of radians/sample. We often also scale the vertical axis by N_0, and use `plot` rather than `stem`. This is common because it is an approximation of the discrete-time Fourier transform of the signal (we discuss

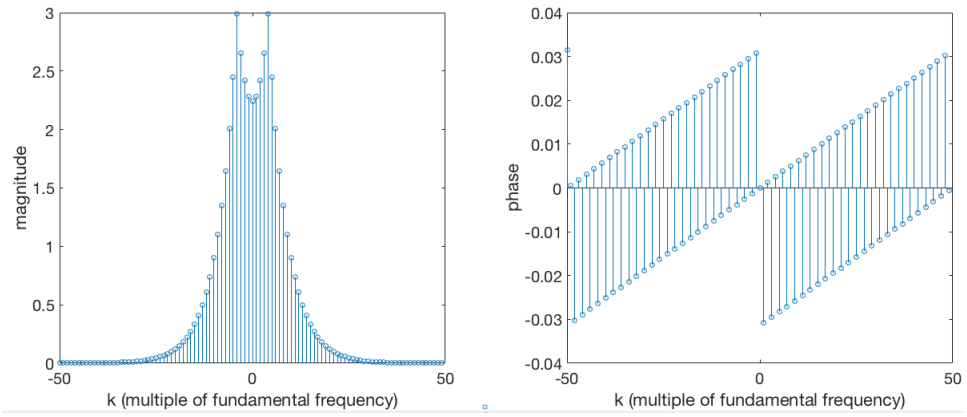

Figure 3: The magnitude (left) and phase (right) of the discrete-time Fourier series coefficients of x_s.

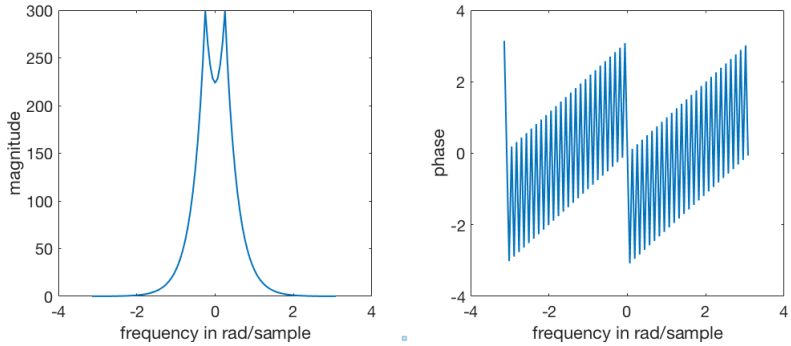

Figure 4: A continuous interpolation of the magnitude (left) and phase (right) of the discrete-time Fourier series coefficients of x_s.

this representation see topic 8). We can construct a plot like this using the code:

```
N = length(x);
X = fft(x);
omega = (-floor(N/2):(N-1-floor(N/2)))*(2*pi/N);
figure(1); plot(omega, fftshift(abs(X)));
figure(2); plot(omega, fftshift(angle(X)));
```

For our example we obtain the plot in Figure 4. This plot still does not take into account the sampling frequency, and has no connection to the continuous-time world.

Rescaling to use continuous frequency units Another way to represent the same data is to rescale the plot so that the horizontal axis is labeled by continuous frequency (say in Hz). To figure out the right rescaling we can look at the units. If we have data in units of rad/sample, then to get data with units rad/second we need to multiply by the sampling rate in samples/second. We can then convert into Hz by dividing by 2π (since there are 2π radians in a circle). This sort of frequency axis rescaling is useful for interpretation because we are often thinking about frequency content in the original continuous-time signal that we have sampled. The following code makes a plot with this frequency rescaling to have frequency units Hz. The

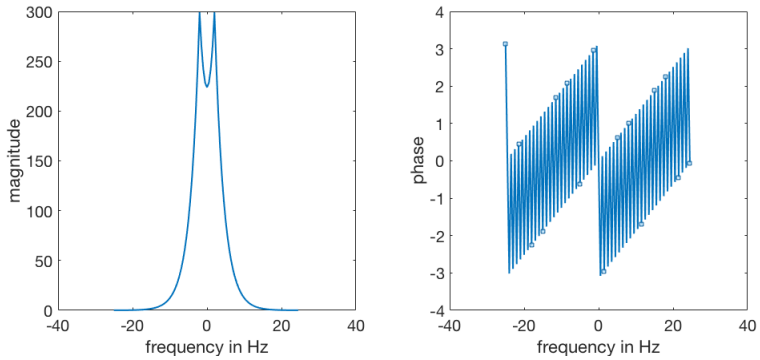

Figure 5: A continuous interpolation of the magnitude (left) and phase (right) of the discrete-time Fourier series coefficients of x_s. The frequency axis is scaled to correspond to Hz.

variable `Fs` is the sampling rate in samples/second.

```
N = length(x);
X = fft(x);
omega = (-floor(N/2):(N-1-floor(N/2)))*(Fs/N);
figure(1); plot(omega, fftshift(abs(X)));
figure(2); plot(omega, fftshift(angle(X)));
```

For our example we obtain the plot in Figure 5. All of these representations are useful. Still, please be aware that both the interpolation (using `plot` not `stem`) and the axis rescaling can be misleading. In topic 10, about sampling and reconstruction, we will discuss how going from continuous-time to discrete-time affects the frequency domain representation of the signal and how to understand more carefully what we are doing here.

1 Identifying the ECG and noise frequencies

In this part of the lab you will take a synthetically generated (by the method described in [1]) yet realistic ECG signal and identify the frequencies which correspond to the ECG, the interferences, and the noise. While the signal of interest is a discrete-time signal, it is obtained by sampling a continuous-time signal. This means that it is easier to think about the frequency content of the signal in terms of continuous frequency measures (like Hz).

1.1 Activities

Write a MATLAB script to carry out the following tasks.

1. Load the signal `ECG.mat` using the `load` command. The signal is sampled at 256 samples/second.

2. (a) Plot the magnitude of the discrete-time Fourier series coefficients of the signal. Label the axes, and save the plot with a sensible filename.

   (b) On a separate figure, plot a representation of the discrete-time Fourier transform of the signal (i.e., use the second plotting option described above). Again label the axes and save the plot.

   (c) On a separate figure, plot a representation of the Fourier transform of a continuous-time version of the signal (i.e., use the third plotting option above). Again label the axes and save the plot.

3. Identify the power-line frequency in rad/sample and in Hz.

4. Plot the signal in time domain over an interval of time that shows the features of the signal well. Use `plot` rather than `stem` since we are thinking of this as a continuous-time signal. The horizontal axis should be in seconds (rather than samples). Label the axes, and save the plot with a sensible filename.

5. How have the low-frequency noise and power-line interference contaminated the ECG, based on the time-domain plot?

Once you have finished this section:

- copy your code into your results document
- copy your plots from item 2 and 4 into your results document.
- Answer the questions in items 3 and 5 in your results document.

2 Removing power-line interference from the ECG

In this part you will design an FIR filter to remove the power-line interference from the ECG. You will use the function `firpm`. Look in the lecture notes and slides for more information about using `firpm`. Also read the MATLAB documentation.

2.1 Prelab questions

Suppose we want to design a notch filter (narrowband bandstop filter) with stopband centered at 60Hz (note that this is not quite the frequency of interest for the lab). If the sampling frequency is 256 samples/second:

1. What frequency in rad/sample should correspond to the centre of the stopband? (Answer correct to two decimal places.)
2. If we write this frequency in the form $f_c \times \pi$ rad/sample, what is the value of f_c? (Answer correct to two decimal places.)

(Note that when we use `firpm` we need to specify the band edge frequency values as a fraction of π rad/sample, as in question 2 above.)

2.2 Activities

1. Note the frequency of the power-line interference in rad/sample. Use `firpm` to design a notch (narrow band stop) filter to remove it. Try three different cut-off frequencies around the power-line frequency as well as three different filter lengths (try 5, 10, and 20).
2. For each choice of parameters (cutoff frequencies and filter lengths),
   (a) Apply the filter to your signal using the `conv` function in MATLAB.
   (b) On the same set of axes, plot the signal in the frequency domain before and after filtering, label the axes, and save the plot with a sensible filename.
   (c) How does filtering affect the signal in the frequency domain? What is the effect of changing the filter length?
   (d) On the same set of axes, plot the signal in the time domain before and after filtering, label the axes, and save the plot with a sensible filename. Restrict your plot to an interval of time that contains a few heart beats, so it is easy to compare the original and filtered signals.
   (e) How does filtering affect the signal in the time domain?
   (f) Calculate the time shift of the output signal with respect to the input signal of each filter. Give your answer in seconds and in samples. (Hint: look at the time domain plots.)
3. How does the filter length affect the time delay that you observe? Why do you think this delay occurs?

Once you have finished this section:

- copy your code into your results document
- copy your plots (from 2(b), 2(d)) into your results document.
- Answer the questions in items 2(c), 2(e), 2(f), and 3 in your results document.

3 Removing baseline wandering and motion artefacts from the ECG

In this part you will design an FIR filter to remove very low frequency components (due to baseline wandering and motion) from the ECG.

3.1 Activities

1. Design and apply a high-pass filter to remove any content below 1 Hz. What might be an appropriate choice of cut-off frequency in cycles per second? What does this correspond to in radians/sample? Try filters of length 100, 200, and 500.

2. For each choice of filter length,

   (a) Apply the filter using the `conv` function in MATLAB.

   (b) Plot the signal in the frequency domain before and after filtering (on the same set of axes), label the axes, and save the plot with a sensible filename.

   (c) How does filtering affect the signal in the frequency domain?

   (d) Plot the signal in the time domain (over a short segment that displays its features well) before and after filtering (on the same set of axes), label the axes, and save the plot with a sensible filename.

   (e) How does filtering affect the signal in the time domain?

Once you have finished this section:

- copy your code into your results document
- copy your plots (from 2(b), 2(d)) into your results document.
- Answer the questions in items 2(c) and 2(e) in your results document.

4 Removing high-frequency noise and interference from the ECG

In this part you will design an FIR filter to remove high frequency components (usually due to noise and unwanted interference) from the ECG.

4.1 Activities

1. Design and apply a low-pass filter to remove any high frequency noise above 40 Hz. Try filters of length 50, 100, and 200.
2. For each filter length,
   (a) Apply the filter using the `conv` function in MATLAB.
   (b) Plot the signal in the frequency domain before and after filtering (on the same set of axes), label the axes, and save the plot with a sensible filename.
   (c) How does filtering affect the signal in the frequency domain?
   (d) Plot the signal in the time domain (over a short segment that displays its features well) before and after filtering (on the same set of axes), label the axes, and save the plot with a sensible filename.
   (e) How does filtering affect the signal in the time domain?

Once you have finished this section:

- copy your code into your results document
- copy your plots (from 2(b), 2(d)) into your results document.
- Answer the questions in items 2(c) and 2(e) in your results document.

5 Estimating the heart rate

In this part of the lab you will apply all three of the filters you designed in Sections 2–4. You will then find the ECG R-peaks in the resulting signal to estimate the heart rate.

5.1 Activities

1. Use your favourite choice of parameters for each of the filters you designed in Sections 2–4. Apply all three filters to your signal, one after the other (in other words, apply the series interconnection of the three systems).

2. Use the function `findpeaks` (read the MATLAB documentation) to find the peaks of the filtered ECG.

3. Find an appropriate threshold so that you select only the peaks with correspond to R-peaks (see figure 1).

4. Plot the filtered ECG in the time-domain over an short segment that shows the features of the ECG. Mark the location of identified R-peaks.

5. Calculate the sequence of time intervals between successive R-peaks (called RR intervals). The function `diff` may be useful. Do this both for time in samples as well time in seconds.

6. Each R-peak corresponds to one heartbeat. Use the RR intervals to calculate the average heart rate in beats per minute (bpm).

7. What average heart rate do you find? How did you compute it?

Once you have finished this section:

- copy your script into your results document
- copy your plot from 4 into your results document.
- Answer the questions in item 7 in your results document.

Once you have completed the lab tasks and understand them, you are ready for the end-of-lab Moodle quiz. This quiz is closely based on the lab tasks. It must be completed **individually**.You may use MATLAB. Unlike the prelab, you only have **one attempt at each question**.

Assessment

This lab is marked out of 12. Your mark is based on the following:

Prelab Correct responses to the two prelab questions submitted via the prelab Moodle quiz (4 marks, 2 per question)

- **Results document:** (4 marks, 3 marks for content and 1 mark for presentation)
  Marks per section: Each of the 5 sections is marked out of 0.6 (3 marks):
  - 0 marks if not attempted
  - 0.3 marks if a reasonable attempt is made, but has clear flaws
  - 0.6 mark if no errors or possibly very minor errors

  Presentation: (1 mark)
  - Results document adheres to the formatting requirements (1 mark)
  - Results document mostly adheres to formatting requirements, but not completely (0.5 mark)
  - Results document rarely adheres to formatting requirements (0 marks)
- **End-of-lab quiz:** Correct responses to the end-of-lab Moodle questions (4 marks, 1 per question).

References

1. McSharry, Patrick E., et al. "A dynamical model for generating synthetic electrocardiogram signals." IEEE Transactions on Biomedical Engineering 50.3 (2003): 289-294.