Homework 8

Due April 20 at 11 pm

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

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

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

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

where $0 < s \leq n$ and $0 \leq p \leq 2^{n-s} - 1$.

   (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 (proving that it is indeed orthonormal and a basis). What is the dimension of V_k?

   (b) Show that one can project onto V_k by averaging (explain what needs to be averaged).

   (c) Show that $V_{k+1} \subset V_k$

   (d) Fix $0 \leq k < n$. Consider 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, so that $V_k = V_{k+1} \oplus W_{k+1}$. Give an orthonormal basis for W_{k+1} (proving that it is indeed orthonormal and a basis).

   (e) 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 $\mathbb{R}^{2^n}$, so that $\mathbb{R}^{2^n} = V_k \oplus W_{\leq k}$. Give an orthonormal basis for $W_{\leq k}$ (proving that it is indeed orthonormal and a basis).

2. (Implementation of Haar wavelets) The code for this exercise is in the haar.py file. Include all generated plots in your submission.

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

   (b) Complete the projectV function that orthogonally projects a given vector onto V_k.

(c) Complete the projectW function that orthogonally projects a given vector onto W_k.

(d) 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.

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

3. (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 and block thresholding 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.

4. (Compression with wavelets) Similar to the STFT representation, a wavelet decomposition of an image is often sparse. In this question, we will exploit this sparsity to compress images. We will retain only the top x% of coefficients in the wavelet domain and set everything else to zero to perform compression. The support code provided in `image_wavelet.ipynb` will load an image, perform its wavelet decomposition, reconstruct the image back from the wavelet domain and compute the Peak signal-to-noise ratio (PSNR) between the reconstructed image and the original image.

   (a) Use `haar` wavelets as provided in the support code, but perform the wavelet decomposition for atleast `[2, 3, 4, 5]` levels. For each of these levels retain the top `[0.5, 1, 1.5, 2, 3, 4, 5, 10, 25, 50, 75]` percentage of coefficients and perform the reconstruction. You don't have to perform the thresholding in each sub-band separately - you can use the entire set of wavelet coefficients to determine the top x%. Plot the original image and reconstructed image side by side for each of the level and threshold using `visualize_image_and_recon()`

   (b) Plot the PSNR of reconstructed image with respect to the clean image for different threshold values. Include different levels of decomposition as different curve in the same graph.