DASP3rd Chapter 6 - Exercises

1 Design of Recursive Audio Filters

1. How can we design a low-frequency shelving filter? Which parameters define the filter? Explain the control parameters.

   Solution:

   - We design low-frequency shelving filters with a lowpass parallel to a direct path
   - Parameters: cutoff frequency f_c and gain G (V_0 at $\omega = 0$: boost or cut)

2. How can we derive a high-frequency shelving filter? Which parameters define the filter?

   Solution:

   - We derive high-frequency shelving filters from low-frequency shelving filters by means of lowpass to highpass transformation: $s \rightarrow \frac{1}{s}$
   - Parameters: cutoff frequency f_c and gain G (V_0 at $\omega = \infty$: boost or cut)

3. What is the difference between first- and second-order shelving filters?

   Solution:

   - Second-order shelving filters have complex poles/zeros
   - Increase of slope of the filter response in the transition band

4. How can we design a peak filter? Which parameters define the filter? What is the filter order? Describe the control parameters. Describe the Q-factor.

   Solution:

   - We design a peak filter by parallel connection of a second-order bandpass and a direct path
   - Parameters: center frequency f_c, gain G and Q-factor Q (or bandwidth f_b)
   - Peak filters are second-order filters
   - Q-Factor: $Q = \frac{f_c}{f_b}$

5. How do we derive the digital transfer function?

   Solution:

   - By mapping from s-domain into Z-domain using bilinear transform: $s = \frac{2}{T}\frac{z-1}{z+1}$, where $T = \frac{1}{f_S}$

6. Derive the digital transfer functions for the first-order shelving filters.

   Solution:

   - See '3. Shelving Filter: Direct Form'

2 Parametric Audio Filters

1. What is the basic idea for parametric filters?

   Solution:

   - The design is based on allpass decomposition
   - Direct access to the parameters of the transfer function
   - Only one coefficient is calculated

2. What is the difference between the Regalia and the Zölzer filter structures? Count the number of multiplications and additions for both filter structures.

   Solution:

   - The Regalia filter structures require at least two multiplications and one (three) addition (see pages 164-165)
   - The Zölzer filter structures require only one multiplication and one (two) addition (see pages 167-170)

3. Derive a signal flow graph for first- and second-order parametric Zölzer filters with a direct-form implementation of the allpass filters?

   Solution:

   - First-order low-frequency shelving filter: direct path plus allpass both weighted by $\frac{H_0}{2}$ plus an additional direct path (see Fig. 6.21 on page 168)
   - First-order high-frequency shelving filter: direct path minus allpass both weighted by $\frac{H_0}{2}$ plus an additional direct path (see Fig. 6.22 on page 169)

4. Is there a complete decoupling of all control parameters for boost and cut? Which parameters are decoupled?

   Solution:

   - Filter coefficient for boost case a_B is only coupled with cutoff frequency ω_c
   - Filter coefficient for cut case a_C is coupled with cutoff frequency ω_c and gain factor V_0

3 Shelving Filter: Direct Form

Derive a first-order low-shelving filter from a purely band-limiting first-order lowpass filter. Use bilinear transform and give the transfer function of the low-shelving filter.

1. State the filter coefficients and calculate the poles/zeros as functions of V_0 and T. Which gain factor do you have if $z = \pm 1$?

   Solution:

   - We derive our shelving filter from lowpass and allpass filter as
   $$H(s) = 1 + \frac{H_0}{s+1} = \frac{s+1+H_0}{s+1} = \frac{s+V_0}{s+1}.$$
   Using bilinear transform, we have:
   $$H(z) = \frac{\frac{2}{T}\frac{z-1}{z+1} + V_0}{\frac{2}{T}\frac{z-1}{z+1} + 1} = \frac{2(z-1) + TV_0(z+1)}{2(z-1) + T(z+1)} = \frac{(2+TV_0)z + TV_0 - 2}{(2+T)z + T - 2}$$
   $$= \frac{\frac{2+TV_0}{T-2}z + \frac{TV_0-2}{T-2}}{\frac{T+2}{T-2}z + 1} = \frac{\frac{TV_0-2}{T-2} + \frac{2+TV_0}{T-2}z}{1 + \frac{T+2}{T-2}z}.$$
   The coefficients are given as $b_0 = \frac{TV_0-2}{T-2}$, $b_1 = \frac{2+TV_0}{T-2}$, $a_0 = 1$, and $a_1 = \frac{T+2}{T-2}$. The zeros and poles can be obtained as $z_0 = \frac{TV_0-2}{2*TV_0}$ and $z_\infty = \frac{2-T}{2+T}$, respectively.
     * if $z = 1 \Rightarrow s = 0$: $H(z) = V_0$
     * if $z = -1 \Rightarrow s = \infty$: $H(z) = 1$

2. What is the difference between purely band-limiting filters and shelving filter?

   Solution:

   - Shelving filters are used to weight certain frequencies and pass the other frequencies
   - Band-limiting filters pass a given frequency band and attenuate frequencies outside this band

3. How can you describe the boost and cut effect related to poles/zeros of the filter?

   Solution:

   - Boost and cut can be adjusted by moving respectively the variable zeros or poles from unit circle.

4. How do we get a transfer function for the boost case from the cut case?

   Solution:

   - We go from cut to boost by swapping poles and zeros.

5. How do we go from a low-shelving filter to a high-shelving filter?

   Solution:

   - We go from low-frequency shelving to high-frequency shelving by replacing $s \rightarrow \frac{1}{s}$ (lowpass/highpass transformation)

4 Shelving Filter: Allpass Form

Implement a first-order high-shelving filter for the boost and cut case with the sampling rate $f_S = 44.1\,\text{kHz}$, the cutoff frequency $f_c = 10\,\text{kHz}$, and the gain $G = 12\,\text{dB}$.

1. Define the allpass parameters and coefficients for the boost and cut case.

   Solution:

   - Boost case: $V_{0b} = 10^{G/20}$, $H_{0b} = V_{0b} - 1, a_b = \frac{tan(2\pi f_c \frac{T}{2})-1}{tan(2\pi f_c \frac{T}{2})+1}$
     Allpass: $B_b = [a_b, 1]$; $A_b = [1, a_b]$
   - Cut case: $V_{0c} = 10^{-G/20}$, $H_{0c} = V_{0c} - 1, a_c = \frac{V_{0c} tan(2\pi f_c \frac{T}{2})-1}{V_{0c} tan(2\pi f_c \frac{T}{2})+1}$
     Allpass: $B_c = [a_c, 1]$; $A_c = [1, a_c]$

2. Derive from the allpass decomposition the complete transfer function of the shelving filter.

   Solution:

   - Since $A(z) = \frac{z^{-1}+a_b}{1+a_b z^{-1}}$, we have:

   $$H(z) = 1 + \frac{H_0}{2}[1 - A(z)] = 1 + \frac{H_0}{2}[1 - \frac{z^{-1}+a_b}{1+a_b z^{-1}}] = 1 + \frac{H_0}{2}\frac{1 - a_b + (a_b - 1)z^{-1}}{1 + a_b z^{-1}}$$
   $$= \frac{1 + \frac{H_0}{2}(1 - a_b) + (\frac{H_0}{2}(-1 + a_b) + a_b)z^{-1}}{1 + a_b z^{-1}}$$

   - Boost case: $B_{bs} = [1 + \frac{H_{0b}}{2}(1 - a_b), \frac{H_{0b}}{2}(-1 + a_b) + a_b]$; $A_b s = [1, a_b]$
   - Cut case: $B_{cs} = [1 + \frac{H_{0c}}{2}(1 - a_c), \frac{H_{0c}}{2}(-1 + a_c) + a_c]$; $A_c = [1, a_c]$

3. Using Matlab, give the magnitude frequency response for boost and cut. Show the result for the case where the boost and cut filters are in a series connection.

   Solution:

   - see Matlab script 'dasp_ex6_4_3'

4. If the input signal to the system is a unit impulse, give the spectrum of the input and output signal for the boost and cut case. What will you expect as result in this case when boost and cut are again cascaded?

   Solution:

   - see Matlab script 'dasp_ex6_4_3'

5 Quantization of Filter Coefficients

For the quantization of the filter coefficients, different methods have been proposed: direct form, Gold and Rader, Kingsbury, and Zölzer.

1. What is the motivation behind this?

   Solution:

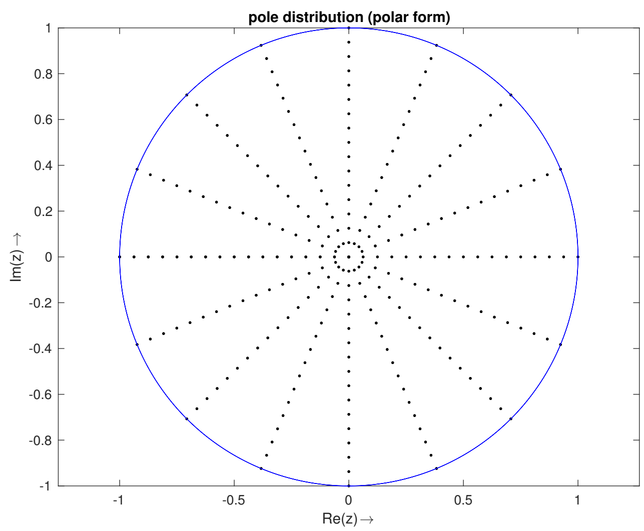
   - By increasing the pole density in a certain region of the z-plane, the coefficient sensitivity and the roundoff noise of the filter structure are reduced
   - Direct form: Decreased pole density close to real axis (see Fig. 6.27 on page 173)
   - Gold and Rader: uniform distribution of pole locations (see Fig. 6.29 on page 174)
   - Kingsbury: Increased pole density at $z = 1$ (see Fig. 6.31 on page 175)
   - Zölzer: Increased pole density at $z = 1$ (see Fig. 6.34 on page 176)
   - see Matlab script 'dasp_ex6_5_1'

2. Plot a pole distribution using the quantized polar representation of a second-order IIR filter

$$H(z) = \frac{N(z)}{1 - 2r\cos\varphi z^{-1} + r^2 z^{-2}} \tag{1}$$

   Solution:

   - see Matlab script 'dasp_ex6_5_2'



6 Signal Quantization Inside the Audio Filter

Now we combine coefficient and signal quantization.

1. Design a digital highpass filter (second-order IIR), with a cutoff frequency $fc = 50\,\text{Hz}$. (Use the Butterworth, Chebyshev, or elliptic design methods implemented in Matlab.)

   Solution:

   - see Matlab script 'dasp_ex6_6_1'

2. Quantize the signal only when it leaves the accumulator (i.e. before it is saved in any state variable).

   Solution:

   - see Matlab script 'dasp_ex6_6_1'

3. Now quantize the coefficients (direct form) too.

   Solution:

   - see Matlab script 'dasp_ex6_6_1'

4. Extend your quantization to every arithmetic operation (i.e. after each addition/multiplication).

   Solution:

   - see Matlab script 'dasp_ex6_6_1'

7 Quantization Effects in Recursive Audio Filters

1. Why is the quantization of signals inside a recursive filter of special interest?

   Solution:

   - Without quantization, every run through the feedback loop increases the word length of the output signal ⇒ A quantization of the output signal is required before it is fed back in order to maintain an output signal with constant word length.

2. Derive the noise transfer function of the second-order direct-form filter. Apply a first- and second-order noise shaping to the quantizer inside the direct-form structure and discuss the influence. What is the difference between second-order noise shaping and double-precision arithmetic?

   Solution:

   - Noise transfer function: $G_1(z) = G_2(z) = \frac{1}{1+b_1 z^{-1}+b_2 z^{-2}}$
   - With first-order noise shaping: $G_{1.O}(z) = \frac{1-z^{-1}}{1+b_1 z^{-1}+b_2 z^{-2}}$
   - With second-order noise shaping: $G_{2.O}(z) = \frac{1-2z^{-1}+z^{-2}}{1+b_1 z^{-1}+b_2 z^{-2}}$
   - If the feedback coefficients are chosen equal to the coefficients b_1 and b_2 in the denominator polynomial, complex zeros are produced that are identical to the complex poles. The noise transfer function $G(z)$ is then reduced to unity. The choice of complex zeros directly at the location of the complex poles corresponds to double-precision arithmetic.

3. Write a Matlab implementation of a second-order filter structure for quantization and noise shaping.

   Solution:

   - see Matlab script 'dasp_ex6_7_3'

8 Fast Convolution

For an input sequence $x(n)$ of length $N_1 = 500$ and the impulse response $h(n)$ of length $N_2 = 31$, perform the discrete time convolution.

1. Give the discrete time convolution sum formula.

   Solution:

   - $y(n) = \sum_{k=0}^{N-1} h(k)\, x(n-k) = \sum_{k=0}^{N-1} x(k)\, h(n-k)$ with $N = N_1 + N_2 - 1$

2. Define with Matlab $x(n)$ as the sum of two sinusoids and derive $h(n)$ with Matlab function *firpm*(..).

   Solution:

   - see Matlab script 'dasp_ex6_8_3'

3. Realize the filter operation with Matlab using:

   - the function *conv*(x, h);
   - the sample by the sample convolution sum method;
   - the FFT method;
   - the FFT with the overlap and add method.

   Solution:

   - see Matlab script 'dasp_ex6_8_3'

4. Describe FIR filtering with the fast convolution technique. Which conditions have to be fulfilled for the input signal and the impulse responses if convolution is performed by equivalent frequency-domain processing?

   Solution:

   - The basis of an efficient implementation is the fast convolution

     $$y(n) = x(n)h(n) \circ\!\!-\!\!\bullet\; Y(k) = X(k) \cdot H(k),$$

     where the convolution in the time domain is performed by transforming the signal and the impulse response into the frequency domain, multiplying the corresponding Fourier transforms and inverse Fourier transform of the product into the time domain signal (see Fig. 6.55 on page 190). The transform is carried out by a discrete Fourier transform of length N, such that $N = N_1 + N_2 - 1$ is valid and time-domain aliasing is avoided.

5. What happens if the input signal and impulse response are as long as the FFT transform length?

   Solution:

   - The transform is carried out by a discrete Fourier transform of length N, such that $N = N_1 + N_2 - 1$ is valid and time-domain aliasing is avoided.

6. How can we perform the IFFT with the FFT algorithm?

   Solution:

   1. Exchange real and imaginary part of the spectral sequence

   $$Y(k) = Y_I(k) + jY_R(k).$$

   2. Transform with FFT algorithm

   $$\text{FFT}[Y(k)] = y_I(n) + jy_R(n).$$

   3. Exchange real and imaginary part of the time sequence

   $$y(n) = y_R(n) + jy_I(n).$$

7. Explain the processing steps:

   - for a segmentation of the input signal into blocks and fast convolution;
   - for a stereo signal by the fast convolution technique;
   - for the segmentation of the impulse response.

   Solution:

   - Segmentation of input signal:
     * Zero-pad $h(n)$ block to length N.
     * Perform DFT of $h(n)$ block: $H(k)$.
     * Decompose the input sequence in $x_i(n)$ blocks of length $N_x = N - N_h + 1$.
     * Read $x_i(n)$ block and zero-pad it to length N.
     * Perform DFT of $x_i(n)$ block: $X_i(k)$.
     * Multiply $X_i(k)$ with $H(k)$
     * Build intermediate result $y_i(n)$ with overlap-add of real parts of IDFT results $\Rightarrow$ hop-size $M - N_x$.
     * Repeat the same procedure for next $x_i(n)$ block.
     * Overlap-add the intermediate results $y_i(n)$: (with same hop-size)
   - Stereo signal:
     * build complex sequence (stereo signal): $z(n) = x(n) + jy(n)$.
     * perform DFT transform on $z(n) \Rightarrow Z(k) = \text{DFT}\{z(n)\} = Z_R(k) + jZ_I(k)$.
       · $X(k) = \text{DFT}\{x(n)\} = X_R(k) + jX_I(k)$
       · $Y(k) = \text{DFT}\{y(n)\} = Y_R(k) + jY_I(k)$
       · $Z(k) = X_R(k) + jX_I(k) + j[Y_R(k) + jY_I(k)]$
       · $Z(k) = X_R(k) - Y_I(k) + j[X_I(k) + Y_R(k)]$
     * build the conjugate symmetric of $Z(k) \Rightarrow Z(N-k) = Z^*(k)$
       · $Z(N-k) = Z_R(N-k) + jZ_I(N-k) = Z^*(k)$
     * Derive the Real part of $X(k)$ from half the sum of real parts of $Z(k)$ and $Z^*(k)$.
       · $X_R(k) = \frac{1}{2} \cdot [Z_R(k) + Z_R(N-k)]$

- ∗ Derive the Real part of $Y(k)$ from half the sum of imaginary parts of $Z(k)$ and $Z^*(k)$.
  - · $Y_R(k) = \frac{1}{2} \cdot [Z_I(k) + Z_I(N-k)]$
- ∗ Derive the Imaginary part of $X(k)$ from half the subtraction of imaginary part of $Z^*(k)$ from imaginary part of $Z(k)$.
  - · $X_I(k) = \frac{1}{2} \cdot [Z_I(k) - Z_I(N-k)]$
- ∗ Derive the Imaginary part of $Y(k)$ from half the subtraction of real part of $Z(k)$ and from real part of $Z^*(k)$.
  - · $Y_I(k) = \frac{1}{2} \cdot [Z_R(N-k) - Z_R(k)]$

- Segmentation of input signal and impulse response:
  - ∗ Partition $h(n)$ in m blocks $h_i(n)$ of length N.
  - ∗ Specify the length of the DFT transform: $M = 2N$.
  - ∗ Zero-pad every $h_i(n)$ block to length M.
  - ∗ Perform DFT of every $h_i(n)$ block and store the result.
  - ∗ Decompose the input sequence in $x_i(n)$ blocks of length $N_x = M - N + 1$.
  - ∗ Read $x_i(n)$ block and zero-pad it to length M.
  - ∗ Perform DFT of $x_i(n)$ block: $X_i(k)$.
  - ∗ Multiply $X_i(k)$ with all $H_i(k)$
  - ∗ Read only real parts of IDFT on successive multiplications of $X_i(k)$ with all $H_i(k)$
  - ∗ Build intermediate result $y_i(n)$ with overlap-add of previous real parts results $\Rightarrow$ hop-size $M - N_x$.
  - ∗ Repeat the same procedure for next $x_i(n)$ block.
  - ∗ Overlap-add the intermediate results $y_i(n)$: (with same hop-size)

8. What is the processing delay of the fast convolution technique?

   Solution:

   - Based on the partitioning of the input signal and the impulse response and the following Fourier transform, the result of each single convolution is only available after a delay of one block of samples. Different methods to reduce computational complexity or overcome the block delay have been proposed [Soo90, Gar95, Ege96, Mül99, Mül01, Garc02]. These methods make use of a hybrid approach where the first part of the impulse response is used for time-domain convolution and the other parts are used for fast convolution in the frequency domain.

9. Write a Matlab program for fast convolution.

   Solution:

   - see Matlab script 'dasp_ex6_8_3'

10. How does quantization of the signal influence the roundoff noise behavior of an FIR filter?

    Solution:

    - The quantization of the signal inside a filter structure is responsible for the maximum dynamic range and determines the noise behavior of the filter. Owing to rounding operations in a filter structure, roundoff noise is produced.

9 FIR Filter Design by Frequency Sampling

1. Why is frequency sampling an important design method for audio equalizers? How do we sample magnitude and phase response?

   Solution:

   - Audio equalizers are used to modify the magnitude spectrum of the input signal in a given frequency range. In order to prevent phase distortion, linear phase FIR filters can be designed using the frequency sampling method.
   - Frequency sampling:
     * Specify desired magnitude response $|H(e^{j\Omega})| = |H(k)|$
     * Derive the spectrum of linear phase system
     * Choose the filter type to design, i.e. Type II:
       · real-valued impulse response
       · even length and even symmetry
       · real part of spectrum with even symmetry
       · imaginary part of spectrum with odd symmetry
     * Decompose the linear phase to derive real and imaginary parts of spectrum:
       $e^{-j\frac{N_F-1}{2}\Omega} = e^{-j2\pi\frac{N_F-1}{2}\frac{k}{N_F}} = \cos\left(2\pi\frac{N_F-1}{2}\frac{k}{N_F}\right) - j\sin\left(2\pi\frac{N_F-1}{2}\frac{k}{N_F}\right)$
     * Build $H(k)$ regarding symmetry properties
     * Derive $h(n)$ from N-point IDFT of $H(k)$

2. What is a linear phase frequency response of a system? What is the effect on an input signal passing through such a system?

   Solution:

   - $H(e^{j\Omega}) = A(e^{j\Omega})e^{-j\frac{N_F-1}{2}\Omega}$
   - For linear phase systems, all frequency components inside the input signal experience the same delay (constant group delay)

3. Explain the derivation of the magnitude and phase response for a linear phase FIR filter.

   Solution:

   - The magnitude $|H(e^{j\Omega})|$ is calculated by sampling in the frequency domain at equidistant places

     $$\frac{f}{F_S} = \frac{k}{N_F}, \qquad k = 0, 1, \ldots N_F - 1$$

     according to

     $$|H(e^{j\Omega})| = A(e^{j\Omega}), \qquad k = 0, 1, \ldots \frac{N_F}{2} - 1.$$

Hence, a filter can be designed by fulfilling conditions in the frequency domain. The linear phase is determined as

$$
\begin{aligned}
e^{-j\frac{N_F-1}{2}\Omega} &= e^{-j2\pi\frac{N_F-1}{2}\frac{k}{N_F}} \\
&= \cos\left(2\pi\frac{N_F-1}{2}\frac{k}{N_F}\right) - j\sin\left(2\pi\frac{N_F-1}{2}\frac{k}{N_F}\right), \\
k &= 0, 1, \ldots \frac{N_F}{2} - 1.
\end{aligned}
$$

4. What is the condition for a real-valued impulse response of even length N? What is the group delay?

   Solution:

   - $H(k = \frac{N_F}{2}) = 0 \wedge H(k) = H^*(N_F - k)$ with $k = 0, 1, \ldots \frac{N_F}{2} - 1$
   - Group delay in samples: $\tau_G = \frac{N_F-1}{2}$

5. Write a Matlab program for the design of an FIR filter and verify the example in the chapter.

   Solution:

   - see Matlab script 'dasp_ex6_9_5'

6. Now the desired frequency response is an ideal lowpass filter of length $N_F = 31$ with cutoff frequency $\Omega_c = \pi/2$, derive the impulse response of this system. What will be the result for $N_F = 32$ and $\Omega_c = \pi$?

   Solution:

   - see Matlab script 'dasp_ex6_9_6'

10 Multi-complementary Filter Bank

1. What is an octave-spaced frequency splitting and how can we design a filter bank for that task?

   Solution:

   - It is a decomposition into octave frequency bands matched to human ear
   - Apply successive lowpass/highpass decomposition into half-bands
   - Downsample the decomposed half-bands by factor 2

2. How can we perform aliasing-free sub-band processing? How can we achieve narrow transition bands for a filter bank? What is the computational complexity of an octave-spaced filter bank?

   Solution:

   - Applying multi-complementary filter bank
   - Design of modified filter bank with $\Omega_{c1} < \frac{\pi}{2}$, where $\Omega_{ck} = \frac{\pi}{3}2^{-k+1}$ and $k = 1, \ldots, N-1$
   - The computational complexity is: $C_{\text{tot}} = HC_1 + HC_2 + 2VC_1$
   - HC_1: horizontal complexity carried out by f_S
   - HC_2: horizontal complexity carried out by $\frac{f_S}{2}$
   - VC_1: vertical complexity carried out by f_S