Assignment 2, Digital Signal processing: FIR filters

Bernd Porr

2020

The task of this assignment is to filter an ECG with FIR filters and to detect the R peaks. In contrast the FFT assignment here 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.

1 ECG filtering

Download the ECG according to the last digit of your matric number.

1. Create a Python FIR filter *class* which implements an FIR filter which has a method of the form `value dofilter(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 FIR_filter:
    def __init__(self,_coefficients):
    # your code here

    def dofilter(self,v):
    # your code here
        return result
    ```

    Implement the FIR filter in an efficeint way for example by using a ring buffer or by smart application of the Python slicing operations.

Minimise the amount of data being shifted and explain how / why you have done it. Put the filter class in a separate file for example fir-filter.py so that it turns into a module which can be imported by the main program. [20%]

2. Add a *unit test* to the module fir-filter.py from 1 which tests if the FIR filter works properly. For example the delay line and the proper multiplication of the coefficients. The unit test should be called if one starts the module with `python fir-filter.py`. Add the unit test by calling a function called `unittest` if run as a main program:

```
if __name__ == "__main__":
    unittest()
```

[20%]

3. Download a *short* ECG from moodle according to the last digits of your matric number. Filter this ECG with the above FIR filter class by removing the 50Hz interference and the DC. Decide which cutoff frequencies are needed and provide explanations by referring to the spectra and/or fundamental frequencies. Calculate the FIR filter coefficients numerically (= using python's IFFT command). 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. [20%].

2 ECG heartrate detection

The task is to detect the *momentary* heart rate $r(t)$ over a longer period of time. For example, after exercise you should see a slow decay of the heart rate to the baseline of perhaps 60 beats per minute. It is not the average heart rate but the frequency derived from the times *between adjacent heartbeats*.

1. Create a matched filter by using one QRST complex from an ECG and detect the R-peaks. Remember that for matched filters DC free signals are important. Here, the pre-filtering of the ECGs can be done differently to the filtering above to reduce as much DC as possible and any interference. [20%]

2. Use an Einthoven II recording where the person is walking using this ECG database: `https://pypi.org/project/ecg-gudb-database/`. The BEng students take subject numbers 0-9 matching their matriculation number and the postgraduate students subject numbers 10-19 with matching matric numbers 0-9. The database has a Python API and there is an example on github (`https://github.com/berndporr/ECG-GUDB`) how to use it. Calculate the *momentary* heart rate $r(t)$ over time (not the average!) by measuring the intervals between the detected hearbeats over the whole period of the ECG. Detect the heartbeats by employing a threshold. Add code which removes wrong detections and explain what the code does. [20%]

Every report must be based on a different ECG recording. Please keep it short but it should make clear what you have done and why you have done it. Include the Python code as well as plots of the ECGs (timedomain) and their frequency representation (with proper lables). If necessary enhance the plots with InkScape, Corel 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 (see Fig. 1) 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 a-causal processing (i.e. data array in and array out) commands will result in zero or very low marks. As before submit a zip 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 commandline. See moodle for the exact filename conventions.

Deadline is 9th November.

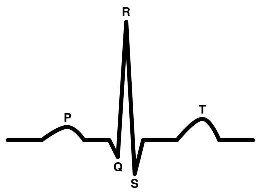

Figure 1: A normal ECG with P,Q,R,S,T waves (taken from Wikipedia)