PSYCH-GA.2211/NEURL-GA.2201 – Fall 2025
Mathematical Tools for Neural and Cognitive Science

Homework 3

Due: 28 Oct 2025
(late homeworks penalized 10% per day)

See the course web site for submission details. Reminder: rather than using the functions `pinv()` and `norm()`, use the linear algebra tools we learned in class. Do yourself a favor, and don't wait until the day before the due date... *start now* !

1. **Fourier transform of periodic signals.**

   (a) Generate and plot a signal (vector) of length 2048 containing the function $f(n) = \text{mod}(n, 32)/32$. This waveform is known as a "sawtooth". Note that it's periodic, with a period of 32 samples. Play it through your computer speakers or headphones, using the function `sound`. Assuming you use the default playback rate of 8192 samples/sec, what is the duration of your signal (in sec), what is the duration of one cycle of the sawtooth, and what is the frequency of the repetitions (in cycles/sec)? What note on the piano is closest to this (consult Google, or a piano!)? In Python you will need a package such as sounddevice, which allows you to set the sampling rate to 8192 samples/sec.

   (b) Compute and plot the Fourier amplitude spectrum, centered at zero. Label the x-axis using units of cycles/sec. What do you see? What about the plot indicates that the signal is periodic, and how can you determine the period? Test your assertion by generating another sawtooth signal, with a period of 24 samples and note what changes in the plot of the Fourier amplitude spectrum.

   (c) Generate and plot another periodic signal, with function $g(n) = (1+cos(2\pi\, 64\, n/2048))^2$. Again, compute and plot the Fourier amplitude spectrum, centered at zero. How does this differ from the plot of the Fourier spectrum of $f(n)$? Is the periodicity the same or different? Compare the shape of the function by plotting a period of this function on top of one period of the previous function. What in the Fourier spectrum indicates that the waveform shape is different? Play this signal using the `sound` function. In what ways does it sound the same, and in what ways does it sound different, compared to the example from the previous part? (You might also want to compare it to the sound of functions that are the same shape, but have a different period.)

2. **Neurons in visual cortex.** The response properties of neurons in primary visual cortex (area V1) are often described using linear filters. We'll examine a one-dimensional cross-section of the most common choice, known as a "Gabor filter" (named after Electrical Engineer/Physicist Denis Gabor, who developed it in 1946 for use in signal processing).

   (a) Create a one-dimensional linear filter that is a product of a Gaussian and a sinusoid, $\exp\left(-\frac{n^2}{2\sigma^2}\right)\cos(\omega n)$, with parameters $\sigma = 3.5$ samples and $\omega = 2\pi * 10/64$ radians/sample. The filter should contain 31 samples, and the Gaussian should be centered on the middle (16th) sample. Plot the filter to verify that it looks like what you'd expect. Plot the amplitude of the Fourier transform of this filter, sampled at 64 locations

(MATLAB's `fft` function takes an optional additional argument). What kind of filter is this? Why does it have this shape, and how is the shape related to the choice of parameters (σ, ω)? Specifically, how does the Fourier amplitude change if you alter each of these parameters?

(b) If you were to convolve this filter with sinusoids of different frequencies, which of them would produce a response with the largest amplitude? Obtain this answer by reasoning about the equation defining the filter (above), and also by finding the maximum of the computed Fourier amplitudes (using the `max` function), and verify that the answers are the same. Compute the *period* of this sinusoid, measured in units of sample spacing, and verify by eye that this is matched to the oscillations in your plot of the filter. Which two sinusoids would produce responses with about 25% of this maximal amplitude?

(c) Create three unit-amplitude 64-sample sinusoidal signals at the three frequencies (low, mid, high) that you found in part (b). Convolve the filter with each, and verify that the amplitude of the response is approximately consistent with the answers you gave in part (b). (Hint: to estimate amplitude, you'll either need to project the response onto sine and cosine of the appropriate frequency, or compute the DFT of the response and measure the amplitude at the appropriate frequency.)

(d) Verify the convolution theorem. Apply the Fourier transform to each of your three stimuli. Multiply each by the Fourier transform of the Gabor filter. Inverse Fourier transform the results and verify that the imaginary part is zero, and the real part is equal to the result you obtain from the convolution.

3. **Deconvolution of the Haemodynamic Response.** Neuronal activity causes local changes in deoxyhemoglobin concentration in the blood, which can be measured using functional magnetic resonance imaging (fMRI). One drawback of fMRI is that the haemodynamic response (blood flow in response to neural activity) is much slower than the underlying neural responses. We can model the delay and spread of the measurements relative to the neural signals using a linear shift-invariant system:

$$r(n) = \sum_k x(n-k)h(k), \tag{1}$$

where $x(n)$ is an input signal delivered over time (for example, a sequence of light intensities), $h(k)$ is the haemodynamic response to a single light flash at time $k = 0$ (i.e., the impulse response of the MRI measurement), and $r(n)$ is the MRI response to the full input signal.

In the file `hrfDeconv.mat`, you will find a response vector r and an input vector x containing a sequence of impulses (indicating flashes of light). Your goal is to estimate the HRF, h, from the data. Each of these signals are sampled at 1 Hz. Plot vectors r and x versus time to get a sense for the data. Use the `stem` command (or `plt.stem` in Python) for x, and label the x-axis.

(a) Convolution is linear, and thus we can re-write the equation above as a matrix multiplication, $r = Xh$, where h is a vector of length M, N is the length of the input x, and X is an $[N + M - 1] \times M$ matrix. Write a matlab function `createConvMat`, that takes as arguments an input vector x and M (the dimensionality of h) and generates a matrix X such that the response $r = Xh$ is as defined in Eq. (1) for any h. Verify that the matrix generated by your function produces the same response as Matlab's `conv` function when applied to a few random h vectors of length $M = 15$. Visualize the matrix X as an image (evaluate `imagesc(X)` in MATLAB or `plt.imshow` in Python), and describe its structure.

(b) Now, given the X generated by your function for $M = 15$, solve for h by formulating a least-squares regression problem:

$$h_{\text{opt}} = \arg\min_{h} ||r - Xh||^2$$

Plot h_{opt} as a function of time (label your x-axis, including units). How would you describe it? How long does it last?

(c) It's often easier to understand an LSI system by viewing it in the frequency domain. Plot the power-spectrum of the HRF (i.e. $|\mathcal{F}(h)|^2$, where $\mathcal{F}(h)$ is the Fourier transform of the HRF). Plot this with the zero frequency (DC) in the middle (in Matlab you can use a built-in function called `fftshift`), and label the x axis, in Hz. Based on this plot, what kind of filter is the HRF? Specifically, which frequencies does it allow to pass, and which does it block?

(d) Use the convolution theorem to now find h_{opt} by working in the Fourier domain. You will need to use the matlab functions `fft` and `ifft`. Remember to be careful about how many samples you choose to have in your fft. Based on the operations you have done, what can you say about when this method will fail? On the same graph, plot the HRF impulse response you recovered from parts (b) and (d).

4. **Sampling and aliasing.** Load the file `myMeasurements.mat` into matlab. It contains a vector, `sig`, containing voltage values measured from an EEG electrode, sampled at 512 Hz. Plot `sig` as a function of vector `time` (time, in seconds, that you should compute).

(a) Examine your EEG result in the frequency domain. Plot the log of the magnitude (amplitude) of the Fourier transform of the original signal, over the range $[-N/2, (N/2)-1]$ (use `fftshift`). By convention, the "Delta" band corresponds to frequencies less than 4 Hz, "Theta" band is 4-7 Hz, "Alpha" band 8-15 Hz, and "Beta" is 16-31 Hz. For these data, which band shows the strongest signal? Is there any power in frequencies outside of these known bands, and if so can you explain the origin of this part of the signal?

(b) Write a function `signalBand = reconstructBand(sig,bandName)` that reconstructs the signal using only frequency components from the band corresponding to the string `bandName` (e.g., for `bandName = 'Delta'` the reconstruction should be a sum of sinusoids with frequencies from 0-4Hz). Run this function and plot the first 5 seconds of each of the bands defined in (a).

(c) The voltage signal is densely sampled and thus expensive to store. Create a *subsampled* version of the signal, which contains every *sixteenth* value. Is the subsampling operation linear? Shift-invariant? For the first second of data, plot the subsampled signal, against the corresponding entries of the `time` vector, on top of the original signal (use Matlab's `hold` function, and plot original with flag 'ko-' and subsampled version with flag 'r*-'). How does the reduced version of the data look, compared to the original? Does it provide a good summary of the original measurements? Explain.

(d) Plot the Fourier magnitude for downsampled signal, after upsampling it back to full size (i.e., make a full-size signal filled with zeros, and set every 4th sample equal to sixteen times the corresponding subsampled value). What is the relationship between these plots and the original frequency plot? Compare the power in each of the frequency bands defined in (a) to those of the original signal. Which band has changed the most (in proportion to its original power)? Plot this band of the original and sampled signals (using the function you wrote in part (b)) - can you see any difference?