fabian.brinkmann@tu-berlin.de Winter 2023/24

3. Homework

Total score: 25 points

Topics: Square wave synthesis, signal flow graphs, second order sections, windowing, aliasing.

Submission

Submit the homework by **Sunday 3. March 2024 (23:55)** via the ISIS portal. Late submissions or submissions that are not made via the ISIS portal will not be considered. Only one submission must be uploaded per group.

The homework must be submitted as an **iPython Notebook** that can include code, plots, text, and equations written with LaTex. Please **run the Notebook** before the submission to make sure all plots and outputs are included.

Include the **names and matriculation numbers** of all group members in the iPython notebook, comment your code, and submit all data required to run the code in a **single zip file** named for example **homework_01_group_A.zip**

The following Python packages might help you to solve the tasks. pyfar can **only** be used if it is explicitly allowed:

`numpy`, `scipy`, `matplotlib`, `sounddevice`, `timeit`, `pyfar`.

1 Square Wave Synthesis

The square wave shown in Fig. 1 is a fundamental wave form that is available in many synthesizers and used in many pieces of electronic music. It can be generated in the time and frequency domain. In the time domain it can be written as a sum of sine signals

$$x(t) = \sum_{k=1}^{K} \frac{1}{2k-1} \sin((2k-1)\omega_0 t) \tag{1}$$

with the fundamental frequency f_0 in Hz and $\omega_0 = 2\pi f_0$.

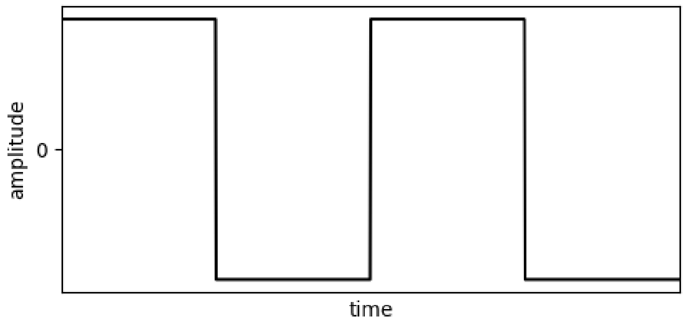

Figure 1: Ideal square wave

a) Generate square waves in the frequency domain by (i) computing the single sided spectrum according to equation (1), (ii) mirroring the spectrum under consideration of the DFT symmetry, and (iii) obtaining the time signal through the inverse DFT. Plot the time signal using $f_0 = 100$ Hz, a sampling rate of $f_s = 48$ kHz, a duration of $N = 480$ samples and for $K = \{1, 2, 8, 100\}$.

Hints:

- You can use the amplitude given in equation (1) as the absolute values for the bins of the spectrum, but what phase must be used?
- The length of the single sided spectrum is `N//2+1`

b) Explain the plot from (a) in comparison to Fig. 1.

c) Loop the time signals from (a) to obtain signals with a duration of 1 s and play back the signals.

Submission: iPython Notebook
8 Points: a: 5 points, b: 2 points, c: 1 points

2 Signal Flow Graphs

Derive the difference equation for the non-recursive signal flow graph shwon in Fig. 2 and eliminate the node signals $w_1[n]$ and $w_2[n]$ from the final expression for $y[n]$.
Submission: written
3 Points

3 Second Order Sections

Second Order Sections (SOS) are often used in digital signal processing due to their robustness against finite precision number formats. An example for a cascaded system

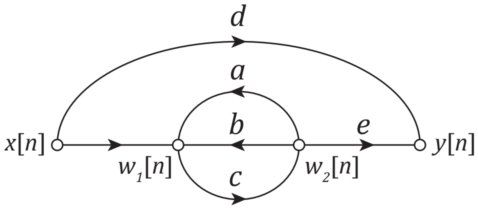

Figure 2: Signal flow graph of a digital system

consisting of two SOS is shown in Fig. 3.

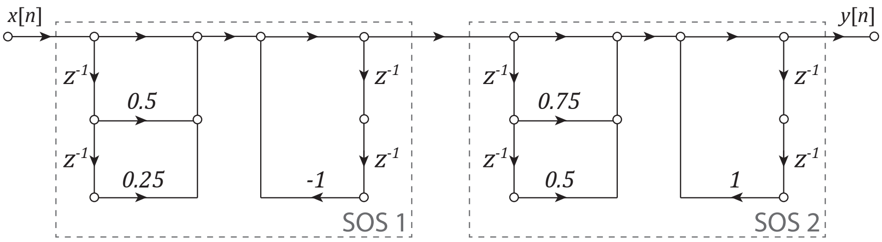

Figure 3: Signal flow graph of a 4th order system consisting of two cascaded SOSs.

a) Redraw the signal flow graph from Fig. 3 in direct dorm II.

b) Give the z-transfer function of the system shown in Fig. 3.

c) Calculate the z-transfer function of the equivalent non-cascaded 4th order system.

d) Draw the signal flow graph of the the equivalent non-cascaded 4th order system.

e) Give the difference equation of the non-cascaded 4th order system.

f) Show that the cascaded and non-cascaded system are equivalent. There are many ways to do this from which you can chose one.

Submission: written (tasks a-e) iPython Notebook (task f)
7 Points: a: 1 point, b: 1 point, c: 2 points, d: 1 point e: 1 point f: 1 point

4 Effect of Time Windows on the Spectrum

Generate a 2 second sine signal with a frequency of 2 kHz and a sampling rate of 16 kHz. Use a rectangular, triangular and Hann window (`scipy.signal.windows`) and apply the windows with two different length (20 periods and 20.25 periods) to the 2 kHz sine signal. Plot the windows (time signals) and the logarithmic magnitude spectra of the six windowed sine signals and explain the results.

Hint: multiply the spectra $X[k]$ by `2 / numpy.sum(window)` to get comparable results.

Submission: iPython Notebook
4 Points: 2 points for the signals and plots, 2 points for the explanations.

5 Sine signals

Create three sine signals with a sampling rate of $f_s = 16$ kHz, a duration of 1 second and the frequencies 1 kHz, 4 kHz and 15 kHz. The amplitude should be 0.5. Plot the first millisecond of each of the four signals in 3 by 1 subplot layout. Play back the signals and compare the heard frequencies. What are the actual frequencies of the digital signals? What effect occurs for the 15 kHz signal?

Submission: Notebook
3 points: Signal generation (1), plot and playback (1), explanations (1)