Homework 10

Due May 3 at 11 pm

Yves Greatti - yg390

1. (Hann window) In this problem we analyze the Hann window in the frequency domain.

   (a) Prove that for any vector $x \in \mathbb{C}^N$, if $y \in \mathbb{C}^N$ is defined as

   $$y\,[j] := x[j] \exp\left(\frac{i2\pi mj}{N}\right), \tag{1}$$

   for some integer m, then the DFT of y equals $\hat{y} = \hat{x}^{\downarrow m}$, where $\hat{x}$ is the DFT of x.

   (b) The Hann window $h \in \mathbb{C}^N$ of width $2w$ equals

   $$h\,[j] := \begin{cases} \frac{1}{2}\left(1 + \cos\left(\frac{\pi j}{w}\right)\right) & \text{if } |j| \leq w, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

   Use the result from part (a) to show that the DFT of h can be expressed as

   $$\hat{h} = \frac{1}{2}\hat{\pi} + \frac{1}{4}\hat{\pi}^{\downarrow -N/2w} + \frac{1}{4}\hat{\pi}^{\downarrow N/2w}. \tag{3}$$

   (c) Plot $\hat{h}$ as well as the different components in Eq. (3). Interpret what you see in terms of the desired properties of a windowing function.

2. (STFT inverse) In this problem we show a simple way to invert the STFT.

   (a) In the definition of the STFT set $w_{[\ell]}$ to be a rectangular window where all entries are equal to one, and let $\alpha_{\text{ov}} = 0.5$. Show that the STFT can be inverted using just two operations: applying the inverse DFT and subsampling.

   (b) What is the disadvantage of using this rectangular window?

3. (Haar wavelet) Define the discrete Haar wavelet $\mu_{2^s,p} \in \mathbb{R}^{2^n}$ at scale 2^s and position p by

   $$\mu_{2^s,p}[j] := \begin{cases} -1/\sqrt{2^s} & \text{if } j \in \{p \cdot 2^s, p \cdot 2^s + 1, \ldots, p \cdot 2^s + 2^{s-1} - 1\}, \\ 1/\sqrt{2^s} & \text{if } j \in \{p \cdot 2^s + 2^{s-1}, 2^s + 2^{s-1} + 1, \ldots, (p+1) \cdot 2^s - 1\}, \\ 0 & \text{otherwise,} \end{cases}$$

   where $0 < s \leq n$ and $0 \leq p \leq 2^{n-s} - 1$. Define the discrete Haar scaling function $\varphi_{2^s,p} \in \mathbb{R}^{2^n}$ at scale 2^s and position p by

   $$\varphi_{2^s,p}[j] = \begin{cases} 1/\sqrt{2^s} & \text{if } j \in \{p \cdot 2^s, p \cdots 2^s + 1, \ldots, (p+1) \cdot 2^s - 1\}, \\ 0 & \text{otherwise,} \end{cases}$$

   where $0 < s \leq n$ and $0 \leq p \leq 2^{n-s} - 1$. The code for this exercise is contained in the haar.py file. Include all generated plots in your submission.

(a) Define $V_0 := \mathbb{R}^{2^n}$. For $k > 0$, let $V_k \subset \mathbb{R}^{2^n}$ denote the subspace of all vectors that are constant on segments of size 2^k. That is

$$V_k := \{x \in \mathbb{R}^{2^n} : x[i] = x[j] \text{ if } \lfloor i/2^k \rfloor = \lfloor j/2^k \rfloor\}.$$

Give an orthonormal basis for V_k. What is the dimension of V_k?

(b) Fix $0 \leq k < n$, and note that $V_k \supset V_{k+1}$. Give an orthonormal basis for the set

$$W_{k+1} = \{x \in V_k : \langle x, y \rangle = 0 \text{ for all } y \in V_{k+1}\},$$

the orthogonal complement of V_{k+1} in V_k. Thus $V_k = V_{k+1} \oplus W_{k+1}$. What is the dimension of W_{k+1}?

(c) For $1 \leq k \leq n$ give an orthonormal basis for the set

$$W_{\leq k} = \{x \in \mathbb{R}^{2^n} : \langle x, y \rangle = 0 \text{ for all } y \in V_k\},$$

the orthogonal complement of V_k in $\mathbb{R}^{2^n}$. Thus $\mathbb{R}^{2^n} = V_k \oplus W_{\leq k}$. What is the dimension of $W_{\leq k}$?

(d) Complete the wavelet and scaling functions in *haar.py* that implement μ and φ above, respectively. See the comments for more details.

(e) Complete the projectV function that orthogonally projects a given vector onto V_k. [Hint: Consider averaging the values on each segment.]

(f) Complete the projectW function that orthogonally projects a given vector onto W_k. [Hint: You can use projectV.]

(g) Complete the function *wavelet_coeffs* which computes all of the (non-overlapping) wavelet coefficients of a given data vector at a given scale. See the comments for more details.

(h) Report the plots generated by the code, which apply your wavelet transform to some electrocardiogram data.

4. (Denoising with the STFT) In the lecture, we saw that STFT often yields sparse representation for a signal but dense representation for noise. Building on this, we derived hard thresholding (Algorithm 4.1 in notes) and block thresholding (Algorithm 4.2 in notes) to denoise signals. In this question, we will denoise audio signals. `audio_denoising.ipynb` contains skeleton code for the task. The notebook will download required dataset and contains other utility functions for loading data, plotting and playing the audio signals. You have to fill in the functions `get_block_L2_norm()` and `stft_denoising()`. Report all the plots generated by the script.