Universität Rostock Traditio et Innovatio

Institute of Communications Engineering

Prof. Dr.-Ing. Sascha Spors
Signal Theory and Digital Signal Processing
Institute of Communications Engineering (INT)
Faculty of Computer Science and Electrical Engineering (IEF)

**Lab exercise III in Digital Signal Processing, winter semester 2018/19** (course #24505)

Lecturer: Vera Erbes, email: vera.erbes@uni-rostock.de, room: W: 8.211
Authors: Frank Schultz, Vera Erbes

**All 4 lab exercises are due 10th Feb. 2019.**

# (Re)-quantisation, dithering & noise shaping

## Exercise 1: Characteristic curve of the uniform midtread quantiser (5 points)

Solve the following tasks in Matlab or Python. Code examples here follow Matlab syntax and have to be adjusted for Python.

a) Write a function `xq = my_quant(x,N)` that quantises a quasi amplitude-continuous signal $x$ with a value range of $-1 \leq x \leq 1$ with an arbitrary, but odd number $N$ of quantisation steps using the midtread uniform quantiser characteristic curve. Make use of the `round` function and the quantisation model that was introduced in the course. Values above 1 or below $-1$ shall be clipped to the highest and lowest quantisation step, respectively. **(1.5 points)**

b) Test `xq = my_quant(x,N)` for a signal vector `x = -1:0.001:1` and generate the left of fig. 1 using $N = 17$ quantisation steps. **(1 point)**

c) Modify `xq = my_quant(x,N)` so that it also works for even $N$. To this end, the last quantisation step for positive amplitudes is increased, cf. fig. 1 on the right. This models typical analogue-to-digital (ADC) / digital-to-analogue (DAC) converters that use $N = 2^w$ quantisation steps with number of bits $w \in \mathbb{N}$. In video applications, typically 8 to 16 bit can be found, in audio typically 16 to 24 bit. **(1.5 points)**

d) Test the modified `xq = my_quant(x,N)` for a signal vector `x = -1:0.001:1` and generate the right of fig. 1 using $N = 16$ quantisation steps, i.e. resolution of 4 bit. **(1 point)**

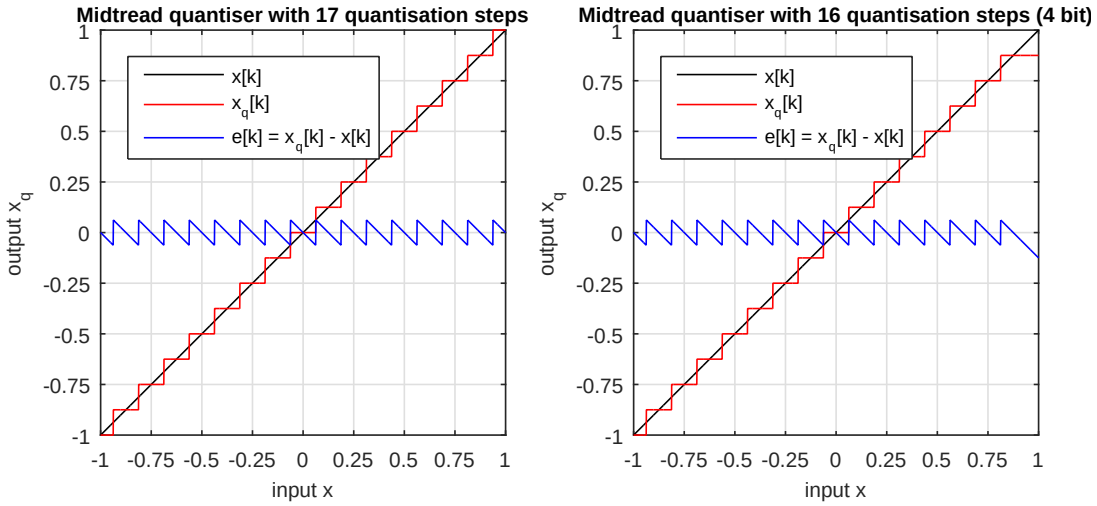


Figure 1: Uniform midtread quantiser

## Exercise 2: Quantisation of different signals, SNR (5 points)

Solve the following tasks in Matlab or Python. Code examples here follow Matlab syntax and have to be adjusted for Python.

a) Generate a signal vector `xSine` that contains a discrete-time, zero-mean **sine signal** $x[k] = A \cdot \sin(\Omega\, k)$ with normalised discrete-time angular frequency $\Omega = \frac{2\pi f}{f_s} = \frac{2\pi}{50}$ and a variance $\sigma_x^2 = \frac{1}{2}$ (as a time average measure) for $0 \leq k < 50000$. **(1 point)**

b) Generate a signal vector `xNorm` that contains a discrete-time, zero-mean, **normally distributed noise signal** (`randn()`) with a variance $\sigma_x^2 = 0.0471$ for $0 \leq k < 50000$. By doing so the signal amplitudes are mostly within the range $-1 \leq$ `xNorm` $\leq 1$ and theoretically only 1 sample out of 100,000 samples has a larger amplitude |`xNorm`| $> 1$ that would clip the quantiser modeled in exercise 1, cf. [Zöl08, fig. 2.5]. **(1 point)**

c) Generate a signal vector `xUniform` that contains a discrete-time, zero-mean, **uniformly distributed noise signal** (`rand()`) with a variance $\sigma_x^2 = \frac{1}{3}$ for $0 \leq k < 50000$. By doing so the signal amplitudes are theoretically within the range $-1 \leq$ `xUniform` $\leq 1$, cf. [Zöl08, p. 23]. **(1 point)**

d) Generate a signal vector `xLaplace` that contains a discrete-time, zero-mean, **noise signal following the Laplace distribution** (`laprnd()`[1]) with a variance $\sigma_x^2 = 0.0236$ for $0 \leq k < 50000$. **(1 point)**

e) Generate fig. 2: apply the 4 generated signals to the quantisation with `xq = my_quant(x,N)` from exercise 1 for different numbers of bits and calculate the signal-to-noise ratio in dB as

$$\mathrm{SNR} = 10 \cdot \log_{10}\left(\frac{\sigma_x^2}{\sigma_e^2}\right) \tag{1}$$

[1] You can get this function at: `http://www.mathworks.com/matlabcentral/fileexchange/13705-laplacian-random-number-generator`

using the quantisation error signal $e[k] = x_q[k] - x[k]$. Note that your simulation results for the noise signals may differ depending on the state of the used random number generator (i.e. how many samples have amplitudes $|x[k]| > 1$). To validate your code use the sine signal as reference. Then compare your results with the theoretical SNRs (cf. course notes and [Zöl08, p. 23-24]):

- zero-mean, uniformly distributed with $\sigma_x^2 = \frac{1}{3}$: $\mathrm{SNR}_{\mathrm{dB}} = 6.02\,\mathrm{dB} \cdot w$
- zero-mean, full scale sine signal with $\sigma_x^2 = \frac{1}{2}$: $\mathrm{SNR}_{\mathrm{dB}} = 6.02\,\mathrm{dB} \cdot w + 1.76\,\mathrm{dB}$
- zero-mean, normally distributed noise with $\sigma_x^2 = 0.0471$: $\mathrm{SNR}_{\mathrm{dB}} = 6.02\,\mathrm{dB} \cdot w - 8.5\,\mathrm{dB}$
- zero-mean, Laplace distributed noise with $\sigma_x^2 = 0.0236$: $\mathrm{SNR}_{\mathrm{dB}} = 6.02\,\mathrm{dB} \cdot w - 9\,\mathrm{dB}$

**(1 point)**

## Exercise 3: Dithering (10 points)

Solve the following tasks in Matlab or Python. Code examples here follow Matlab syntax and have to be adjusted for Python.

The discrete-time sine signal $x[k] = Q \cdot \sin(\frac{2\pi f_{\sin}}{f_s} k)$, $0 \leq k < 50000$ with $f_{\sin} = 960\,\mathrm{Hz}$ and sampling frequency $f_s = 48\,\mathrm{kHz}$ shall be quantised with the $w = 3$ Bit midtread quantiser shown in fig. 3 ($Q = 1/4$, $|x_{\max}| = 1$).
Before quantising $x[k]$, a dither noise signal $d[k]$ shall be added to $x[k]$ according to fig. 4. This dither signal with small amplitudes should decorrelate the quantisation error $e[k]$ from the quantised signal $x_q[k]$, which is especially important for small amplitudes of $x[k]$. This technique is called *dithering*. For $d[k] = 0$ no dithering is applied.
Since the quantisation error may be in the range $-Q/2 \leq e[k] \leq Q/2$ (assuming uniform distribution), it appears reasonable to use a dither noise with a probability density function (PDF) of

$$p_{\mathrm{RECT}}(d) = \frac{1}{Q} \operatorname{rect}\left(\frac{d}{Q}\right), \tag{2}$$

i.e. a **zero-mean, uniformly distributed noise** with maximum amplitude $|d[k]| = Q/2$. It can be shown that this dither noise improves the quality of the quantised signal. However, there is still a noise modulation (i.e. a too high correlation between $x_q[k]$ and $e[k]$) that depends on the amplitude of the input signal.
The noise modulation can be almost completely eliminated with a zero-mean noise signal exhibiting a **symmetric triangular PDF**:

$$p_{\mathrm{TRI}}(d) = \frac{1}{Q} \operatorname{tri}\left(\frac{d}{Q}\right) \tag{3}$$

with maximum amplitude $|d[k]| = Q$. By doing so, an almost ideal decorrelation between $x_q[k]$ and $e[k]$ is realised. In audio, this technique is called TPDF-Dithering (Triangular Probability Density Function Dithering) and can be applied in the mastering process of audio material that is to be distributed e.g. on a CD. The very first CDs in the middle of the 1980s did not have any dithering included, which is one of the reasons why their sound is often described as harsh.

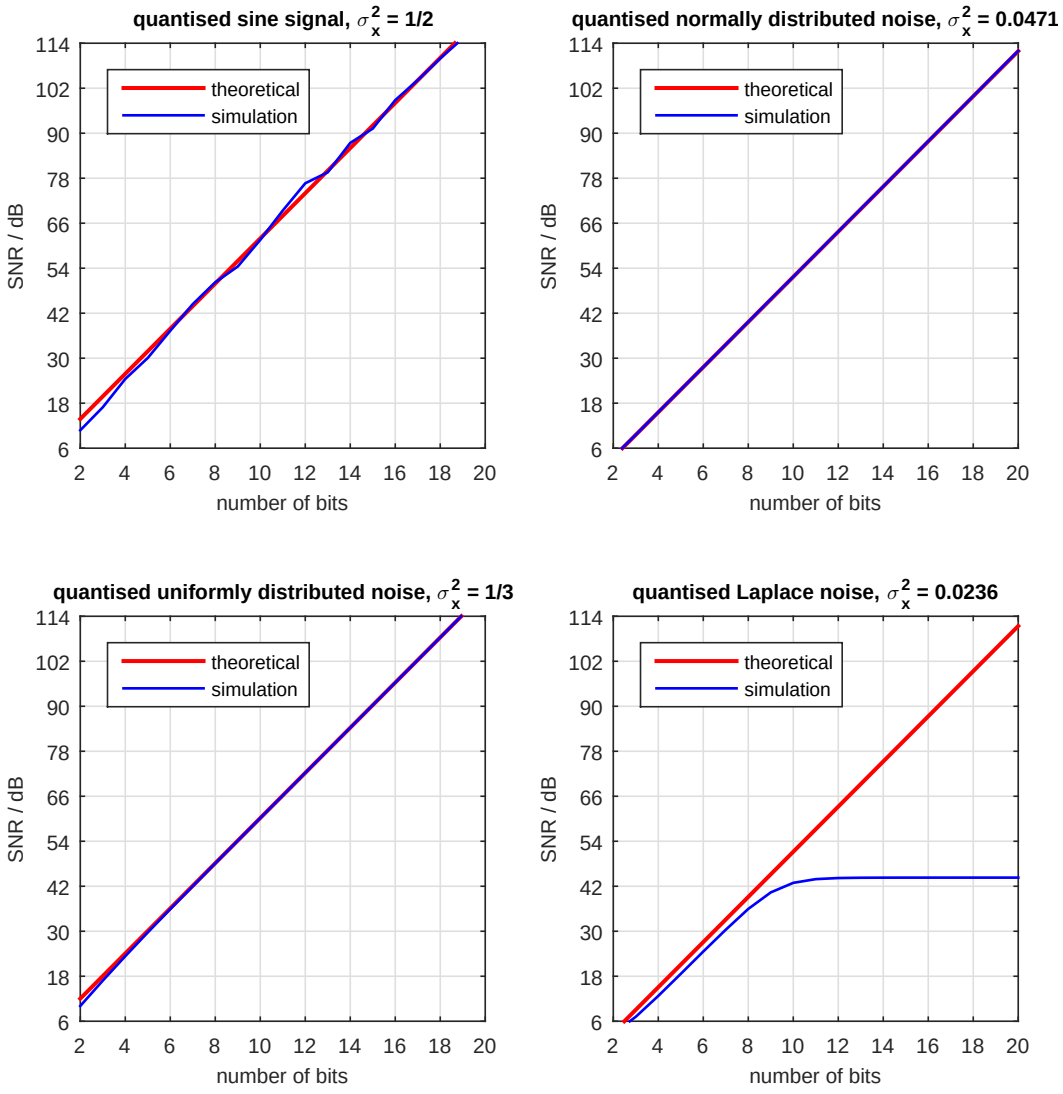


Figure 2: Signal quantisation vs. SNR

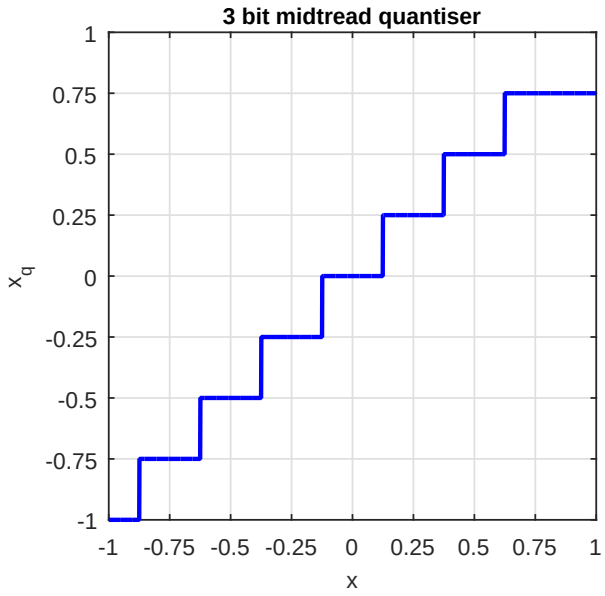


Figure 3: Midtread quantiser 3 bit, 8 quantisation steps

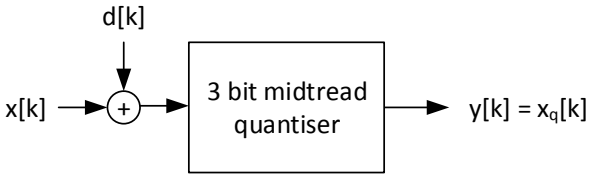


Figure 4: Quantiser with preceding dithering

To get an impression on how dithering may be implemented and what quantised signals sound like, the following exercises shall be performed:

a) Generate the sine signal $x[k]$ defined above. **(0.5 points)**

b) Generate the dither noise $d_{\mathrm{RECT}}[k]$ according to the PDF $p_{\mathrm{RECT}}(d) = \frac{1}{Q}\mathrm{rect}(\frac{d}{Q})$. Check the resulting amplitude and distribution carefully. The length of $d_{\mathrm{RECT}}[k]$ and $x[k]$ must be equal. **(2 points)**

c) Generate the dither noise $d_{\mathrm{TRI}}[k]$ according to the PDF $p_{\mathrm{TRI}}(d) = \frac{1}{Q}\mathrm{tri}(\frac{d}{Q})$. Check the resulting amplitude and distribution carefully. The length of $d_{\mathrm{TRI}}[k]$ and $x[k]$ must be equal. **(2 points)**

d) Add each dither noise $d_{\mathrm{RECT}}[k]$ and $d_{\mathrm{TRI}}[k]$ individually to $x[k]$. Together with the original signal without dithering you now have three vectors, e.g. named `xNODITH`, `xRECT` and `xTRI`. **(0.5 points)**

e) Quantise these signals individually with `xq = my_quant(x,N)` from exercise 1, leading to `xqNODITH`, `xqRECT` and `xqTRI` for $N = 2^w = 2^3 = 8$ quantisation steps. **(1 point)**

f) Generate the left column of fig. 5. Plot three signal periods of the sine function. Please note: (i) since you are dealing with discrete-time signals, stem plots would be the most correct visualisation, which for reasons of clarity and comprehensibility should not be used here and (ii) since random signal generators are involved, your results might slightly differ. Interpret the graphics. **(2 points)**

g) For each graphic, render WAV files (in Matlab: `audiowrite('xqNODITH.wav',xq,fs, 'BitsPerSample',32)`) from $x[k]$, $x_q[k]$ und $e[k]$ and listen to them. Please be careful when testing, do not harm your ears! Pay special attention to the sound of the quantisation error, how it is correlated with the quantised signal and how loud it appears. **(1 point)**

h) You are now able to quantise and listen to signals whose amplitude is even below the quantisation step size, i.e. $< Q$. For that, use the sine signal this time with amplitude $Q/8$ and dithering with RECT and TRI dither noise. Compare with the results from g). **(1 point)**

## Exercise 4: 2nd order noise shaping (10 points)

A further improvement in the (re-)quantisation process that is included in virtually all ADCs and DACs nowadays is a technique called noise shaping. The key idea of noise shaping is to filter the quantisation error such that it is perceived as less disturbing. For audio signals the energy is typically shifted towards higher frequencies since the ear is less sensitive there. In principle, any suitable filter function can be employed.
In fig. 6, two systems for potential, very simple 2nd order noise shaping (named after the involved 2nd order IIR filters) are depicted that spectrally shape the quantisation error $e[k] = x_q[k] - x[k]$. As before in the case of simple dithering, the dither noise is added to decorrelate the quantisation error from the quantised signal. The two systems differ in the application of the dither signal.

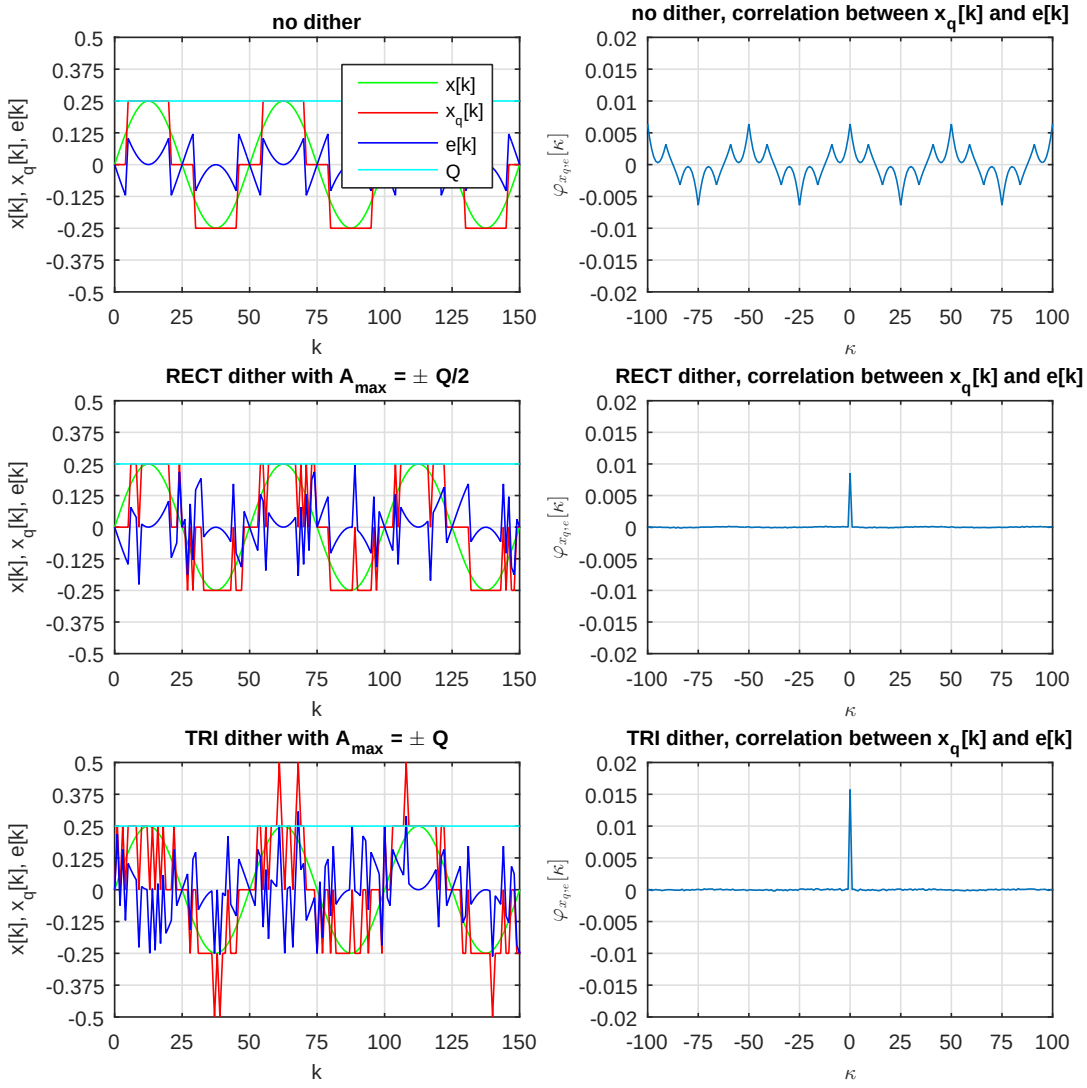


Figure 5: Dither techniques

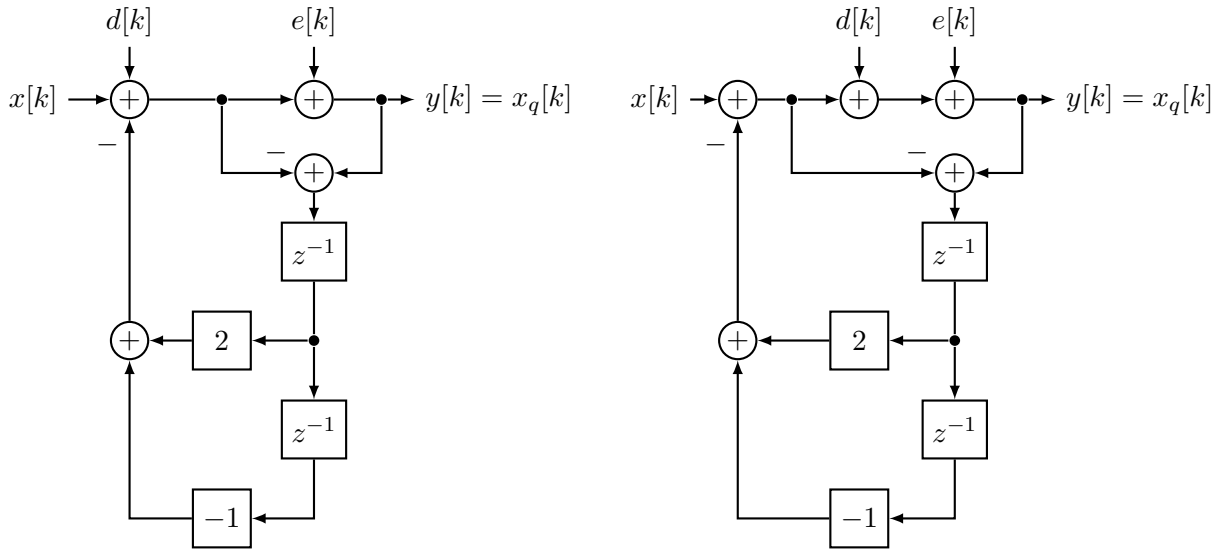


Figure 6: Two systems for dithering plus noise shaping of 2nd order

In order to investigate the performance of the two noise shaping systems, the following tasks shall be performed:

a) Give the difference equation for both systems. **(2 points)**

b) Derive the signal transfer function $H_X(z) = \frac{Y(z)}{X(z)}$ (with $D(z)$ and $E(z)$ being zero) and the so-called noise transfer function $H_E(z) = \frac{Y(z)}{E(z)}$ (with $X(z)$ and $D(z)$ being zero) for both systems. **(2 points)**

c) Give the DTFT absolute magnitude spectrum $|H_E(\Omega)|$ of $H_E(z)$ and simplify as much as possible towards a rather simple expression for both systems. **(2 points)**

d) Visualise the absolute magnitude spectrum $|H_E(\Omega)|$ and discuss the system characteristics. **(2 points)**

e) What is the difference between both systems? Which system appears more sensible for our purpose? **(2 points)**

## References

[Zöl08] Zölzer, U. (2008): *Digital Audio Signal Processing*. Chichester: Wiley, 2. ed.