Assignment 2

Digital Signal processing

FIR filters

Bernd Porr, Nick Bailey

Form groups of 2 people.

The task of this assignment is to filter an ECG with FIR filters and to detect the R peaks. In contrast to the FFT assignment we write filter code which can be used for *realtime processing*. This means that the FIR filter needs to be implemented with the help of delay lines and the impulse response is truncated.

ECG filtering

Record an ECG from one team member (see instructions on moodle how to record it and complying with the ethics by signing the consent sheet) as little artefacts as possible: lie down and for best results place the electrodes on the shoulders and belly as close to the hips. This avoids the large muscles such as hip muscle and biceps.

1. Create a function which calculates and returns the FIR filter coefficients *numerically* (= using python's IFFT command) for a *combined* highpass and bandstop filter.
   The function should automatically decide how many coefficients are required. The function arguments should be a) the sampling rate and b) the cutoff frequencies. Decide which cutoff frequencies are needed and provide explanations by referring to the spectra and/or fundamental frequencies of the ECG. **[25%]**

2. Create an efficient Python FIR filter class which implements an FIR filter which has a method of the form value dofilter(self,value) where both the value argument and return value are *scalars* and not vectors (!) so that it can be used in a realtime system. The constructor of the class takes the coefficients as its input:

```
class FIRfilter:
def __init__(self,_coefficients):
# your code here
def dofilter(self,v):
# your code here
return result
```

Filter your ECG with the above FIR filter class using the coefficients from 1. Simulate realtime processing by feeding the ECG sample by sample into your FIR filter class. Make sure that the ECG looks intact and that it is not distorted (PQRST intact). Provide appropriate plots. **[15%]**

3. Instead of using a bandstop filter to remove the 50Hz and DC use an adaptive LMS filter by providing it with a 50Hz sine wave with DC as reference. Add an adaptive LMS filter command to your FIR filter class and name it: “doFilterAdaptive(self,signal,noise,learningRate)” which returns the cleaned up ECG. As before also this function must receive only scalars (i.e. sample by sample) and return a scalar. Plot and compare the result from the adaptive filter and that from the FIR filter design. Optimise the learning rate by ensuring stability, high speed of convergence and minimal ECG distortion. Justify your design decisions in the report. **[35%]**

4. ECG heartbeat detection: Record a *2nd ECG* from the other team member which is noisy, while standing and having the electrodes on the wrists & ankles. The task is to detect R-peaks in this noisy ECG. Write a matched filter which uses an R-peak template. This shall be taken from the noise free ECG. Employ heuristics which remove spurious detections and in general make sure the resulting heartrate is without any detection errors. Plot the momentary heartrate (i.e. inverse intervals between R-peaks) against time. **[25%]**

Every report must be based on different ECG recordings and every team needs to have two different ECGs (one noise-free lying down and one with noise standing). Please keep it short but it should make clear what you have done and why you have done it. Include the complete Python code as well as plots of the ECGs (timedomain) and their frequency representation (with proper labels). If necessary annotate the plots with InkScape, Corel Draw or Illustrator by labelling the ECG peaks and remember to use vector based image formats, for example EPS, SVG, PDF or EMF and not pixel based formats for the report. These figures need to be in the report at the correct place and not attached separately. Also, show zoomed in ECG traces of one heartbeat so that it is possible to identify the different parts of a single heartbeat and that it’s possible to check if it’s still intact.

No high level Python functions except of FFT/IFFT and the window functions are allowed. Any use of “lfilter”, “firwin”, “conv”, “correl” and any acausal processing (i.e. data array in and array out) commands will result in zero or very low marks. As before submit a zip file containing all files and test the zip before submission by unzipping it and then running python from the command-line in a terminal (not spyder). Also check that all plots are generated when running the script from the command line. See moodle for the exact filename conventions.

Deadline is 6th November, 3pm.