Assignment 3, Digital Signal processing: IIR filters

Bernd Porr

2020

This assignment covers IIR filters.

> Your task is to solve a problem which requires realtime filtering of a physical quantity.

You have two options for the data acquisition:

- Buy an Arduino. They work under Windows, Mac and Linux. They have a sampling rate of 100 Hz.
- If you don't want to spend any money you can use your webcam as an RGB colour sensor, sampling at its frame rate.

Every team needs to have a different *application* for their measurement. Think of something simple such as measuring the speed of a fan with an LDR or the webcam. Perhaps you have a fishtank and you'd like to know how often a fish swims past. Or if the flowerpot is watered. Add your topic to the wiki provided on moodle as soon as possible. It needs to solve a practical real life problem and require low, high, bandpass or bandstop filtering. Excessive use of identical topics will result in low marks.

Again you work in teams of two students and one report is submitted per team.

1. Present a realtime measurement problem to be solved which requires filtering. For example noise needs to be removed or a signal needs to be detected. Marks are given for initiative, inventiveness and originality (= ideas which haven't come straight from the lecturer, lab demonstrators or other groups). Document the experiment with (all compulsory):

- photos of the setup
- dataflow diagrams
- YouTube clip(s)

in addition to your report. How would you like to present the results? Just as a plot or perhaps a bar graph? QT for Python might be an option to look into. [20%]

2. Check the sampling rate of your acquisition by running it for example for 10secs and check against the number of samples expected. How could you check for jitter in the sampling? [20%]

3. Determine the filter response(s) which are required and justify them. Generate the SOS coefficients for the filter(s) either with the help of Python's high level functions or analytical solutions shown in the lecture. [20%]

4. Write two classes:

   (a) `IIR2Filter` which implements a 2nd order IIR filter which takes the coefficients in the constructor and has a method called:

   ```
   y=IIR2Filter.filter(x)
   ```

   where `y` and `x` are simple scalars (no arrays) as usual. Optimise this class that it won't need any arrays for its buffers and coefficients.

   (b) a class `IIRFilter` which directly takes the SOS array from the high level IIR design commands as its constructor argument and which then creates a chain of 2nd order filter instances of IIR2Filter classes. Thus they form an array of instances of IIR2Filter. Again implement a function which then filters the signal:

   ```
   y=IIRFilter.filter(x)
   ```

   and then internally processes the data x by sending it through the chain of 2nd order IIR2Filter classes. Implement the filtering operation again in the most effective way by not using index operations.

   [30%]

5. Compare your filtered results with the original recordings, show both signals in a realtime demo (YouTube clip) and discuss if you have been successful. Do a critical analysis. [10%]

High level *design* commands are allowed such as "butter" but the actual IIR filtering operations need be written from scratch as outlined above. Any use of "lfilter", "conv" or other high level python filter operation will result again in zero marks. Proof of realtime processing in form of a video needs to be given and the video needs to show clearly what it's about. Please add your link to the wiki and also add a readme to your zip containing all the files.

The code needs to be again submitted as a zip file and it will be tested if it runs. Crashing code will result in low marks.

As before I expect sharp figures in vector format in the report. The complete code needs to be in the appendix and also uploaded to moodle.

Since this task is more difficult also to coordinate in a distributed team there is more time to finish it.

Deadline for the report is 7th Dec.