Speech Signal Processing
— Exercise 7 —
Beamforming

Timo Gerkmann, Kristina Tesch, Danilo Oliveira

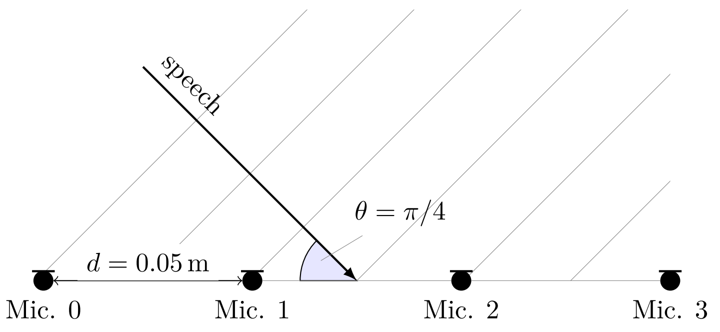

Figure 1: Array with $M = 4$ microphones with an inter-microphone distance of $d = 0.05\,\mathrm{m}$. The desired speech signal is arriving at the array at an angle of $\theta = \pi/4$

In this exercise session we will implement, use, and analyze two different beamformers in different acoustic scenarios.

1 Notation

Before starting with the actual exercise, here we briefly introduce the notation that we will use throughout this exercise:

- We will implement the beamformers in the STFT domain. The complex-valued STFT coefficients in segment l and frequency band k of the noisy recording in microphone i are denoted as

$$Y_i(k,l) = S_i(k,l) + V_i(k,l),$$

  where S_i is the clean speech signal and V_i is the additive noise recorded at microphone i.
- For a concise notation, the STFT coefficients of all 4 microphones are combined into a single vector:

$$\boldsymbol{Y}(k,l) = [Y_0(k,l), Y_1(k,l), Y_2(k,l), Y_3(k,l)]^{\mathrm{T}} = \boldsymbol{S}(k,l) + \boldsymbol{V}(k,l),$$

  with

$$\boldsymbol{S}(k,l) = [S_0(k,l), S_1(k,l), S_2(k,l), S_3(k,l)]^{\mathrm{T}}$$
$$\boldsymbol{V}(k,l) = [V_0(k,l), V_1(k,l), V_2(k,l), V_3(k,l)]^{\mathrm{T}}$$

- Note that we use boldface to denote vectors. $(\cdot)^{\mathrm{T}}$ is the transposition operator, while $(\cdot)^{\mathrm{H}}$ is the Hermitian operator, i.e., a transposition followed by taking the conjugate complex.

2 Delay-and-Sum Beamformer

- Load the 4 microphone signals from `noisySensor.wav`. The recordings contain clean speech arriving from an angle of $\theta = \pi/4$ as depicted in Figure 1 and are degraded by additive, spectrally white Gaussian noise. The goal of a beamformer is to reduce the noise while maintaining the desired speech sound.
- In this part, we want to implement the delay-and-sum beamformer in the STFT domain. Therefore, apply your STFT implementation to all the microphone recordings. Use a frame length of 128 ms, a frame shift of 32 ms, and use square-root Hann windows as the analysis window and synthesis window. Further, scale the synthesis window by 1/2.
  Question: Why is it required to scale the synthesis window by 1/2?
- For each frequency band k, compute the steering vector $\boldsymbol{a}(k)$ for the direction-of-arrival of the desired source $\theta = \pi/4$:
  1. We assume that the source-microphone distance is much larger than the inter-microphone distance (far-field assumption). Furthermore, we make the free-field assumption, i.e. there is no reverberation or reflections. Under these assumptions, the time delay τ_i between microphone 0 and microphone i is
  $$\tau_i = \frac{id}{c}\cos(\theta),$$
  where $c = 340\,\frac{\mathrm{m}}{\mathrm{s}}$ is the speed of sound.
  Question: Why is this the correct formula to compute the time delay?
  2. Using the time delays τ_i, compute the steering vector
  $$\boldsymbol{a}(k) = \left[1,\ \exp\left(-j\frac{2\pi k}{N}f_{\mathrm{s}}\tau_1\right),\ \exp\left(-j\frac{2\pi k}{N}f_{\mathrm{s}}\tau_2\right),\ \exp\left(-j\frac{2\pi k}{N}f_{\mathrm{s}}\tau_3\right)\right]^{\mathrm{T}}.$$
  Remember that we work with only half of the spectrum but need to consider all frequency bins when determining their total number N and make sure that k is an index and not a value in Hz.
- Align the desired speech in all microphones and add up the delayed signals to obtain an estimate of the clean speech $\widehat{S}$. In the spectral domain, this is achieved via:
  $$\widehat{S}(k,l) = \frac{1}{M}\boldsymbol{a}^{\mathrm{H}}(k)\boldsymbol{Y}(k,l)$$
  Question: Why do we normalize by the number of microphones M?
- Plot the magnitude spectograms of the noisy signal in the leftmost microphone ($i = 0$) and of the speech estimate.
- Transform the speech estimate into the time domain via the inverse STFT. Listen to the results and compare it to the noisy recording. Has the noise been reduced relative to the unprocessed reference signal of microphone 0?

3 Minimum Variance Distortionless Response (MVDR) Beamformer

- After implementing a rather simple delay-and-sum beamformer, let us now proceed to the more powerful MVDR beamformer. In each frequency band k, we define the MVDR filter as
  $$\boldsymbol{h}(k) = \frac{\boldsymbol{\Phi_V}^{-1}(k)\boldsymbol{a}(k)}{\boldsymbol{a}^{\mathrm{H}}(k)\boldsymbol{\Phi_V}^{-1}(k)\boldsymbol{a}(k)}.$$
- You already computed the steering vector $\boldsymbol{a}(k)$ for the delay-and-sum beamformer in the previous assignment. Accordingly, the only part of the MVDR that still needs to be computed is the noise covariance matrix
  $$\boldsymbol{\Phi_V}(k) = \mathrm{E}(\boldsymbol{V}(k)\boldsymbol{V}^{\mathrm{H}}(k)) \in \mathbb{C}^{4\times 4}$$
  The first second of the recording only contains noise, i.e. we have $\boldsymbol{Y}(k,l) = \boldsymbol{V}(k,l)$. If we approximate the expected value E() with the average over these noise only frames, in each frequency band k we get
  $$\boldsymbol{\Phi_V}(k) \approx \frac{1}{L}\sum_{l=0}^{L-1}\boldsymbol{Y}(k,l)\boldsymbol{Y}^{\mathrm{H}}(k,l),$$

where L is the number of noise-only segments at the beginning of the recording.

- Now compute $\boldsymbol{h}(k)$ using your estimate of $\boldsymbol{\Phi}_{\boldsymbol{V}}(k)$. For a numerically more stable implementation of the MVDR filter $\boldsymbol{h}(k)$, please use `numpy.linalg.solve(phiV,a)` to solve ${\boldsymbol{\Phi}_{\boldsymbol{V}}}^{-1}\boldsymbol{a}$ instead of explicitly computing the matrix inverse.
- The clean speech estimate is obtained via

$$\widehat{S}(k,l) = \boldsymbol{h}^{\mathrm{H}}(k)\boldsymbol{Y}(k,l)$$

- Transform the speech estimate into the time domain. Listen to the results and compare it to the noisy recording and the output of the delay-and-sum beamformer.
- If your implementations of the MVDR and the delay-and-sum beamformer are correct, the difference between the two estimates should be really small (in theory, the results should even be identical). So what is the advantage of the MVDR over the delay-and-sum beamformer? To show this, we will test the beamformers on two different recordings.
  - In the recordings that you have used so far, the noise is mutually independent between microphones. This is a typical model for sensor noise. Under this condition, it can be shown that the MVDR reduces to the delay-and-sum beamformer.
  - Now load the recordings `noisyIsotropic.wav`. In this recording, we have a spherically isotropic noise field, i.e., the noise is emitted from the surface of a sphere with the microphone array in its center. This is a typical example of diffuse noise, where the noise is not coming from a specific direction, but rather from every direction. In this kind of noise field, the noise is not uncorrelated between microphones anymore.
  - Finally, load the recordings `noisyDirectional.wav`. In this recording, we have a directional noise source in the far field with a direction-of-arrival of $\theta_N = \frac{150}{180}\pi$, i.e., 150 degrees (front-right). Also in this case the noise is correlated between microphones.
- Apply both beamformers to both recordings. Listen to the results. Describe differences between the MVDR estimate and the delay-and-sum beamformer estimate.
- Show the spectrograms of the enhanced speech for the MVDR and the delay-and-sum beamformer for different noise types in your report.

4 Implementation Tips

- Transposition $(a)^{\mathrm{T}}$:
  - `a.T`
- Hermitian $(a)^{\mathrm{H}}$:
  - `a.conj().T`
- Vector/matrix multiplication in Python:
  - With two-dimensional numpy arrays `a` and `b`: `c = a @ b` or `np.matmul`
- Vectorized computation of inner products:
  - Compute $A = [b_1^H b_1, b_2^H b_2, ..., b_n^H b_n]$ from a matrix $B = [b_1, b_2, .., b_n]$ of shape (m, n) with `np.einsum("ij,ji->i", B.conj().T, B)`

5 Bonus: Make your own recordings and apply beamformers to them

- Finally, you may also want to use our equipment to make your own recordings.
- Apply both beamformers to your own recordings and listen to the results.
- Compare the enhancement performance to the one you achieved for the provided recordings.
- Can you find reasons for these differences? What are potential solutions?