UW ENGR 510
Frequency Domain Signal Processing, Autumn 2024

DUE: 11:59pm, Friday December 6th
Please submit a .pdf file with plots, discussion questions, and code blocks (when requested), **in addition to** a Python (.py) or Jupyter Notebook (.ipynb) file with the filename `signal_process.py` or `signal_process.ipynb`.

Since most questions will request code blocks while others will use the autograder, we note that you are welcome to include as much or as little in your code file as you would like, as long as (1) you provide code in your pdf when it is requested, and (2) your code passes the autograder successfully.

Exercise 1: Building an orthogonal cosine basis

Consider the space of bounded functions on the interval $[0, 1]$ with the following inner product

$$\langle f, g \rangle = \int_0^1 f(x)g(x)dx.$$

Bounded means that the magnitude of both f and g never exceed some fixed large number on the interval $[0, 1]$.

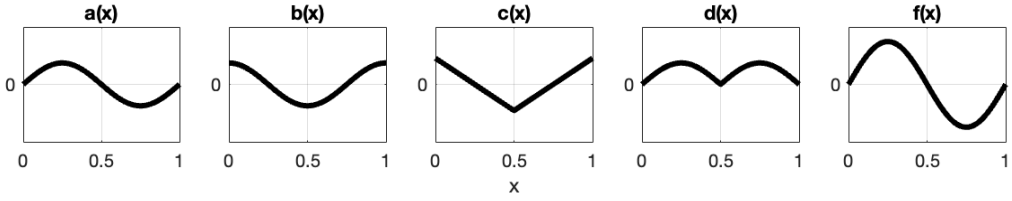
(a) Show that the functions $\cos(\pi mx)$ and $\cos(\pi nx)$ for non-negative integers m and n are orthogonal (using the inner product above) for all $m \neq n$. (You may look up in a table the solution to the integral.)

(b) Plot $\cos(\pi mx)$ on the interval $[0, 1]$ with $dx = 0.01$ for $m = 0, 1, 2, 3, 4$, and 5. Verify numerically, using trapezoidal integration (i.e. `scipy.integrate.trapezoid`), that $\cos(\pi mx)$ and $\cos(\pi nx)$ are orthogonal for the following (m, n) pairs: $(1, 4)$, $(2, 6)$, and $(3, 15)$. Please do so by printing the values of the integrals that you compute. Note that this means that you should provide 3 integral values total (one for each pair). Please submit plots, print outputs, and your code as a part of your pdf submission.

Note that we have an *infinite* set of orthogonal functions, which each represent a unique and orthogonal *vector direction* in the inner product space of bounded functions on $[0, 1]$. We are starting to build an infinite dimensional vector space (called a Hilbert space) for representing functions. (Note that we will eventually need to include sine functions as well.)

Exercise 2: Inner product, revisited

Recall that the inner product can be used as a measure of distance or angle between two vectors or functions. This should be interpreted to mean how close to linearly dependent or independent the two functions or vectors are.



Using the functions in the figure provided, visually estimate the inner products, and *justify your answer* with 1-2 sentences. Example estimates are 'zero', 'close to zero', 'one', 'close to one', 'greater than one', etc.

(a) $\langle a, b \rangle$

(b) $\langle b, c \rangle$

(c) $\langle a, d \rangle$

(d) $\langle a, f \rangle$

The norm of each function on the domain $x = [0, 1]$ is 1, except for the function f(x), which has a norm of 2. The axis limits on the plots are all the same.

Exercise 3: Image compression

This is the only exercise in this assignment that will use the autograder. Please remember to name your variables accordingly and to check the results of the autograder.

Load the image `recorder.jpg` as a NumPy array and convert to grayscale. You can do so with:

```
import numpy as np
import matplotlib.pyplot as plt
from skimage.color import rgb2gray

# Read image:
B = plt.imread("recorder.jpg")

# Convert to grayscale:
B_gray = rgb2gray(B)
```

Note that this code uses the package `scikit-image`.

(a) Use `numpy.fft` to compute the 2-dimensional FFT of the image. Submit the resulting Fourier coefficients as the NumPy array `A1` to the autograder. Note that the entries of `A1` should be complex.

```
AUTOGRADER FORMATTING INFORMATION:
type(A1) = numpy.ndarray
A1.shape = (958, 1875)
```

(b) Plot the PSD (power spectral density) of the Fourier coefficients on a log scale. Include units in your axis labels.

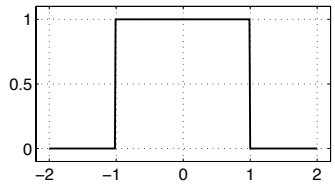
(c) Design a compression threshold to keep a particular percentage of the original Fourier coefficients. Show the compressed images with 10%, 1%, and 0.2% of the original Fourier coefficients.

(d) Describe, in words, how the images change with each compression, and how that relates to truncating Fourier coefficients.

(e) Compute the L_2 norm of the *error* between the new compressed images and the original image. Submit as a 3-element NumPy array called `A2`, whose entries contain the error for 10%, 1%, and 0.2% respectively.

```
AUTOGRADER FORMATTING INFORMATION:
type(A2) = numpy.ndarray
A2.shape = (3,)
```

Exercise 4: Fourier Series vs Fourier Transform



(a) Compute, by hand, the **Fourier series** representation for the square wave defined on $x \in [-2, 2]$:

$$f(x) = \begin{cases} 0 & -2 < x < -1 \\ 1 & -1 \leq x \leq 1 \\ 0 & 1 < x < 2 \end{cases}$$

(b) In Matlab or python, plot the mode coefficients a_n and b_n for the first 100 cosine and sine modes (i.e. for the first $n = 1$ to $n = 100$).

(c) Plot the approximation using $n = 10$ modes on top of the true square wave. Try $\Delta x = 0.01$.

(d) Compute the **Fourier transform** by hand for this function.

(e) In a few sentences, explain the difference between the Fourier series and the Fourier transform.

Remember to include a copy of your plotting code in your pdf submission (code block, copy and paste, screenshot, or save a copy of your script as pdf).

Exercise 5: Using spectrograms for species ID

Just like in so many other tasks, one of the tricky parts of creating spectograms is tuning the parameters, whether it is being interpreted by a human or used as input data for a machine learning task. For this exercise, you'll read two blog posts from the Cornell lab of ornithology about how they determined some of the key spectrograms parameters for a bird species classification task, and then answer the questions below.

- Optional: You may want to read this introductory article, for context.
- The first blog article is about creating the spectrograms. A note on *hop size*: One parameter for creating spectrograms is how much over overlap is desired between each window. Hop size = Number of samples in window - number of samples that overlap.
- The second blog article is about processing the spectrogram to improve image classification model performance.

(a) What are some of the main reasons the authors choose to use spectrograms as the model input, rather than using the time-domain audio waveform directly?

(b) What are the horizontal and vertical axes of a spectrogram (what would you use as axis labels?)

(c) STFT window length is one of the most important parameters when building a spectrogram. With this in mind, what would be the challenge when dealing with a bird species with a song that changes very quickly in time (e.g. fast tempo or trills) and a wide frequency ranges (e.g. both very high and very low notes)?

(d) Why would the authors choose exactly 256 or 512 samples per window (window length), rather than 250 or 500?

(e) The authors mention that there is a trade-off between time resolution and frequency resolution, which is affected by window length. Explain this trade-off, using 1-3 sentences (and diagrams or math if helpful), to someone who is familiar with the FFT but not with the STFT or spectrograms.

(f) In the second blog post, the authors mention normalizing the signal. Why do they do this?

(g) The authors discuss logarithmic scaling, or decibel scaling, as the default for spectrograms. Why do we usually use logarithmic or other scaling rather than using the raw amplitudes?

(h) How did the authors take advantage of the three color bands (red, green, blue) common in computer vision models, even though they only have one color? (The blog illustrations use a blue/yellow colormap to help human eyes distinguish high/low intensity, but it's likely they used grayscale images for training).

(i) In 1-3 sentences, what do the authors conclude about custom scaling methods for spectrograms, beyond the standard approach of logarithmic scaling?

Exercise 6: Connections to your own work

(a) Is there an application related to your own field of engineering where it would be useful to perform classification on frequency-domain data? (This may be currently done, or something you imagine would be useful to do.)

(b) Speculatively (you do not need to research what is currently being done), would you first try this task using the raw time-domain data, an FFT of the data, or a spectrogram (STFT) of the data, or some other approach? Justify your answer with 1-3 sentences.