Assignment 3 COMP 546

Instructor: Mike Langer

Posted Tues. March 27, 2018.
Due: Fri. April 13, 2018 at midnight

Instructions

- The assignment has a total of 100 points.
- Late penalty 20 % of your score per day.
- Many of these questions require computing Fourier transforms in Matlab using `fft`. See the Matlab notes in Q3 for a few tips.
- For all figures that you submit, label the axes. Be sure to give meaningful definitions of frequency. For some questions, frequency will be in k cycles per N pixels, but for other questions it will be ω cycles per second (Hz).

Questions

1. **(15 points)**

   The classical neuroscience method for discovering receptive field profiles of cells is to choose a type of stimulus (dot, line, moving line, etc) and to manually vary the parameters of that stimulus (orientation, position, etc) and examine which parameters gave the best response. See video from Hubel and Wiesel in lecture 5. The problem with such a method is that one has to know in advance what type of stimulus to use and what parameters to vary.

   The past few decades, researchers have developed better methods. One method is to generate noise stimuli and to ask which noise stimuli give a large response from the cell. We will discuss how this is done ("spike triggered averaging") in lecture 21. Here you will explore the basic underlying concept.

   Write a script `spikeTriggeredAverage.m` that does the following:

   - Define a filter (cell, template). It can be anything you like, e.g. a Gabor or DOG, a disk, a handwritten or letter, a picture of a face. Make it small, say 32 x 32. This filter is the receptive field profile of the cell. Here, you know the filter but in a real situation you don't, and it is what you are trying to figure out!
   - Use Matlab's `rand(N,N)` function to generate a sequence of random white noise images of the same size as your filter. For each noise image, take the inner product of the filter and the noise image. This is the linear response of your filter.
   - Compute the average of *those images that produce a response greater than some chosen threshold* τ. Compare this average image to the filter. The idea here is that the experimenter provides the noise images and measures the responses, and then estimates the filter to be the average response.

- Vary the threshold and examine (either visually or with numbers) how accurately the average image approximates the filter, as the threshold varies. Explain your findings.

Note that there are no spikes in this question, and the name of the script is there just to connect it to the concept of spike triggered averaging which is quite similar to this.

2. **(15 points)**

   This question will give you some experience with psychometric functions, thresholds, and contrast sensitivity. It will also help you understand the frequency domain properties of filtering.

   Suppose you are given two 1D signals $I_1(x)$ and $I_2(x)$. One is just Gaussian noise (mean 0, and some variance that you can choose), which is independent between pixels. The other is the sum of Gaussian noise and a sine function of some variable frequency and variable amplitude. (We ignore phase.)

   The function `idealObserver.m` takes the two images as input, and decides whether the sine was contained in the first or the second image. The function `gaborObserver.m` does the same, but uses one Gabor family only. You will need to read the code to see what these functions do. In particular, note that the Gabor defined by `make1DGabor.m` has real and imaginary parts.

   The script `contrastSensitivity.m` generates two figures. Figure 1 shows psychometric functions for many different frequencies. Each psychometric function defines a threshold (see the code). The thresholds are plotted in Figure 2.

   The contrast sensitivity function is roughly constant for the ideal observer, but not for the Gabor observer. *Explain why.* Be specific and relate the shape of the Gabor observer's contrast sensitivity function to the amplitude spectrum of that Gabor.

   Hint: Examine the amplitude spectra of the noise functions. Verify that the *average* amplitude spectrum of a set of noise functions is constant over frequency. Also examine the amplitude spectrum of the sine function and the amplitude spectrum of your Gabor, and verify that they have the properties mentioned in lecture 17.

3. **(15 points)**

   Recall the filters $B(x)$ and $D(x)$ from the lectures. $B(x)$ is non-ideal lowpass, and the $D(x)$ is non-ideal bandpass. Design an $N \times 1$ filter that uses only three non-zero samples like $B(x)$, but that is non-ideal *highpass. Derive the Fourier transform of the filter by hand and verify correctness by showing a Matlab plot.* Use an $N = 64$ or some other large power of 2.

   Hints for the Matlab part:

   The `fft` function takes as input a $1 \times N$ matrix and considers values at indices $1, \ldots, N$ to hold x (or t) values $0, \cdots, N-1$. Note that this leads commonly to "off by one" errors. Typically we work with filters that are defined not just on positive values of x but also negative values. One can define such a filter in Matlab in two ways:

- Consider indices 1 to $N/2$ to hold the value $x = 0$ to $\frac{N}{2} - 1$, and indices $N/2$ to N to hold values from $x = -\frac{N}{2}$ to -1. For example, suppose `N=8`. You could define your local difference `D` by `[0 -1 0 0 0 0 0 1]`.
- Think of the filter as periodic, and Consider indices 1 to N to hold values corresponding to $x = -\frac{N}{2}$ to $\frac{N}{2} - 1$, so that the origin $x = 0$ is at array index $\frac{N}{2} + 1$. Then apply the Matlab function `fftshift` which shifts the array to the scheme above.

  For the example of `N=8` would define `D` by `fftshift([0 0 0 1 0 -1 0 0])`. Try this for yourself now and verify that `fft` applied to this matrix gives the result stated in the lecture for the Fourier transform for `D`.

4. **(15 points)**

   Typical images and audio signals are not periodic, but the Fourier transform treats them as if they are. This can lead to artifacts where the left and right boundaries produce high frequency components. For example, an image that consists of a single linear ramp intensity rising from the left to right edge would be expected to have almost no high frequency components. Yet it does have high frequency components because of the intensity edge defined at the image boundary. (Here I am arguing about 1D images only, say a row in an image. We did not discuss 2D Fourier transform this year.)

   A common way to reduce this artifact is to *multiply* the image by a "window" whose weight goes to 0 at the image boundaries. For example, in 1D image with N pixels, the window might be defined:

$$W(x) = 1 - \cos(\frac{2\pi}{N}\, x).$$

   This window removes the edge at the image boundary, by forcing the windowed image to fall off continuously to zero on both sides of the boundary.

   What is the effect of such a window on the Fourier transform of the signal? Demonstrate this effect using:

   - *a sinusoid whose wavelength is much larger than N (and which gives a roughly linear ramp as mentioned above)*
   - *a sinusoid whose wavelength is much smaller than N.*

   Hint: recall the second version of the convolution theorem presented in lecture 17.

5. **(15 points)**

   This question is relevant for understanding voiced speech. (See lecture 19.)

   Let Δt be an integer, and consider a *sampling* function $f(t)$ which is a sum of delta functions separated by Δt,

$$f(t) = \sum_{j=0}^{\frac{T}{\Delta t}-1} \delta(t - j\ \Delta t).$$

   where $f(t)$ is defined on $0, 1, \ldots, T-1$. You may assume that $\frac{T}{\Delta t}$ is an integer.

Show that the Fourier transform of $f(t)$ is

$$\hat{f}(\omega) = \frac{T}{\Delta t} \sum_{m=0}^{\frac{T}{\Delta t}-1} \delta(\omega - m\, \frac{T}{\Delta t}).$$

Verify your derivation using Matlab and give a plot to show an example.

6. **(25 points)**

The purpose of this question is give you some experience analysis the frequency components of real sounds. You will need to record sounds and analyze them. You should be able to record sounds with your laptop or cell phone microphone (or ask a friend to borrow theirs).

Record several clips such as:

- voiced vowel sound e.g. "eeeeeeee"; (Use a different vowels in different sound clips)
- voiced sounds that vary the frequency, e.g. when asking a question, one often raises the frequency, *right?*
- a complex sound such as crumpling paper or scraping.

Feel free to use your imagination and have some fun with this.

Use Matlab functions `audioread` and `sound` to read and play the sound files. For each sound,

(a) Take the Fourier transform (`fft`) and plot the log of the amplitude spectrum. Only plot the first 5 kHz. Convert the frequencies to Hz.

(b) For comparison, plot the spectrogram using the Matlab function
`spectrogram(signal, samplesPerBlock, 0, freqPerBlock, samplingRate ).`
Set `freqPerBlock` equal to `samplesPerBlock`. For the voice sounds, give examples of values that resolve the formants or glottal pulses. (The sampling rate is determined by your microphone and software. It is typically 44,100 for high quality audio.)

Describe the features that you see (or were expecting to see but don't) in your computed amplitude spectra and spectrograms. When appropriate, relate these to the features of voiced sounds discussed in lecture 19.