Lab 8

Discrete Fourier Transform Applications

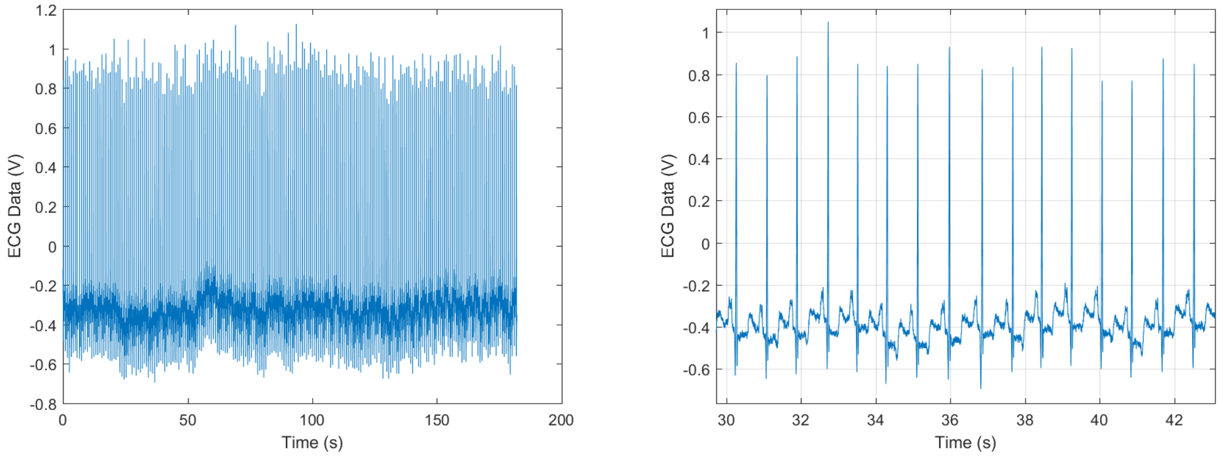
Filtering with the FFT/IFFT (4 points)

Matlab filename must be `exer01.m`.

An electrocardiogram signal is given in file `ecg.mat`. In order to load the data you must do the following in Matlab's command line:

```
          >> load ecg.mat;
```

After loading the file you have two new signals in your workspace, `sig` (the electrocardiogram data) and `tm` (the time vector). The sampling frequency used is 360 Hz. The signal is a 3 minutes recording of a ECG (shown in figure below.)



We need to remove several noisy elements of this signal. The first one in the low frequency components that can be seen in the left-hand side figure.

1. Using the FFT and the IFFT, write a Matlab script that removes from the initial signal frequency components below 0.5Hz.
2. You might have notice from the ECG spectrum that there are also an important frequency component at 60Hz and its harmonics (120Hz and 180Hz.) Show the presence of these components in the signal spectrum
3. Remove from the signal the frequency components in the vicinity of 60, 120 and 180 Hz.

Data Compression with FFT (4 points)

Matlab filename must be `exer02.m`.

One way to compress data is to calculate its Fourier coefficients and keep only those that are sufficiently large to ensure a good reconstruction of the original signal. This type of compression is called lossy because the uncompressed signal is different from the original which makes it suitable for audio and video compression. The audio data file can be downloaded in the lab assignment section of ADI. Divide the signal into non-overlapping sections of 8192 points:

1. Write an algorithm that determines the number of frequency components that needs to be kept in order to have 99.9% of the original signal.
2. Regenerate the signal using the Fourier components found above and plot the result along with the original signal.
3. Determine the compression ratio (defined as the ratio between the original size and the compressed size) obtained by using just the frequency components found above.

Note: To load the audio file use the following syntax in Matlab:

```
[y,fm] = audioread('audio_file.wav');
```