

4.4.1 *Audio signals as vectors*

Discrete time audio signals consist of a series of audio samples in time. For example, typical digital audio files are sampled at a rate of 44.1 kHz (44,100 samples per second). If we have a finite length audio file with N samples, we can view it as vector of dimension N . For example, a 1 second (mono) file, \mathbf{a} , with the standard sampling rate would be a vector in \mathbb{R}^{44100} . Notice that we can construct the standard (canonical) basis for this high dimensional space as a series of N vectors, $\mathbf{u}_i \in \mathbb{R}^N$, $i = 1 \dots N$ where all entries of \mathbf{u}_i are zero, except the entry at position i , which is set to 1. The set of all these basis vectors can be written as an $N \times N$ diagonal matrix with all diagonal entries equal to 1. This is the $N \times N$ identity matrix. In digital signal processing we represent signals (such as audio signals) as infinite length time series. Thus $\text{audio}(n)$ is the sample at time n of the audio signal and we assume that $n \in \mathbb{Z}$.

4.4.2 *Loading audio signals into Matlab*

We have provided code to load audio files and play back the sound in `script1.m`. For each problem, you have to write your own code only in the `TODO` part in `script1.m`. In the code provided in `script1.m`, we first load an audio series with 12 seconds: 'a male voice counts to 10'. The audio file is recorded with a rate of 44.1kHz (44,100 samples per second). Second, we extract an audio clip with a duration around 0.5 seconds from the original 12 second clips. Finally, we downsample the audio clip by 10 (selecting 1 sample out of 10)¹. As a result, we will get an audio series with 2000 samples ($N = 2000$). Downsampling is used so that

¹ <https://en.wikipedia.org/wiki/Downsampling>

we can work with smaller vectors and the computations are simpler. In practice, most audio processing (e.g., mp3 coding) is based on dividing audio files into blocks and processing each block independently. We provide one demo in `script2.m` by processing audio blocks and remove high frequency components.

4.4.3 Audio Representation by Standard Basis

We define the delta (or impulse) function $\delta(n)$ as an infinite length signal such that $\delta(0) = 1$ and $\delta(n) = 0$ for any other $n \neq 0$. We can also define a shifted delta function $\delta_k(n) = \delta(n - k)$ such that $\delta_k(k) = 1$ and $\delta_k(n) = 0$ for $n \neq k$. Obviously for the purpose of this lab we are not manipulating infinite length signals, so you can consider any audio signal as a time sequence of length N and equivalently as a vector in \mathbb{R}^N . For the audio series we provided, $N = 2000$.

Problem 4.1. Bases for \mathbb{R}^N

Assume we consider all `audio(n)` signals of length N , use $\delta(n)$ to define a basis for \mathbb{R}^N . How do you write this basis in matrix form? Define this matrix as **A** in `script1.m` in Matlab. **Hint:** choose a small N (e.g., $N = 4$) and solve this by hand. The result is trivial and identical to an example we did in class.

Problem 4.2. Standard Basis for \mathbb{R}^N

Find the representation of `audio(n)` in terms of the standard basis (the Identity matrix, with dimension $N \times N$). In other words, if **b** is our audio signal, and **A** is our standard basis for \mathbb{R}^N , what is **x** if we have $\mathbf{Ax} = \mathbf{b}$?

4.4.4 Audio Representation by Discrete Cosine Transform (DCT) Basis

Problem 4.3. DCT Matrix as a Basis

In the section provided in `script1.m`, generate the DCT matrix \mathbf{A}_{DCT} of size N using `dctmtx()`. Next, write code to prove that \mathbf{A}_{DCT} is a basis for \mathbb{R}^N . Finally, write out your justification for your procedure.

Hints:

- If \mathbf{A}_{DCT} is a basis for \mathbb{R}^N , then what can we say about its columns?
- Functions that might be useful: `rank()`, `rref()`. This part may take some time with a matrix of dimension $N = 2000$.
- For this problem only, you are allowed to use a smaller N (e.g. $N = 100$) and prove that the columns of \mathbf{A}_{DCT} form a basis in your code. However, you must generate a DCT matrix with dimension $N = 2000$ for Problems 4.4 through 4.6

Problem 4.4. DCT Vectors as Elementary Audio Signals

The DCT vectors can be viewed as elementary audio signals, each representing some elementary audio frequency. They are ordered by frequency, with the first column in the matrix \mathbf{A}_{DCT} corresponding to the lowest frequency. Write code to play back each of the basis vectors to show that they lead to increasing frequency.

Hints:

- Each column of the DCT matrix \mathbf{A}_{DCT} corresponds to one frequency.

- You can access the i^{th} column of a matrix \mathbf{A}_{DCT} in Matlab by using `A(:, i)`
- Use Matlab Function `soundsc()` to play the sound for each basis vectors
- The Matlab function `pause(n)` can be used after a `soundsc()` function call to pause execution by n seconds. This can be used to play sound clips one at a time with pauses in between to avoid overlapping sounds.

Problem 4.5. *Representing Signals in Terms of a New Basis*

We are given a signal `audio(n)`: find its representation in terms of the DCT basis (the columns of \mathbf{A}_{DCT}) by formulating and solving a linear system of equations.

Hints:

- Functions that might be useful: `linsolve()`.
- Recall how we represented this problem in Problem 4.2. Solve for \mathbf{x}_{DCT} with this new problem formulation.

Problem 4.6. *Audio Processing*

Design an audio processor that removes the higher frequencies from audio and plays it back. For example, since the given audio signal has length $N = 2000$ you could remove the largest $K = 1000$ frequencies. Next, repeat the experiment by removing the lowest K frequencies. Provide code for both versions of this experiment. Describe your findings by listening to the processed audio signal with different amounts of low or high frequencies removed (in other words, change K and repeat for each of the two versions of the experiment). Compare your results between removing the low or the high frequency vectors.

Hints:

- Here you will be making changes to \mathbf{x}_{DCT} . Ensure that for each variation of the experiments you work with a clean copy of \mathbf{x}_{DCT} .
- We mentioned in Problem 4.4 that each column of \mathbf{A}_{DCT} corresponds to an elementary audio signal. That means in our DCT basis representation, each element in \mathbf{x}_{DCT} represents the contribution of the i^{th} elementary audio signal. What should we do to $\mathbf{x}_{\text{DCT}}(i)$ to remove that contribution?
- Using the formulations described in Problems 4.2 and 4.5, how do we obtain an audio sample \mathbf{b} from the DCT representation \mathbf{x}_{DCT} ?
- You may find `script2.m` helpful as a reference for this problem