EQ2300 Digital Signal Processing
Project Assignment, Fall 2015

About the Project

This project assignment should be documented in a short report. The report should be organized as a typical technical report. If you are unsure about how to write a report, check the report template found in the project zip-file on the course web-page. In particular, a list of answers to the 'tasks' is not a report. The report should be written so that a colleague of yours can **understand** and **reproduce** your results (however, since the colleague might use some other numerical tool than Matlab, Matlab code should **not** be included). The report should be written in English and will be graded based on technical content, presentation, layout, and language. The report should be at most **4 pages** (including figures), in either PDF or Word format, and submitted though the course web at KTH social by the deadline. You may make it shorter than 4 pages if you wish, if it is still understandable and answers all questions.

You can work on the project individually or in a group. However, each report can be authored by a maximum of two people.

The following is required for a passing grade:

- Solutions to all the project tasks, documented in a clear and concise report (not simply as a list of answers).
- One or two authors per report. The author names, personal numbers, email addresses and the course name "EQ2300 Digital Signal Processing" should be clearly written on the front page.
- If you collaborate in a group of more than two people, state this as a remark in the report, identifying the other individuals.
- The report should be unique and authored only by the authors stated on the front page.
- When solving the problem, you should only use Matlab functions that you understand well enough to be able to explain.

The reports will be graded with **P**ass or **F**ail. A passing grade is required for you to complete the course. The reports will be graded within two weeks from the due date. If the report does not pass, it should be revised and turned in before a new deadline. Note that if the report does not pass the second time, a new project has to be completed the following year. Reports that are turned in after the deadline will not be graded.

Deadline: Monday, Dec 14, 2015, 08:00 via the course web at KTH Social.

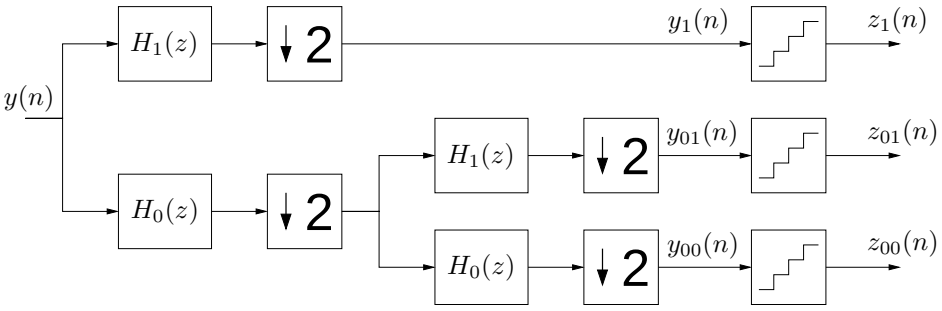

Figure 1: Analysis filter bank followed by quantization.

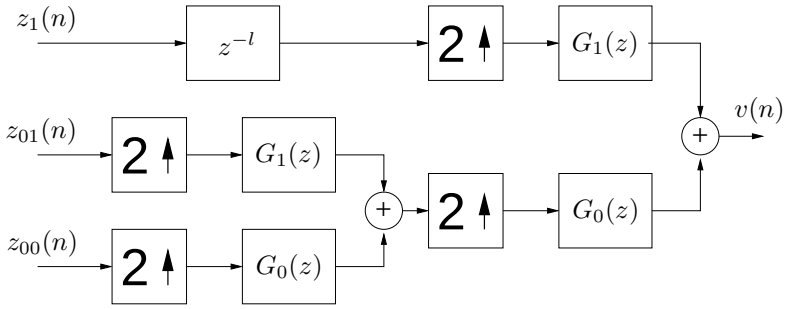

Figure 2: Synthesis filter bank.

Filter Banks and Sound Compression

Filter banks are used in some compression schemes for audio and video. The idea is to divide the signal into different frequency bands. Since the different frequency bands of the signal contain different amount of "information", different number of bits could be used to represent the different bands. This means that the data rate (number of bits per second) used to transmit or store the signal, can be reduced significantly.

In this project, you should implement the system shown in Figures 1–2. Here, $H_0(z)$ represents a lowpass filter and $H_1(z)$ the corresponding highpass filter. In a typical application, the signals $z_1(n)$, $z_{01}(n)$ and $z_{00}(n)$, are transmitted via radio or cable or stored in some kind of memory before the original signal is reconstructed. It is therefore important to limit the total number of bits used.

Filter Banks

In the course material (in the complementary reading and possibly the book), the following conditions are given that guarantee perfect reconstruction,

$$H_1(z) = G_0(-z) \tag{1a}$$

$$G_1(z) = -H_0(-z) \tag{1b}$$

$$G_0(z)H_0(z) - G_0(-z)H_0(-z) = 2z^{-l} \tag{1c}$$

where l denotes the delay of the reconstructed signal. Equivalently, if we work with the Fourier transform, the corresponding conditions are

$$H_1(f) = G_0(f - \tfrac{1}{2}) \tag{2a}$$
$$G_1(f) = -H_0(f - \tfrac{1}{2}) \tag{2b}$$
$$G_0(f)H_0(f) - G_0(f - \tfrac{1}{2})H_0(f - \tfrac{1}{2}) = 2e^{-j2\pi fl} \tag{2c}$$

A popular choice of filters is to let $G_0(z)H_0(z) = (1 + z^{-1})^{2p}Q(z)$, where the polynomial $Q(z)$ has degree $2p-2$ and is determined such that (1) holds for all z. When $Q(z)$ is determined, it is still possible to choose the factorization into $G_0(z)$ and $H_0(z)$ in different ways.

For simplicity, we recommend you to use $p = 2$ in this project, i.e., determine $Q(z) = q_0 + q_1z^{-1} + q_2z^{-2}$ such that $G_0(z)H_0(z) = (1 + z^{-1})^4Q(z)$ fulfills the constraints (1) with $l = 3$. Furthermore, to save time, you could concentrate on the choice $G_0(z) = \frac{1}{2}(1 + z^{-1})^2$.

Note that this choice of filters does not provide symmetry between $H_0(f)$ and $H_1(f)$, but $H_0(f)G_0(f)$ will be symmetric to $H_1(f)G_1(f)$.

Quantization

The three decimated signals, $y_1(n)$, $y_{01}(n)$ and $y_{00}(n)$, should be quantized, to reduce the number of bits needed to represent the signal. To use the full dynamic range of the quantizer, each signal should be scaled before the quantization and rescaled afterwards. You may read more about quantization in the course material.

In Matlab, the scaling and quantization of a signal `x` to `b` bits, could be implemented with the following lines of code (using the class `fixed` supplied with project zip-file on the homepage).

```
scaling=max(abs(x))/(1-pow2(-b));
xquantized = scaling*double(fixed(b, x/scaling));
```

The number of bits for each subsignal will determine the total number of bits needed to transmit the signal. The test signals have 8 bit resolution and 8 kHz sampling frequency, corresponding to a bit rate of 64 kbit/s. The goal of the project is to make this rate half of the original, i.e., 32 kbit/s. However, the quantization will introduce extra noise into the signal. We define the Signal to Quantization Noise Ratio (SQNR) as

$$\text{SQNR} = \frac{\text{E}[y^2(n)]}{\text{E}[(v(n) - y(n - L))^2]} ,$$

where L denotes the total delay in the system, from $y(n)$ to $v(n)$. The question now becomes how one can maximize the SQNR by a clever allocation of bits to the 3 decimated signals, while satisfying the overall rate-constraint of 32 kbit/s.

Task

Design and implement the system described above. Evaluate the system using the two test signals in the files `orinoccio.wav` and `thank.wav` found in the project zip-file on the course web-page. In particular, you should:

1. Verify that the filters you have derived provide perfect reconstruction in the absence of quantizers. Determine the total delay L of the system in Figures 1–2.

2. Plot the frequency response of the four filters, H_0, H_1, G_0 and G_1.

3. Discuss how the decimated signals are related to the input signal? Compare spectra and listen to the signals. Do not forget to use a sampling frequency matched to the data rate of each signal.

4. Describe how the total data rate (in kbit/s) depends on how many bits you allocate per sample in each frequency band. Obtain a closed form expression for this.

5. Choose *good* bit allocation schemes for the two signals (orinoccio and thank) and compute the corresponding SQNR values. Make sure that your total data rate does not exceed 32 kbits/s. Compare your result to the SQNR you obtain when quantizing the original signal to 4 bits per sample (which also correspond to 32 kbit/s). Report your bit allocation, your SQNR values (in dB), and explain why you chose your bit allocation.

6. Describe the perceived quality of the quantized, and reconstructed, signals (listen to the signals) and relate to the SQNR. Describe what you hear. Is the SQNR a good measure of quality for audio signal?

Practical Details

The test signals are found in the project zip-file. The sampling rate is 8 kHz for both signals. To read a `.wav` file into Matlab, use the command

`[y,Fs,nbits]=wavread('`*filename*`.wav');`

The command will find the file if it is in the current working directory or if it is in the Matlab search path. You can listen to `y` by writing `soundsc(y,Fs)` in Matlab. It may be a good idea to first try the system with a (simple) sine wave as input to get a better understanding.

Another important tool when designing and analyzing a system like this is spectral estimation to understand how the frequency properties of the signals are affected by the different operations. A function `plotspectrum` is included in the project zip-file, which estimates the power spectral density of a signal using the Welch's method and plots the result. The Matlab function `freqz` can be used to calculate the frequency response of a filter.

It is generally a good idea to verify (in Matlab) that the filter banks provide perfect reconstruction, before you add the quantization (the difference should be zero within the machine precision).