Coding #04: Fourier Transforms

10 Points Possible

2025/9/24

Attempt 1

In Progress

NEXT UP: Submit Assignment

Add Comment

Unlimited Attempts Allowed

2025/9/17 to 2025/10/1

Details

📂 Case File #4 — *The Spy Who Played Notes*

Narrative

The city's soundtrack is a mess — a noisy audio recording has landed in your inbox. But intelligence reports say it contains **one of ten possible musical sequences**. The problem: the audio is too distorted to simply correlate with the known sequences. Your only clue lies in the **frequency domain**: the magnitude of the Fourier transform reveals the notes that actually occurred.

Your mission is to **identify the correct sequence** using the given partial Fourier data.

Provided Tools

Provided is a `.p` file: **get_spy_audio.p (https://ufl.instructure.com/courses/540008/files/100159365?wrap=1) (https://ufl.instructure.com/courses/540008/files/100159365/download?download_frd=1)**

and required audio data: **midi_audio.zip (https://ufl.instructure.com/courses/540008/files/98989613?wrap=1) (https://ufl.instructure.com/courses/540008/files/98989613/download?download_frd=1)**

The function is defined by:

```
[X, zn, freq, Fs] = get_spy_audio('########');
```

Where:

Variable	Description	Units / Notes
`X`	Matrix of 10 audio sequences, each column is a sequence	size `N × 10`, unitless (normalized audio)
`zn`	Noisy audio vector to analyze	length `N`, same units as `X`
`freq`	Frequencies of interest (target notes)	Hz
`Fs`	Sampling frequency	Hz

`'########'` is your UFID.

Students should not open the underlying `.m` code for `X` and `zn` — it is part of the “classified intelligence.”

Problem 1 — Partial DTFT Magnitudes

Compute the **partial DTFT magnitude** of the noisy signal `zn` at the frequencies provided in `freq`.

Function header provided to students:

```
function Zn = partialDTFT(x, freq, Fs)
% x    : Nx1 vector of audio samples
% freq : Kx1 vector of frequencies (Hz) at which to compute DTFT
% Fs   : Sampling frequency (Hz)
% Zn   : Kx1 vector of DTFT magnitudes at the requested frequencies
```

Requirements:

1. Compute the partial DTFT as:

$$\mathrm{Zn}(k) = \left| \sum_{n=0}^{N-1} x[n]\, e^{-j2\pi f_k n/F_s} \right|$$

for $k = 1, \ldots, K$.

2. **Output:** `Zn` is a column vector of length `K` containing the magnitudes.
3. Units: Magnitude is **unitless** (since the input `x` is normalized).

Problem 2 — Identify the Matching Audio Sequence

Once you have `Zn`, compare it to the known audio sequences in `X` to identify which column matches the noisy signal.

Hints / Requirements:

1. Compute a **similarity metric** between `Zn` and the magnitude of the DTFTs of each column of `X` (use only the frequencies in

`freq`).

2. Output the **index of the matching column**:

```
function indexMatch = findMatchingAudio(X, Z, freq, Fs)
% X          : Nx10 matrix of known audio sequences
% Z          : Nx1 vector of audio sequence to match
% freq       : Kx1 vector of frequencies (Hz)
% Fs         : Sampling frequency (Hz)
% indexMatch : scalar integer, the column in X that best matches Z
```

3. The index `indexMatch` must be an integer from 1 to 10 corresponding to the correct sequence in `X`.
4. **Unitless:** `indexMatch` is purely a label — no physical units.

Deliverables:

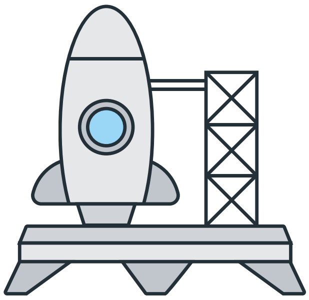
- `partialDTFT.m` — computes the magnitude of the noisy audio at the requested frequencies.
- `findMatchingAudio.m` — returns the correct column index.
- A `.mat` file containing `indexMatch` for grading.

Detective Notes

- This case emphasizes **frequency-domain identification** rather than simple inner-product correlation.
- The “notes” in the audio may be slightly shifted or noisy, so **using only magnitude at known frequencies** avoids false negatives due to phase differences.
- Students are explicitly **given the header for the partial DTFT function** to scaffold implementation.

Choose a submission type

Upload Google Drive More



Choose a file to upload

File permitted: M, MAT

or

Canvas Files

Submit Assignment