EE320 Assignment 1 — Filtering and Source Separation

1 Purpose

This assignment should give you some hands-on experience with designing and implementing discrete time ("digital") systems, and apply them to filtering audio signals.

The assignment will contribute 10% to your overall mark in EE320.

2 Completion, Submission, and Assessment

Please submit your assignment via MyPlace by Monday 5/2/2023, noon. Your submission should include

1. an m-file containing your implementations and generating any plots;
2. a brief document containing your answers and any supporting graphs; this does *not* have to be a formal report — answers should be brief, but *justifications are important*, and any supporting evidence (matlab plots, calculations / references to material covered in the lectures) should be included.
   Scans or photographs of your handwritten workings are perfectly acceptable.

You can receive assistance with your assignment during a number of surgeries which we will hold during December 2023 and January 2024. These will be typically Tuesdays 9-11am in R446. Detaiils be announced separately via MyPlace.

We will assess your solution based on the correctness of your implementations, and the justifications that you have provided. We may hold interviews to assess your understanding and to ensure that the submitted work is yours; in this case an interview will contribute to or overwrite the assessment of your submission. Therefore, you should only submit what is your own work, and be sure that you understand its purpose.

If you wish to work on this problem during the University holidays and do not have access to Matlab, you can consider the free Matlab clone Octave (https://www.gnu.org/software/octave/) which can do almost everything possibly except for playing audio files.

Important: this exercise involves audio files. If you listen to any audio files, particularly when using headphones, please ensure that you select a low volume, in case the signal amplitude is high!

3 Problem

Two ears / microphones pick up signals $x_1[n]$ and $x_2[n]$, which are a mix of two speakers and a sinusoidal interference. The two speakers are active concurrently ("cocktail party effect"), and the aim will be to retrieve their signals individually and without the interference. The solution will be based on temporal and spatial filtering.

The situation is illustrated in Fig. 1. We assume two microphones that are separated by $d = 17.15$cm, mimicking the distance between the two ears of an average (or at least weiss-sized) head. As sources, two speakers $s_i[n]$, $i = 1, 2$, simultaneously speak; we assume that these speakers are in the far-field and that the propagation is free-space, i.e. there are no reverberations or echos, and the speakers' signals arrive as planar wavefronts at the microphones. The impulse response between a source $s_i[n]$ and the microphone signal $x_m[n]$ therefore will only be a delay and gain factor. Additionally, there is a sinusoidal interferer present, which corrupts the two microphone signals $x_m[n]$, $m = 1, 2$. Overall, we have

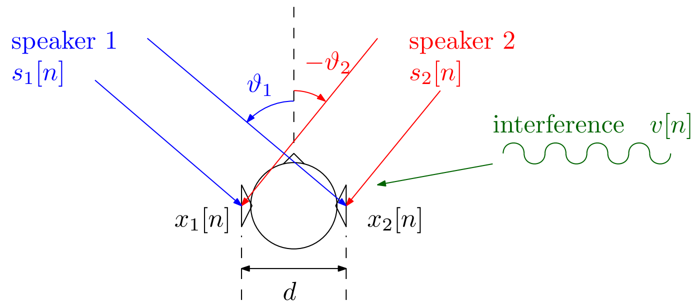

Fig. 1: Scenario with two speakers and sinusoidal interference at two microphone mimicking the positions of ears on a head.

$$\begin{aligned} x_m[n] &= \rho_{m,1}s_1[n-\tau_{m,1}] \;+\; \rho_{m,2}s_2[n-\tau_{m,2}] \;+\; \rho_{m,\mathrm{v}}\cdot v[n] && (1)\\ &= \rho_{m,1}\delta[n-\tau_{m,1}] * s_1[n] \;\;+\;\; \rho_{m,2}\delta[n-\tau_{m,2}] * s_2[n] \;\;+\;\; \rho_{m,\mathrm{v}}\cdot v[n]\;, && (2)\end{aligned}$$

where τ_j, $i=1,2$, are delays and $\rho_{m,i}$ and $\rho_{m,\mathrm{v}}$, $m,i=1,2$ are gain factors.

We are only interested in the relative delay and gain that a source experiences across the two microphone signals. For example, if we were only to receive a single signal, say $s_1[n]$, then for the two microphone signals $x_m[n]$, we have

$$\begin{aligned} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix} &= \begin{bmatrix} \rho_{1,1}\delta[n-\tau_{1,1}] \\ \rho_{2,1}\delta[n-\tau_{2,1}] \end{bmatrix} * s_1[n] \\ &= \begin{bmatrix} \frac{\rho_{1,1}}{\rho_{2,1}}\delta[n-\tau_{1,1}+\tau_{2,1}] \\ \delta[n] \end{bmatrix} * \rho_{2,1}s_1[n-\tau_{2,1}] && (3)\\ &= \begin{bmatrix} \delta[n] \\ \frac{\rho_{2,1}}{\rho_{1,1}}\delta[n-\tau_{2,1}+\tau_{1,1}] \end{bmatrix} * \rho_{1,1}s_1[n-\tau_{1,1}] && (4)\\ &= \begin{bmatrix} h_{1,1}[n] \\ h_{2,1}[n] \end{bmatrix} * \tilde{s}_1[n] && (5)\end{aligned}$$

We want to work with positive delays, such that a delay function $R_1\delta[n-T_1]$ is causal and has a gain $R_1 < 1$. For $\vartheta_1 > 0$, the path to $x_2[n]$ will have a longer delay, and hence a greater attenuation, and (3) applies. In the case $\vartheta_1 < 0$, we select representation (4). The general model in (5) ties this to impulse responses $h_{m,1}[n]$, where $H_{m,1}(z) \bullet\!\!-\!\!\circ\; h_{m,1}[n]$ are known as relative transfer functions. We would like to recover the potentially shifted and attenuated source signal $\tilde{s}_1[n]$.

In order to separate the sources, we require an estimate of the relative transfer function matrix

$$\boldsymbol{H}(z) = \begin{bmatrix} H_{1,1}(z) & H_{1,2}(z) \\ H_{2,1}(z) & H_{2,2}(z) \end{bmatrix}\,. \qquad (6)$$

We can then create an inverse matrix $\boldsymbol{G}(z) = \boldsymbol{H}^{-1}(z)$. Such an inverse does not always exist, but is possible for our parameter range, and detailed in Appendix A. With $\mathbf{G}[n] \circ\!\!-\!\!\bullet\; \boldsymbol{G}(z)$, we can the create two outputs $y'_m[n]$, $m=1,2$,

$$\begin{bmatrix} y'_1[n] \\ y'_2[n] \end{bmatrix} = \begin{bmatrix} g_{1,1}[n]*x_1[n]+g_{1,2}[n]*x_2[n] \\ g_{2,1}[n]*x_1[n]+g_{2,2}[n]*x_2[n] \end{bmatrix} = \mathbf{G}[n] * \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix}\,, \qquad (7)$$

in which the speakers should appear separated. The design of this filter $\mathbf{G}[n]$ is detailed in Fig. 2.

The separated signals $y'_m[n]$ are still subject to sinusoidal interference. To remove this interference, the aim is to construct a bandstop filter $Q(z)$ that is capable to suppressing this noise component while preserving the speech

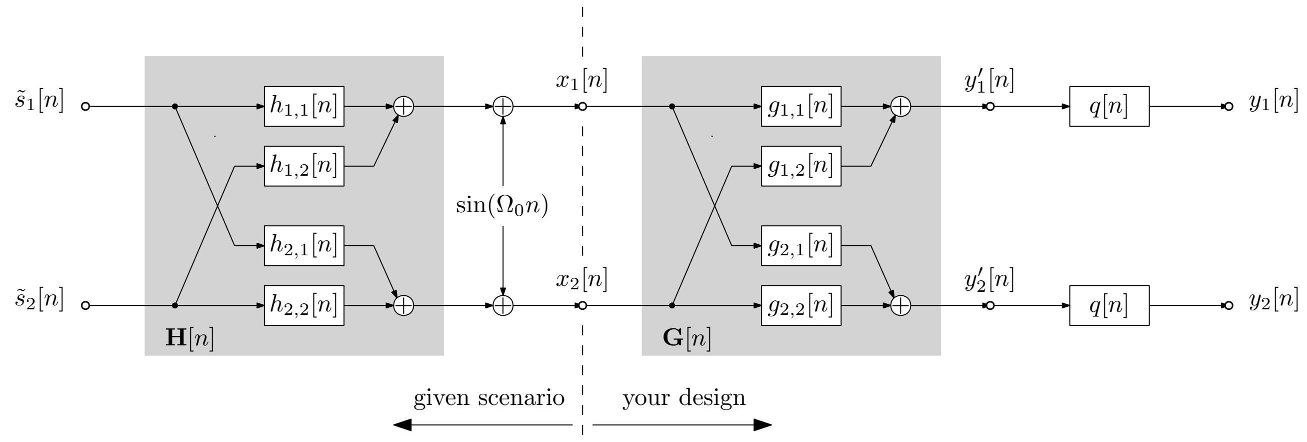

Fig. 2: Scenario with two speakers and sinusoidal interference captured by two microphone signals $x_1[n]$ and $x_2[n]$. The schematic shows the processing that you are expected to implement in order to separate the source signals and to suppress the interference, based on an analysis of the paths $h_{m,i}[n]$ and the interfernece frequency Ω_0.

signal as best as possible. The outputs $y_m[n]$ from this filtering operation in Fig. 2 should ideally sound close to the separated, noise-free speech signals $\tilde{s}_m[n]$, $m = 1, 2$.

4 Precursor

In Matlab, you can read in and play audio files, including stereo ones. To appreciate the spatial impression that you can get from stereo channels, feel free to try the following:

```
u1 = audioread('speaker1.wav');   % read in a single channel signal
u2 = filter([0 0 1],1,u1);        % delay by two sampling periods
fs = 8000;                        % sampling frequency in Hertz
sound([u1 u2],fs);                % play back stereo
```

The sound should appear to come slightly from one side. If you are unconvinced of the directionality, play

```
sound([u2 u1],fs);                % reverse stereo channels
```

This should sound different — the source should now appear to be on the other side.

5 Scenario and Analysis

Based on your student registration number, `StudRegNum`, as a parameter, you can obtain the signals $x_1[n]$ and $x_2[n]$ as the output of the function

```
[x1,x2] = AssignmentScenario(StudRegNum);
```

Please ensure that you have downloaded the files `speaker1.wav` and `speaker2.wav`, and that they are visible from your Matlab working directory. These files contain your source signals $s_i[n]$, $i = 1, 2$, and these files need to be read by the function `AssignmentScenario()`. If you listen to one output via e.g. `sound(x1,8000)`, you should hear a mix of two speakers and a tone.

5.1 Analysis of Sinusoidal Interference

For later interference cancellation, we require a good estimate of the interference frequency — you are asked to perform this in both the time and frequency domains.

Question 1 (Time domain analysis of interference frequency.) *Estimate the frequency f_0 or normalised angular frequency $\Omega_0 = 2\pi f_0/f_s$ of the interferer by inspecting the time domain waveform $x_1[n]$. This is best over a segment of the signal where the speakers are otherwise quiet (i.e. interference dominates), such as during the first couple of hundred samples of $x_1[n]$.* ■

Question 2 (Frequency domain analysis of interference frequency.) *Estimate the interference frequency by evaluating the discrete-time Fourier transform*

```
fs = 8000;                              % sampling frequency in Hertz
f = (0:length(x1)-1)/length(x1)*fs;     % frequency scale
plot(f,abs(fft(x1)));                   % discrete Fourier transform of x
xlabel('frequency f / [Hz]');           % label axes
ylabel('magnitude');
```

and compare the estimate to your result of Question 1. ■

5.2 Analysis of Delay/Angle of Arrival and Gain

The next questions will explore the parameters of the mixing system $\mathbf{H}[n]$.

Question 3 (Delay estimation in the time domain.) *Over the range of sample indices* $19500 < n \leq 21548$, *speaker 2 is relatively quiet. From the time domain signals $x_1[n]$ and $x_2[n]$, try to estimate the relative delay between these two signals.* ■

Question 4 (Delay and gain analysis in the Fourier domain.) *For the segments in Question 3, let $X_m(\mathrm{e}^{\mathrm{j}\Omega})$, $m = 1, 2$, be their discrete time Fourier transforms. Assuming that $X_1(\mathrm{e}^{\mathrm{j}\Omega}) \neq 0 \forall \Omega$, what would you expect for the quantities*

$$G(\Omega) = |\frac{X_2(\mathrm{e}^{\mathrm{j}\Omega})}{X_1(\mathrm{e}^{\mathrm{j}\Omega})}| \tag{8}$$

and

$$A(\Omega) = \angle\{\frac{X_2(\mathrm{e}^{\mathrm{j}\Omega})}{X_1(\mathrm{e}^{\mathrm{j}\Omega})}\} \text{ ?} \tag{9}$$

Justify your answer through analysis and Fourier transform properties. ■

Question 5 (Delay and gain estimation in the Fourier domain.) *For a signal consisting of N samples, command `fft()` approximates (closely enough for our purposes) the Fourier coefficients of the discrete time Fourier transform at a N of isolated, equispaced frequency points between DC and the sampling frequency. For example,*

```
fs = 8000;                       % sampling frequency in Hertz
Range = (19501:21548);           % window of samples to investigate
N = length(Range);               % number of samples
X1 = fft(x1(Range));             % Fourier coefficients via fft()
f = (0:(N-1))/N*fs;              % frequency scale
plot(f,abs(X1));                 % plot magnitude of coefficient
xlabel('frequency f / [Hz]');    % label axes
ylabel('magnitude');
```

will show you the magnitude of the Fourier coefficients obtained from $x_1[n]$. How can you utilise this to estimate the gain and delay difference between $x_1[n]$ and $x_2[n]$? W.r.t. Fig. 1, what is the estimated angle ϑ_1? ■

Question 6 (Estimation for 2nd speaker.) *The first speaker is quiet during the period $24800 < n \leq 25824$. Use this to estimate the gain and delay difference between $x_1[n]$ and $x_2[n]$ for speaker 2. W.r.t. Fig. 1, what is the estimated angle ϑ_2?* ■

6 Source Separation Design and Implementation

Question 7 (Estimation of relative transfer functions.) *Using the estimated relative gains and delays for the two speakers, determine the matrix of relative transfer functions $\boldsymbol{H}(z) \bullet\!\!-\!\!\circ \mathbf{H}[n]$.* ■

Question 8 (Construction of separation filters.) *Based on your estimated $\boldsymbol{H}(z)$, construct and implement the unmixing matrix $\boldsymbol{G}(z) = \boldsymbol{H}^{-1}(z)$, using e.g. the inversion method detailed in Appendix A. What do the output signals of $\boldsymbol{G}(z)$ sound like?* ■

Question 9 (Bonus – efficient implementation.) *Is there a way to implement $\boldsymbol{G}(z)$ more efficiently than through four separate infinite impulse response filters?* ■

7 Interference Suppression

To suppress the sinusoidal interference, we want to utilise an IIR notch filter with transfer function

$$Q(z) = \frac{(1 - e^{j\Omega_0} z^{-1})(1 - e^{-j\Omega_0} z^{-1})}{(1 - \gamma e^{j\Omega_0} z^{-1})(1 - \gamma e^{-j\Omega_0} z^{-1})} \tag{10}$$

to operate on $y'_m[n]$, $m = 1, 2$, and generate an output $y_m[n]$. This design is based on your interference frequency analysis in Questions 1 and 2. Initially assume a value of $\gamma = 0.9$.

Question 10 (Notch filter anaysis.) *Determine the denominator and numerator coefficients a_i and b_i, $i = 0, 1, 2$ of the notch filter when written as*

$$Q(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{a_0 + a_1 z^{-1} + a_2 z^{-2}} \quad ; \tag{11}$$

and plot the magnitude response $|Q(e^{j\Omega})|$ in Matlab. Also calculate the coefficients and magnitude response for a value of $\gamma = 0.99$. ■

Question 11 (Notch filter implementation and application.) *Implement the notch filter as sketched in Fig. 2. How do the output signals $y_1[n]$ and $y_2[n]$ sound when using the notch filters with different $\gamma = 0.9$ and $\gamma = 0.99$?* ■

Question 12 (Filter sequence.) *Very briefly justify whether or not it is possible to swap the notch filters $Q(z)$ with the unmixing system $\boldsymbol{G}(z)$.* ■

8 Final Comments

This assignment uses your signals and systems tools to take you beyond applications that you have encountered in the lectures. Please engage during the surgery / Q&A sessions to ensure that you have understood the task in all its aspects, and feel free to discuss your approach. You may find it interesting that the topic of this assignment takes you to the cutting edge of research, where relative transfer functions are making an impact in the audio word, and matrices of transfer functions are a generic way of describing and solving signals & systems problems.

A Appendix: Inverting a Matrix of Transfer Functions

For a standard 2×2 matrix, its inverse can be easily found as

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} , \tag{12}$$

as long as the determinant satisfies $ad - bc \neq 0$.

For a matrix $\boldsymbol{H}(z)$ of transfer functions as in (6), as long as the determinant $D(z) = H_{1,1}(z)H_{2,2}(z) - H_{2,1}(z)H_{1,2}(z)$ does not possess any zeros on the unit circle, we have

$$\boldsymbol{G}(z) = \frac{1}{D(z)} \begin{bmatrix} H_{2,2}(z) & -H_{1,2}(z) \\ -H_{2,1}(z) & H_{1,1}(z) \end{bmatrix} . \tag{13}$$

Example: For $\boldsymbol{H}(z)$ with $H_{1,1}(z) = H_{2,2}(z) = 1$, $H_{1,2} = \rho z^{-2}$ and $H_{2,1} = \rho z^{-1}$, the inverse $\boldsymbol{G}(z)$ is

$$\boldsymbol{G}(z) = \begin{bmatrix} \frac{1}{1-\rho^2 z^{-3}} & \frac{-\rho z^{-2}}{1-\rho^2 z^{-3}} \\ \frac{-\rho z^{-1}}{1-\rho^2 z^{-3}} & \frac{1}{1-\rho^2 z^{-3}} \end{bmatrix} . \tag{14}$$

The implementation could use the `filter()` command in Matlab, such that for inputs `yprime1` and `yprime2` and some defined factor `rho` with a modulus smaller than one, the outputs are

```
y1 = filter(-1,[1 0 0 -rho^2],yprime1)       + filter([0 0 rho],[1 0 0 -rho^2],yprime2);
y2 = filter([0 -rho],[1 0 0 -rho^2],yprime1)  + filter(-1,[1 0 0 -rho^2],yprime2);
```

S. Weiss, November 30, 2023