# MATLAB Assignment 1

## Digital filters

- **This is an individual assignment.**
- **Use data files (.mat) that are attached as instructed throughout the assignment.**
- **Submission guidelines**

| Submission document | Submission method | Notes |
|---|---|---|
| Report | Upload the softcopy to Moodle | Should include observations and discussions with relevant plots to support your answers. |
| MATLAB scripts | Upload a single ZIP file including all the .m files to Moodle | Name each script according to the question number. Also, include necessary comments on the scripts for better read-ability. |

## Introduction

This assignment allows implementing digital filters on MATLAB and applying them to biomedical signals. The student is expected to generate and analyse the results and to get familiarised with the practical limitations of implemented digital filters.

### 1. Smoothing filters

One of the most common signal processing tasks is smoothing of the data to reduce high frequency noise arising from electromagnetic interferences, quantization errors and from peripheral physiological signals. Here, the application of moving average filters of order *N* (*MA(N)*) and Savitzky-Golay filters of order *N* and length *L'=2L+1 (SG(N,L))*: refer the lecture note for nomenclature.

#### 1.1. Moving average *MA(N)* filter

A *MA(N)* filter can be visualised as a moving window across the signal. A point of the filtered signal $y(n)$ is derived from the average of the windowed points on the input signal $x(n)$.

$$y(n) = \frac{1}{N}\sum_{k=0}^{N-1} x(n-k)$$

*Equation 1*

Preliminaries

i) Load `ECG_template.mat` acquisition parameters of this signal are as shown in Table 1.

| | |
|---|---|
| Sampling frequency ($f_s$) | 500 Hz |
| Amplitude range | mV |

*Table 1*

ii) Plot the loaded signal with the adjusted time scale (according to the $f_s$). Observe typical ECG characteristics.

iii) Add white Gaussian noise of 5 dB using the following command (name the noise added signal `nECG`). `awgn(x,snr,'measured')`

iv) Plot the power spectral density (PSD) estimate of the nECG using the following command. `periodogram(x,window,nfft,fs)`

*MA(3)* filter implementation with a customised script

i) Write a MATLAB script for a *MA(3)* filter formulated in Equation 1. Name the filtered signal `ma3ECG_1`.

ii) Derive the group delay.

iii) Plot the delay compensated `ma3ECG_1`and `nECG` on the same plot and compare.

iv) Produce overlapping PSDs of `ma3ECG_1`and `nECG` and compared the filter effect.

*MA(3)* filter implementation with the MATLAB built-in function

i) Use the `filter(b,a,X)` command to filter the `nECG` signal with a *MA(3)* filter while compensating for the group delay. Name the filtered signal as `ma3ECG_2`.

ii) Plot `nECG, ECG_template and ma3ECG_2` on the same plot and compare.

iii) Use the `fvtool(b,a)` to inspect the magnitude response, phase response and the pole-zero plot of the *MA(3)* filter.

*MA(10)* filter implementation with the MATLAB built-in function

i) Identify the improvement of the *MA(10)* filter over the *MA(3)* filter using the `fvtool(b,a)`.

ii) Filter the nECG signal using the above *MA(10)* filter while compensating for the group delay. Name the filtered signal `ma10ECG`.

iii) Plot `nECG, ECG_template, ma3ECG_2` and `ma10ECG` on the same plot to compare the improvement.

Optimum *MA(N)* filter order

i) Write a MATLAB script to calculate the mean-squared-error (MSE) between the noise free signal and the filtered signal with the *MA(N)* filter.

ii) Hence test for a rage of *N* values and determine the optimum filter order which gives the minimum MSE by plotting MSE vs. *N.*

iii) Suggest the reason/s for large MSE values at low and high filter orders.

### 1.2. Savitzky-Golay *SG(N,L)* filter

Savitzky-Golay filter fits a polynomial of order *N* to an odd number of data points *L'=2L+1(where L' is an odd integer)* in predefined window in a least-squares sense. A unique solution requires $N \leq L'-1$.

Application of *SG(N,L)*

i) Apply a *SG(3,11)* filter on the `nECG` signal while compensating the group delay. Name the filtered signal as `sg310ECG`. Use the following command. `sgolayfilt(x,N,L')`
ii) Plot `nECG`, `ECG_template` and `sg310ECG` on the same plot and compare.

Optimum *SG(N,L)* filter parameters

i) Calculate the MSE values for a range of parameters of the *SG(N,L)* filter and determine the optimum filter parameters which gives the minimum MSE. (Hint: use `surf(·)` or `pcolor(·)`)
ii) Plot `ECG_template`, `sg310ECG` and the signal obtained from optimum *SG(N,L)* filter on the same plot to compare the improvement.
iii) Compare signal features and computational complexity of the optimum filtered signals derived from *MA(N)* and *SG(N,L)* filters.

## 2. Ensemble averaging

In the case of overlapping signal and noise spectra, the synchronized averaging technique is an effective and a simple method for noise removal with minimal signal distortions. However, for synchronized averaging to be applicable, there should be input data either having multiple measurements (e.g. EPs) or one signal having repetitive patterns (e.g. ECG).

### 2.1. Signal with multiple measurements

Consider the signal with multiple measurements to be the Auditory Brainstem Response (ABR) which is the early part of the auditory evoked potential. The ABR is generated as a response to an auditory stimulus and recorded from EEG electrodes placed on the scalp. Multiple ABRs are recorded by multiple stimulations given at a constant rate. The amplitude of the ABR is in the range of µV which is over-shadowed by the ongoing EEG noise which is in the range of mV. This section uses a recorded ABR pulse train (over 1000 ABRs) having properties given in Table 2 and removes underlying EEG noise using ensemble averaging.

| | |
|---|---|
| Name of the recording | ABR_rec.mat |
| Sampling frequency | 40 kHz |
| Amplitude range | mV |
| Duration of the ABR | 10ms from the point of stimulation |
| Other notes | ABR_rec column 1 = auditory pulse stimulus (0.1ms, f=21.1Hz)<br>ABR_ rec column 2 = ABR+EEG |

*Table 2*

Preliminaries

i) Clear the workspace and the command window.

```
clear all; clc;
```

ii) Load `ABR_rec.mat` to MATLAB workspace.

```
load ABR_rec.mat;
```

iii) Plot the train of stimuli and ABRs on a single plot and observe; Amplitudes and the waveforms

```
figure, plot(ABR_rec), legend('Stimuli','ABR train');
```

iv) Determine a voltage threshold (e.g. 50) to automatically detect the likely stimuli occurrences.

```
thresh = find(ABR_rec(:,1)>50);
```

v) Extract actual stimulus points.

```
j=1;
for i=1:length(thresh)-1
  if thresh(i+1)-thresh(i)>1; stim_point(j,1)=thresh(i+1); j=j+1;
  end
end
```

vi) Window ABR epochs according to the extracted stimulus points. Consider the window length to be 12ms (-2ms to 10ms) with reference to the stimulus time point (i.e. -80 to 399 sample points at $f_s$=40 kHz).

```
j = 0;
for   i=1:length(stim_point) j = j + 1;
epochs(:,j) = ABR_rec((stim_point(i)-80:stim_point(i)+399),2);
end
```

vii) Calculate the ensemble average of all the extracted epochs.

```
ensmbl_avg = mean(epochs(:,(1:length(stim_point))),2);
```

viii) Plot the ensemble averaged ABR waveform.

```
figure,
plot((-80:399)/40,ensmbl_avg)
xlabel('Time (ms)'), ylabel('Voltage(uV)')
title([Ensemble averaged ABR from ',num2str(length(epochs)),'
epochs'])
```

ix) The plot should be similar to Figure 1 except labelling.

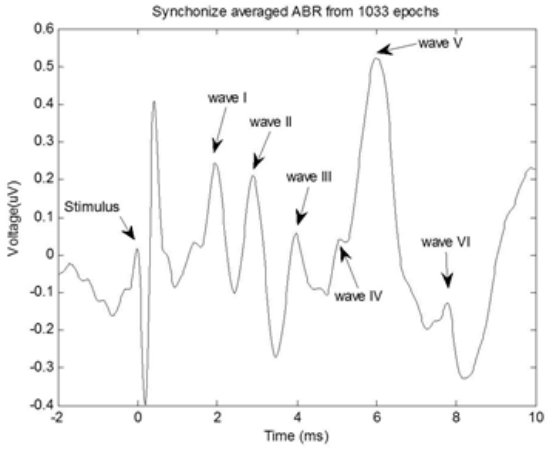


*Figure 1*

Improvement of the SNR

i) Using the ensemble averaged ABR (`ensmbl_avg`) as the template, write a MATLAB scrip to calculate an array containing progressive MSEs as per the following equation.

$$MSE_k = \sqrt{\frac{\sum_{n=1}^{N}(\hat{y}(n) - \tilde{y}_k(n))^2}{N}}, \quad k = 1,2, \dots, M$$

Where,
$\hat{y}(n)$: template
$N$: length of the a single ABR epoch
$M$: epochs in the ensemble average
$\tilde{y}_k(n)$: progressive average of single epochs defined as follows;
$\tilde{y}_1(n) = x_1(n)$
$\tilde{y}_2(n) = \frac{x_1(n) + x_2(n)}{2}$
$\tilde{y}_M(n) = \frac{x_1(n) + x_2(n) + \cdots + x_M(n)}{M}$
Where $x_M(n)$ is a single ABR epoch.

ii) Plot a graph of $\mathrm{MSE_k}$ against $\mathrm{k}$ and describe the behaviour with reference to the theoretically derived improvement of the SNR.

### 2.2. Signal with repetitive patterns

Consider an almost periodic recording of an ECG pulse train having acquisition parameters given in Table 3. The task in this section is to add Gaussian white noise to this signal to emulate practical conditions and extract a single denoised ECG pulse using ensemble averaging. Here, the usage of ensemble averaging is justified since the ECG spectrum inevitably overlaps with white noise.

| | |
|---|---|
| Name | ECG_rec.mat |
| Recorded lead | II |
| Sampling frequency | 128 Hz |
| Amplitude range | mV |

*Table 3*

Viewing the signal and addition Gaussian white noise

i) Clear the workspace and the command window and load `ECG_rec.mat` to MATLAB workspace.

ii) Plot the data and observe amplitudes and the waveforms. The scale and labelling should be similar to Figure 2.

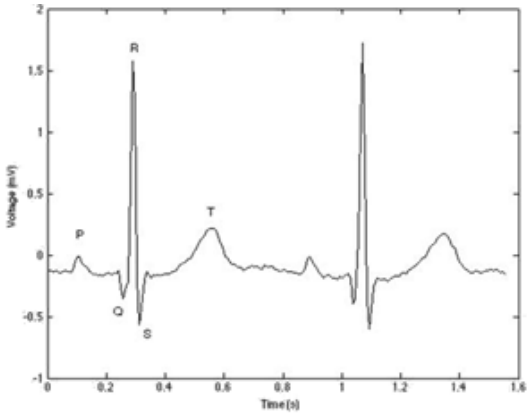


*Figure 2*

iii) Extract a single PQRST waveform and define it as the `ECG_template`.
iv) Add Gaussian white noise of 5 dB to the `ECG_rec` and name it as `nECG`.

Segmenting ECG into separate epochs and ensemble averaging

Given that an ECG template already exists, it is a better method to use points of maximum correlation with noisy ECG pulse train rather than merely detecting the R-wave to segment the ECG pulse train into separate epochs.

i) Calculate the normalised cross-correlation between the `ECG_template` and the `nECG`. Use either `xcorr(x,y)` with modifications or implement the following equation.

$$\Theta_{xy}(k) = \frac{\sum_{n=1}^{N}([x(n) - \bar{x}][y(n-k) - \bar{y}])}{\sqrt{\sum_{n=1}^{N}(x(n) - \bar{x})^2 \sum_{n=1}^{N}(y(n-k) - \bar{y})^2}}$$

Where
$x(\cdot)$: `nECG`, $y(\cdot)$: `ECG_template`, $\bar{x}$, $\bar{y}$: mean of `nECG` and `ECG_template`, $k$: lag, $N$: length of the template

ii) Plot the normalised cross-correlation values against the adjusted lag axis converted to time axis.
iii) Segment ECG pulses by defining a threshold and store in a separate matrix.
iv) Calculate and plot the improvement in SNR as the number of ECG pulses included in the ensemble average is increased.
v) Plot and compare (in the one graph), a selected noisy ECG pulse and two arbitrary selected ensemble averaged ECG pulses.
vi) Suggest a justification to the claim *'it is a better method to use points of maximum correlation with noisy ECG pulse train rather than merely detecting the R-wave to segment the ECG pulse train into separate epochs'.*

## 3. FIR derivative filters

### 3.1. FIR derivative filter properties (use the `fvtool(b,a)`)

i) Plot and observe the;
- pole-zero plot
- magnitude response plot in linear scale
- magnitude response plot in logarithmic scale

of first order and three-point central difference derivative filters as derived in slide '*First and second order derivative filters'*.

ii) What are the multiplying factors (G) that should be substituted (such that $G.H(z)$) to make the maximum gain of these filters to be unity (0 dB)? What's the importance of this?

### 3.2. FIR derivative filter application

In this section, noise will be artificially added to an ECG pulse train and observe the result when above derivative filters are applied on them.

i) Clear the workspace and the command window and load `ECG_rec.mat` to MATLAB workspace.

ii) Add following noise components to the `ECG_rec` and name it as `nECG`. `nECG` should be similar to Figure 3.

- Gaussian white noise of 10 dB
- $EMG(t) = 2\sin\left(\frac{2\pi}{4}t\right) + 3\sin\left(\frac{2\pi}{2}t + \frac{\pi}{4}\right)$ representing low frequency muscle artifacts

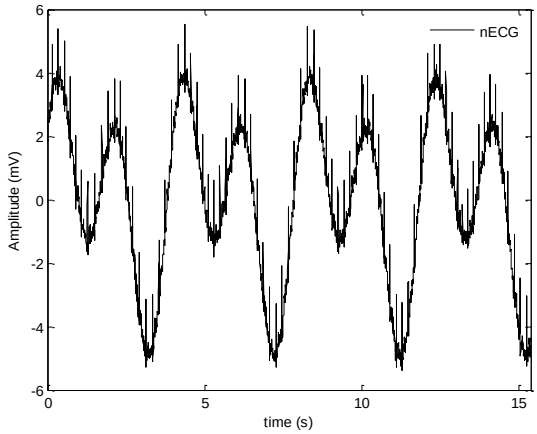


*Figure 3*

iii) Apply the derivative filters derived in the previous question on the `nECG`.

iv) Plot and analyse the results.

## 4. Designing FIR filters using windows

### 4.1. Characteristics of window functions (use the `fdatool`)

i) Rectangular window function is the most simplest. Explain the effect of the length of this window function (*M*) using overlaying plots of the impulse response for $\boldsymbol{M = 5, 50, 100}$.

ii) For a lowpass filter having a cut off frequency at $\omega_c = 0.4\pi$ (normalised frequency), plot the comparative magnitude response with a linear magnitude scale and their comparative phase response plot for $M = 5, 50, 100$. (for example, refer the slide '*Rectangular window function*')

iii) To improve the magnitude response, smoother windows have been designed removing discontinuities of the rectangular window. Plot and explain the following comparative characteristics of window functions; rectangular, Hanning, Hamming and Blackman of length *M*=50.

- Morphology of the window
- Magnitude response with a linear magnitude scale
- Magnitude response with a logarithmic magnitude scale
- Phase response

### 4.2. FIR Filter design and application using the Kaiser window

The Kaiser window is a generic function which can approximate a variety of windows by varying the shaping parameter $\boldsymbol{\beta}$ and the window length *M*. In this section, a lowpass and a highpass filter using the Kaiser window and a FIR comb filter should be designed to filter out noise embedded in an ECG signal of which the information is given in Table 4.

| | |
|---|---|
| Name | ECG_with_noise.mat |
| Recorded lead | II |
| Sampling frequency | 500 Hz |
| Amplitude range | mV |

*Table 4*

i) Plot the time domain signal and the power spectral density (PSD) estimate (or in other words frequency spectrum) of the above signal using the following command. Identify its frequency content with respect to a bandwidth of a normal ECG.
`periodogram(x,window,nfft,fs).`

ii) Decide on parameters for the following filters that need to be applied on the above ECG. For a guide use the highpass cut off frequency as 5 Hz and lowpass cut off frequency as 125 Hz.

| | Highpass | Lowpass |
|---|---|---|
| $f_{pass}$ | | |
| $f_{stop}$ | | |
| $\delta$ | | |

| | Comb |
|---|---|
| $f_{stop}\,1$ | |
| $f_{stop}\,2$ | |
| $f_{stop}\,3$ | |

iii) Calculate the relevant $\beta$ and *M* values for the highpass and lowpass filters.

iv) Visualise the windows (highpass and lowpass), magnitude response and the phase response of above filters.

v) Apply these filters one by one to the ECG_with_noise.mat signal with necessary compensations for the group delay. Plot the effects in the time domain.

vi) Plot the magnitude response of the combined three filters and the PSD of the final filtered ECG that was derived in the previous question. (refer the slide '*Overall filtering process applied on an ECG'*)

## 5. IIR filters

This section explores the realisation of IIR Butterworth filters, effect of the non-linear phase response, forward-backward filtering and a comparison to FIR filters implemented in the previous section. Here, use MATLAB functions to calculate IIR filter parameters NOT the `fdatool`.

### 5.1. Realising IIR filters

i) Obtain filter coefficients of a Butterworth lowpass filter with the same cut off frequency and of the same order that was used to implement the FIR lowpass filter in the previous section. Use the MATLAB command `[b,a]=butter(n,Wn)`.

ii) Visualise the magnitude response, phase response and the group delay using `fvtool(b,a)`.

iii) Obtain the filter coefficients of the highpass and the comb filter and visualise them (use `[b,a]=iircomb(n,bw)` to obtain the comb filter coefficients).

iv) Plot the magnitude response of the combined three filters and compare it with the corresponding FIR filter plot.

### 5.2. Filtering methods using IIR filters

i) To apply forward filtering, use the `filter(b,a,x)` command to filter the ECG_with_noise.mat signal with the IIR filters realised in the previous section.

ii) To apply forward-backward filtering use the `filtfilt(b,a,x)` command to filter the ECG_with_noise.mat signal with the IIR filters realised in the previous section.

iii) Generate an overlapping time domain plots of the FIR filtered ECG, IIR forward filtered ECG and IIR forward-backward filtered ECG. Compare the effect with respect to theoretical interpretations. (zoom-in to include only two ECG pulses to improve clarity)

iv) Generate overlapping plots of the PSDs of the FIR filtered ECG, IIR forward filtered ECG and IIR forward-backward filtered ECG and compare with respect to theoretical interpretations.